Joint Spatial-Temporal Representation for Host Intrusion Detection System

Hao Li[1,2], Zehui Wang[1,2], Shang Shang[1,2], Zhengwei Jiang[1,2], Qiuyun Wang[1,2], Fangli Ren[1,2(✉)], and Baoxu Liu[1,2]

[1] Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China
{lihao,renfangli}@iie.ac.cn
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing 100049, China

Abstract. Host-based Intrusion Detection Systems (HIDS) collect host system logs and generate alerts when the host is attacked. However, existing research fails to adequately capture the spatiotemporal relationships within host behaviors, limiting the accuracy of their representation and modeling. To address this, we propose a spatiotemporal graph representation learning method. This method extracts key data from system logs to construct provenance graphs. A spatiotemporal joint encoder decomposes features along spatial and temporal dimensions independently, then aggregates them to capture spatiotemporal dependencies, explicitly modeling these relationships in host behavior. Experiments on the Streamspot and DARPA-Theia datasets show that the proposed method effectively captures interaction patterns and outperforms baseline models in recall rate, false positive rate, and other evaluation metrics.

Keywords: Network Attack Detection · Graph Deep learning · Spatial-Temporal Representation.

1 Introduction

Network attack detection has been a key focus in computer security research. As hosts are primary targets, these attacks often leave traceable footprints within the host systems. Host-based Intrusion Detection Systems (HIDS) can effectively capture these traces and identify attack behaviors, playing a critical role in network security defense [1].

To better represent the contextual information of host events, existing research models log data as a Directed Acyclic Graph (DAG), known as the provenance graph, as shown in Fig.1. In this graph, nodes represent system entities, and edges represent system events. The Provenance Graph organizes the spatial (semantic) and temporal (interaction) information of each system entity, offering a crucial framework for capturing complex behavior patterns [2–8].

However, existing methods typically represent semantic and temporal information implicitly, which limits effective modeling of system behavior. In log

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

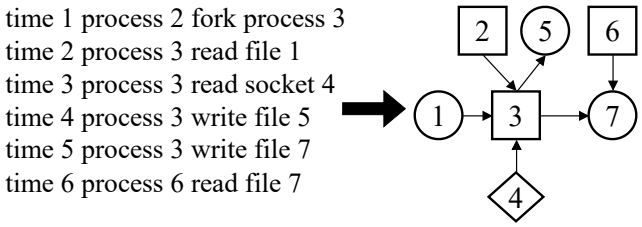

Fig. 1. System Log to Provenance Graph

data, substantial semantic information is often sparse, providing limited details on activity. Additionally, temporal interactions exhibit multiple structural forms, complicating the learning of underlying spatiotemporal dependencies.

In this study, we construct a provenance graph by extracting key information from logs and explicitly model the semantic and interaction data of entities using Graph Attention Networks (GAT) [9] and Temporal Graph Networks (TGN) [10]. A spatiotemporal joint encoder decomposes features along spatial and temporal dimensions, followed by joint aggregation to capture spatiotemporal dependencies. Attack behaviors are assessed by combining reconstruction errors from the decoder with anomalous node associations. Experimental results demonstrate that the proposed method significantly outperforms baseline models across several public datasets.

The main contributions of this study can be summarized as follows:

- Provenance Graph Construction: We extracted key entity information from system logs and constructed a provenance graph to describe system behavior.
- Joint Spatial-Temporal Representation: We independently decomposed features along spatial and temporal dimensions, then aggregated them jointly to capture spatiotemporal dependencies, explicitly modeling these relationships in host behavior.
- Attack Behavior Association: During the training phase, we use an encoder to learn normal behaviors. In the testing phase, the system assigns higher Reconstruction Error (RE) scores to deviations from the known baseline behavior and associates attack behaviors through queue linking. This approach improves the accuracy and precision of the alert system.

The paper is structured as follows: Section 2 reviews related work on host anomaly detection; Section 3 introduces the proposed model and its technical details; Section 4 presents data processing and experimental results and discusses the findings; and the final section concludes the paper and outlines future work.

2 Related work

In recent years, Advanced Persistent Threats (APT) have become a significant challenge in cybersecurity. Intrusion Detection Systems (IDS) play a crucial role

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

in detecting attack behaviors within host systems. IDS are primarily classified into two categories: anomaly-based detection [2–6, 11–16] and heuristic rule-based detection [17–23]. Effectively detecting APT attacks is challenging, as these attacks often exploit zero-day vulnerabilities. Defense systems relying on known threat intelligence struggle to identify new or unknown attack strategies. Current research mainly focuses on anomaly-based IDS, which analyze system behavior data—such as network traffic, process activity, and file operations—to establish baselines of normal behavior. This method is effective in identifying deviations from normal host activities.

Recently, embedding techniques have been widely applied in Intrusion Detection Systems. These methods typically use machine learning models, including neural networks and n-gram models, to convert logs into vector representations. Unicorn [4] employs graph sketching technology to summarize system behavior over long periods, addressing slow attacks that occur over extended time spans. Deeplog [27] models system logs as natural language sequences using Long Short-Term Memory (LSTM) networks. Attack2Vec [28] uses a time-based word embedding model to simulate attack steps. Flash [29] combines Graph Neural Networks (GNN) and Word2Vec embeddings in a lightweight classifier to detect potential malicious activities.

However, existing research often fails to explicitly model the spatiotemporal relationships within host behaviors, making it difficult to learn and accurately represent system activities. As a result, current methods need further enhancement to explicitly capture spatiotemporal dependencies. This paper proposes a spatiotemporal joint encoder method that independently extracts features in both the spatial and temporal dimensions, followed by joint aggregation to capture spatiotemporal dependencies. The decoder's reconstruction error and its correlation with anomalous nodes are then used to accurately detect attack behaviors.

3 Proposed Method

Fig.2 illustrates the construction process of Joint Spatial-Temporal Host Intrusion Detection System (JST-HIDS), designed to analyze host behavior across spatial and temporal dimensions. It covers behavioral logs, such as network, file, and process activities. Through spatiotemporal fusion representation, JST-HIDS learns host behavior patterns effectively, enhancing anomaly detection and generating precise alerts by correlating attack behaviors.

The process begins by constructing a comprehensive traceability graph from the log data. Next, the spatiotemporal encoder independently extracts features from both spatial and temporal dimensions. Through joint aggregation, it captures spatiotemporal dependencies. The decoder identifies anomalous events by predicting and reconstruction error from the true values. Finally, event associations are analyzed to detect attack behaviors, thereby enhancing the accuracy of attack analysis.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

3.1 Construction of the Provenance Graph

This study constructs the provenance graph by extracting log data from the host's kernel auditing systems. These systems track all operational activities within the host, including processes, files, network addresses, and other entities.

In the provenance graph, nodes represent system entities (e.g., processes, files, network addresses), while edges indicate control flow (e.g., process 1 calling process 2) and data flow (e.g., process 1 writing to file 1) between entities. We focus on system events related to attack steps, as listed in Table1. The provenance graph organizes and presents the causal relationships and contextual information of each system entity.

Table 1. System Events and Type

events	type
process and file	read,write,create,chmod,rename
process and process	fork,clone,execve,pipe
process and ip	sendto,recvfrom,recvmsg,sendmsg

In-depth analysis of the provenance graph reveals correlations between attack behaviors and their temporal sequence, offering a comprehensive view for constructing an attack chain. This analysis aids in accurately identifying attack patterns and provides essential support for attack tracing and the development of defense strategies.

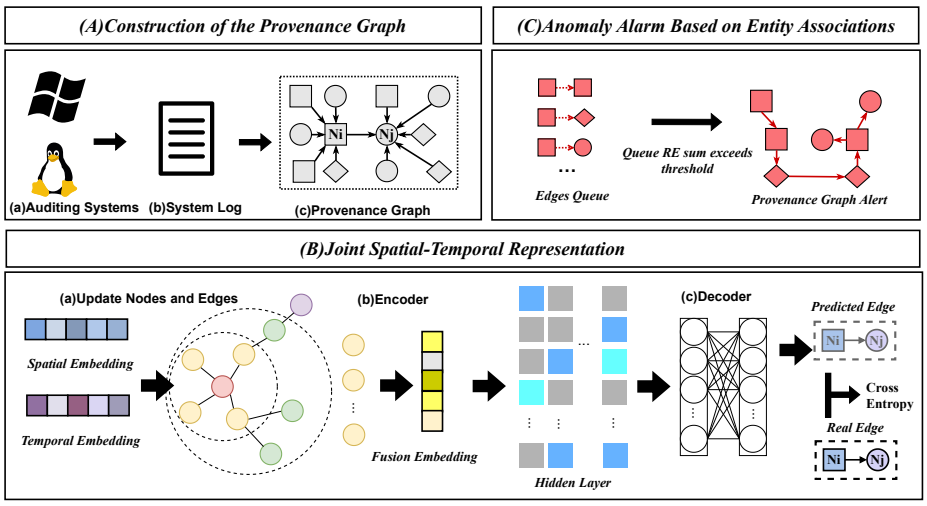

Fig. 2. Construction Process of JST-HIDS

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

3.2 Spatiotemporal Joint Representation

The provenance graph contains both temporal and spatial information, as shown in Fig.2. In the anomaly detection process, a benign provenance graph is used to simultaneously train the encoder and decoder. The goal of the training is to minimize the error between the actual edge type (when a new edge appears in the graph) and the type predicted by the decoder from the embedded vector of the edge. This error is referred to as the Reconstruction Error (RE).

During the testing phase, if the graph structure encoded by the edge's embedding closely matches the structure observed from benign system activities in a similar temporal context, the decoder assigns a small reconstruction error. Otherwise, the decoder assigns a larger reconstruction error, with the magnitude of the error reflecting the degree of deviation from the normal baseline. Host behaviors in the test samples with a reconstruction error exceeding a predefined threshold are identified by the detection module as anomalous behaviors.

Spatial Semantic Embedding of Entities in the Provenance Graph. Entities in the provenance graph are categorized into three types: processes, files, and network addresses. First, the features of these entities (such as command lines, file paths, and IP addresses) are converted into natural language sentences. For example, the file path "/usr/local/bin/app" is transformed into the sentence "usr local bin app". Next, the Natural Language Processing (NLP) technique [30] is applied to eliminate meaningless non-natural language components, such as the hash string found in the path. Finally, FastText [31] is used to project these sentences into numerical vectors, generating the semantic embedding f_i for each node i.

Entity Temporal Interaction Information Embedding. To capture the evolving characteristics of entity interactions in a dynamic provenance graph, node states record the interaction history of each node. At time t, the historical information of the provenance graph is represented by the state vector $s_i(t)$ for each node i, which compresses the entity's interaction history. When a new node is added to the graph, its state is initialized as a zero vector and updated as it interacts with other nodes. For instance, at time t, when a new edge e_{ij} appears, indicating an interaction between nodes i and j, the message m transmitted by edge e_{ij}, such as a process writing to a file, is calculated. This message updates the state vectors $s_i(t)$ and $s_j(t)$ for nodes i and j, respectively. The messages are computed from the perspectives of both the source node and the destination node:

$$m_i\left(t_1\right) = msg\left(s_i\left(t_0\right), s_j\left(t_0\right), \Delta t, e_{ij}\left(t_1\right)\right) \tag{1}$$

$$m_j\left(t_1\right) = msg\left(s_j\left(t_0\right), s_i\left(t_0\right), \Delta t, e_{ij}\left(t_1\right)\right) \tag{2}$$

To thoroughly capture the characteristics of dynamic provenance graphs as they change over time, node states are used to record each node's interaction

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

history. The Gated Recurrent Unit (GRU) model [32] is employed to continuously update the node states through iterative processing:

$$s_i\left(t_1\right) = GRU\left(m_i\left(t_1\right), s_i\left(t_0\right)\right) \tag{3}$$

The encoder uses a Temporal Graph Network (TGN) [10] architecture to encode source graph features into edge embeddings. At time t, the model, based on Graph Neural Networks (GNNs), generates edge embeddings f_{e_t} for new edges:

$$f_i\left(t\right) = Encoder\left(i, t\right) \tag{4}$$

$$f_i\left(t\right) = \sum_{j \in \Gamma_i^K([0,t])} h\left(s_i\left(t\right), s_j\left(t\right), e_{ij}\right) \tag{5}$$

$$f_{e_t} = f_{v_{src}} + f_{v_{des}} \tag{6}$$

This formula represents the embedding of an edge f_{e_t} at time t in a dynamic graph. The edge embedding is computed as the sum of the embeddings of the source node $f_{v_{src}}$ and the destination node $f_{v_{des}}$.

Learning Trajectories with Spatiotemporal Joint Embedding. Estimating probability distributions related to time and semantic spaces is a complex task, but it plays a crucial role in capturing spatiotemporal patterns of host behavior. However, traditional methods extract the semantic space information distribution $p(s|t)$ under time conditions. However, the representations learned from $p(s|t)$ are often implicit and limited, leading to a reduction in expressive power. To address this issue, we integrate semantic space and interaction temporal features, learning the spatiotemporal correlations. To effectively capture the correlations in both spatial and temporal domains, we adopt a generalized graph neural network (GNN) model with a three-layer structure [33], which aids in learning the joint spatiotemporal distribution. Specifically, the information update process of vertex v is as follows:

$$m_{vn}^i = ReLu\left(f_n^i + f_{e_{vn}}^i + f_{\omega_{vn}}^i\right) + \epsilon \tag{7}$$

The message m_{vn}^i is updated by combining the features of node n, the edge e_{vn}, and the normalized edge weight $f_{\omega_{vn}}$, followed by a ReLU activation and adding noise ϵ.

$$m_{e_{vn}}^i = MLP\left(Concate\left(f_v^i, f_n^i\right)\right) \tag{8}$$

The edge message $m_{e_{vn}}^i$ is updated by concatenating the features of nodes v and n, and passing the result through Multi-Layer Perceptron(MLP) [33].

$$m_v^i = \sum_{n \in V} \frac{exp(\alpha m_{vn})}{\sum_{j \in N(V)} exp(\alpha m_{vj})} \tag{9}$$

The equation computes the aggregated message m_v^i for node v at the i-th iteration, where m_{vn} represents the information exchanged between node v and its

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

neighboring nodes n. This is done by performing a weighted sum of the information from all neighboring nodes n in the graph, with the weights α determined by an attention mechanism based on the information.

$$f_{e_{vn}}^{i+1} = MLP(f_{e_{vn}}^{i} + c \cdot \left|\left|f_{e_{vn}}^{i}\right|\right|_2 \cdot \frac{m_{e_{vn}}^{i}}{\left|\left|m_{e_{vn}}^{i}\right|\right|_2}) \tag{10}$$

The equation describes the feature update process for edge e_{vn} at the i+1-th iteration. $f_{e_{vn}}^{i}$ represents the feature of edge e_{vn} at the i-th iteration, while $m_{e_{vn}}^{i}$ denotes the information exchanged between node v and its neighboring node n. Both $\left|\left|f_{e_{vn}}^{i}\right|\right|_2$ and $\left|\left|m_{e_{vn}}^{i}\right|\right|_2$ are normalized using their respective L2 norms. The constant c is used to regulate the normalization factor. All these components are processed through MLP, resulting in the updated feature $f_{e_{vn}}^{i+1}$ for edge e_{vn}.

$$f_{v}^{i+1} = MLP(f_{v}^{i} + c \cdot \left|\left|f_{v}^{i}\right|\right|_2 \cdot \frac{m_{v}^{i}}{\left|\left|m_{v}^{i}\right|\right|_2}) \tag{11}$$

This equation updates the feature f_v^{i+1} of node v, following a similar approach as the previous equation. Here, f_v^i represents the feature of node v at the i-th iteration, and m_v^i denotes the aggregated message for node v. Both $\left|\left|f_v^i\right|\right|_2$ and $\left|\left|m_v^i\right|\right|_2$ are normalized using their respective L2 norms. After normalizing these features and messages, they are processed through MLP, resulting in the updated feature f_v^{i+1} for node v.

$$R_p\left(e_{vn}\right) = MLP(concat(f_v, f_{e_{vn}}, f_n)) \tag{12}$$

This equation is used to make the final prediction for edge e_{vn}. First, the feature vectors of the two nodes v and n along with the edge feature vector $f_{e_{vn}}$ are concatenated to form a unified vector containing all relevant information. This concatenated vector is then processed through MLP to predict the edge, with the resulting output representing the probability distribution over the possible edge types.

$$RE = CrossEntropy\left(R_p\left(e_{vn}\right), R_r\left(e_{vn}\right)\right) \tag{13}$$

The output of the decoder is a vector $R_p\left(e_{vn}\right)$, representing the model's predicted probabilities for edge e_t being one of the edge types. During training, the model's optimization objective is to minimize the RE between $R_r\left(e_{vn}\right)$ and the observed edge type $R_p\left(e_{vn}\right)$ from the benign provenance graph. During testing, for edges whose structural and temporal context information is similar to those learned from the benign provenance graph, the model assigns a lower RE score. Conversely, if there are edges in the graph that significantly deviate from the known normal system baseline behavior, the model assigns a higher RE score.

3.3 Anomaly Alarm Based on Entity Associations

After obtaining the reconstruction errors for each event, we construct a queue based on the associative relationships between system entities to accurately pinpoint anomalous events. The queue stores the related events, and anomalies are

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

detected by calculating the sum of the RE values of the nodes within the queue. An associative queue Q is used to store events based on the association between system entities. If there is a strong association between events E_i and E_j, they will be stored in the same queue:

$$Q = (E_i, E_j) \mid \text{Edge}(E_i, E_j) = 1 \tag{14}$$

Where $\text{Edge}(E_i, E_j) = 1$ indicates that there is a direct edge or calling relationship between events E_i and E_j, meaning that one event calls or triggers the other.

$$RE_{\text{queue}} = \sum_{(E_i, E_j) \in Q} RE(E_i) \tag{15}$$

$$\text{If } RE_{\text{queue}} \geq T_{\text{alarm}}, \text{ then trigger alarm.} \tag{16}$$

When the total RE exceeds a predefined threshold, the system triggers an alarm. The threshold is set with careful consideration of the system's fault tolerance and normal operating range, ensuring that alarms are triggered only when a node's error significantly deviates from normal values and exhibits strong associations with other anomalous nodes. This method effectively prevents false alarms caused by minor errors in local or unrelated nodes, thereby improving the accuracy and precision of the alarm system.

4 Evaluation and Discussion

In this chapter, we compare the proposed method with the existing state-of-the-art methods [4, 29], demonstrating the superiority and effectiveness of the approach presented in this paper.

4.1 Experimental Setup

All experiments were conducted on a server running Ubuntu 18.04 with 64GB of RAM and a 2.20GHz 20-core Intel Xeon CPU. The node feature embedding dimension is 16, the node state dimension is 100, the neighborhood size is 20, and the edge embedding dimension is 200. We recorded the outliers (reconstruction errors) for each event in different systems and labeled the anomalous queues based on the event reconstruction errors (the process is shown in the Fig.2). The threshold is determined by the upper limit when testing for partial benign behavior. Finally, we computed the accuracy, recall, and F1 score, with the formulas as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{17}$$

$$Recall = \frac{TP}{TP + FN} \tag{18}$$

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

$$Precision = \frac{TP}{TP + FP} \tag{19}$$

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{20}$$

4.2 Datasets

We used two publicly available datasets, StreamSpot and Darpa-e3-Theia, as shown in Table 2. The following provides a detailed description of these datasets:

The StreamSpot dataset contains system logs from six different scenarios, with five normal scenarios, including YouTube, email detection, download tasks, CNN, and VGame. The attack scenario involves downloading a program from a malicious URL and exploiting a flash vulnerability to gain system administrator privileges.

The DARPA TC dataset is from the Transparent Computing (TC) project by the Defense Advanced Research Projects Agency (DARPA). In this project, red teams and blue teams conduct offense-defense exercises. During the process, fine-grained system behavior data is collected for attack detection and forensic tracing, which is then used to generate reports.

Table 2. Overview of Datasets

Dataset	*Nodes*	*Edges*	*Attack Edges*	*Proportion*
STREAMSPOT	999,999	89.8 millions	2,842,345	3.165%
E3-THEIA	690,105	32.4 millions	3,119	0.010%

4.3 Visualization

We visualize the extracted embedded features (Fig.3) and reconstruction errors (Fig.4) on the StreamSpot dataset to intuitively demonstrate the effectiveness of our proposed method.

The Fig.3 show the feature representations of attack events after PCA dimensionality reduction, marked by red areas. Our proposed spatiotemporal joint representation method effectively captures the inherent spatiotemporal dependencies in the data. The encoder distinguishes attack events from benign ones with high accuracy, while the decoder produces higher reconstruction errors for anomalous interaction patterns. This approach enhances the system's ability to differentiate between attack and normal behaviors. As illustrated, previously unseen anomalous patterns complicate the original detector's interpretation of attack events, leading to higher reconstruction errors.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

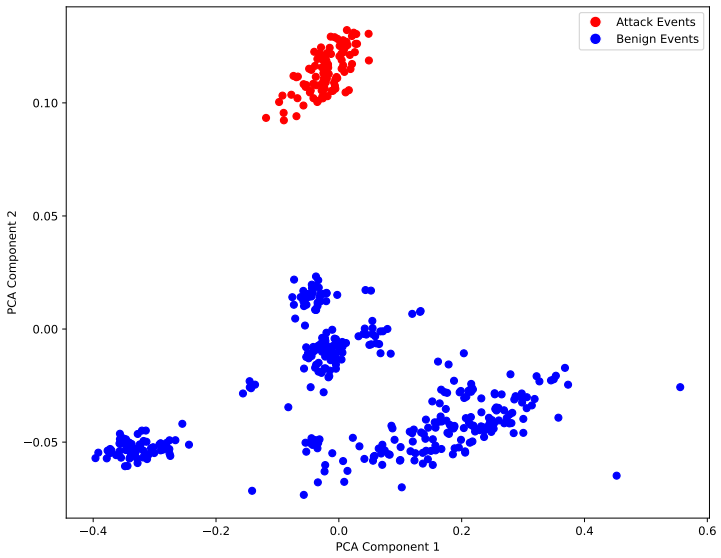

Fig. 3. Host Event Feature Representation After PCA Dimensionality Reduction in StreamSpot Dataset

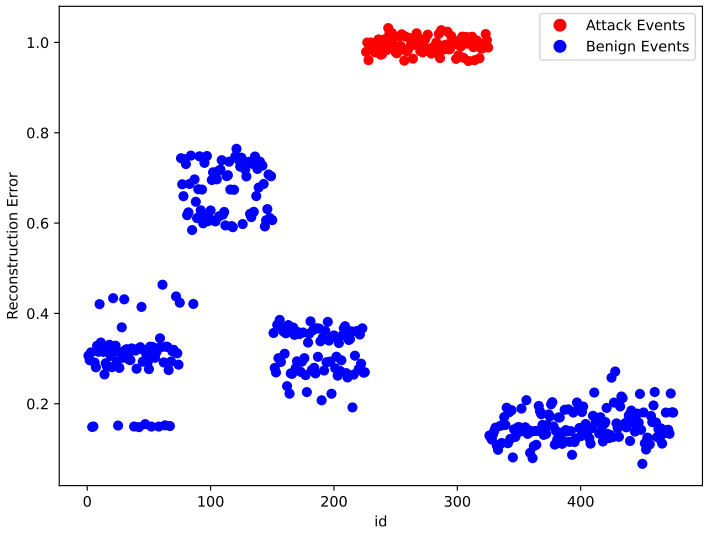

Fig. 4. Reconstruction Error of Host Events in StreamSpot Dataset

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

4.4 Metrics

We selected two HIDS[4, 29] as baseline methods for comparison and conducted ablation experiments, the JST-S system uses only spatial features, the JST-T system uses only temporal features, and the JST system uses the spatiotemporal fusion features proposed in this paper. The experimental results are shown in the Table3 below.

Table 3. Performance Comparison of HIDS

Streamspot Dataset			
System	***Accuracy (%)***	***Recall (%)***	***F1 Score***
Flash	98.0	95.0	97.4
Unicorn	96.2	93.1	95.4
JST-S	100.0	100.0	100.0
JST-T	100.0	100.0	100.0
JST	100.0	100.0	100.0
E3-THEIA Dataset			
System	***Accuracy (%)***	***Recall (%)***	***F1 Score***
Flash	35.9	99.8	23.3
Unicorn	36.4	100.0	24.4
JST-S	95.6	100.0	64.3
JST-T	84.1	88.9	30.8
JST	96.9	100.0	72.0

The proposed method effectively identifies potential entity interaction patterns and significantly outperforms all baseline models across various evaluation metrics, including recall and false positive rates. Specifically, while both baseline models can detect nearly all attack behaviors in the E3-THEIA dataset, they exhibit high false positive rates by misclassifying benign behaviors as attacks. Observations reveal that the baseline models adopt a broader concept of attack nodes, which reflects their reliance on only semantic information and shallow temporal features, resulting in imprecise modeling and numerous false positives. In ablation experiments, although the performance declines when only temporal or semantic features are used, the proposed method consistently outperforms baseline models. This indicates that the relationships among attack nodes helps reduce false positives while maintaining a high recall rate.

We tested different thresholds on the E3-THEIA dataset and observed variations in True Positive Rate (TPR) and True Negative Rate (TNR), which represent the accuracy of detecting attack and benign behaviors, respectively. As shown in the Fig.5, increasing the threshold reduces the false positive rate but results in more missed detections. The optimal threshold for balancing these metrics should be determined based on specific environmental conditions, emphasizing the importance of precise modeling of system behavior.

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

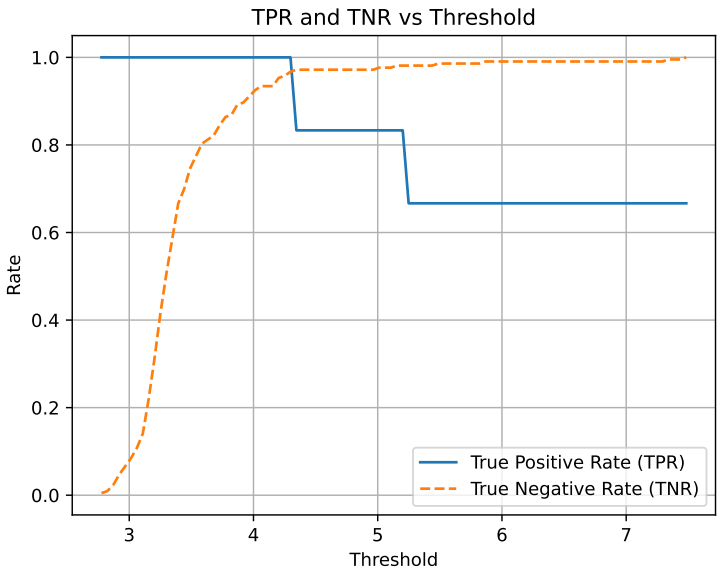

Fig. 5. Relationship between TPR and TNR across Different Thresholds in Theia Dataset

Although the proposed method performs best in terms of false positive rate, false positives remain an unresolved issue. This is particularly true in practical applications, where the large volume of data may prevent timely processing of the generated false positives [34]. The main cause of this phenomenon is the potential misclassification of benign activity patterns as anomalies, which is a common challenge in anomaly detection systems. Leveraging the world knowledge of large language models [35] is a promising avenue for reducing false positives. We plan to explore the integration of large model techniques in our future work to address this issue.

5 Conclusion

To more accurately extract the embedding representations of host behaviors, this paper proposes a joint spatiotemporal graph representation learning method that explicitly models the spatiotemporal dependencies within host behaviors. The method uses a spatiotemporal joint encoder to independently extract features in both spatial and temporal dimensions, while a joint aggregation mechanism effectively captures spatiotemporal dependencies. Based on the relationships between anomalous behaviors and host activities, an alarm queue is constructed, which generates a comprehensive alarm. Experimental results on the Streamspot and DARPA-Theia datasets demonstrate that the proposed method effectively

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

identifies potential entity interaction patterns and significantly outperforms all baseline models across multiple evaluation metrics, such as recall rate and false positive rate. Future work will integrate large language models for alarm analysis to further enhance the automated handling of false alarms, thereby improving the method's applicability in real-world scenarios.

Acknowledgments. We would like to express our sincere gratitude for the financial support provided by Youth Innovation Promotion Association, CAS (No.2023170).

References

1. Li, Z., Chen, Q. A., Yang, R., et al.: Threat detection and investigation with system-level provenance graphs: A survey. *Comput. Secur.* **106**(C), 102282 (2021). https://doi.org/10.1016/j.cose.2021.102282
2. Manzoor, E., Milajerdi, S. M., Akoglu, L.: Fast memory-efficient anomaly detection in streaming heterogeneous graphs. In: KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1035–1044. Association for Computing Machinery, New York, NY, USA (2016)
3. Xie, Y., Feng, D., Hu, 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 (2020)
4. Han, X., Pasquier, T. F. J., Bates, A., et al.: 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)
5. Han, X., Yu, X., Pasquier, T., et al.: SIGL: Securing Software Installations Through Deep Graph Learning. In: Security Symposium (USENIX Sec'21). USENIX (2021)
6. Wang, Q., Hassan, W. U., Li, D., et al.: You are what you do: Hunting stealthy malware via data provenance analysis. In: Proceedings 2020 Network and Distributed System Security Symposium (2020)
7. Li, S., Dong, F., et al.: Nodlink: An online system for fine-grained APT attack detection and investigation. In: Proceedings of the Network and Distributed System Security Symposium (2024)
8. Cheng, Z., Lv, Q., Liang, J., Wang, Y., Sun, D., Pasquier, T., Han, X.: KAIROS: Practical Intrusion Detection and Investigation using Whole-system Provenance. In: 2024 IEEE Symposium on Security and Privacy (2024)
9. Veličkovi'c, P., Cucurull, G., Casanova, A., Romero, A., Li'o, P., Bengio, Y.: Graph Attention Networks. In: International Conference on Learning Representations (2018)
10. Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F., Bronstein, M.: Temporal Graph Networks for Deep Learning on Dynamic Graphs. In: ICML 2020 Workshop on Graph Representation Learning (2020)
11. Han, X., Pasquier, T., Ranjan, T., et al.: Frappuccino: Fault-detection through runtime analysis of provenance. In: HotCloud '17: Proceedings of the 9th USENIX Conference on Hot Topics in Cloud Computing, p. 18. USENIX Association, USA (2017)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

12. Xie, Y., Feng, D., Tan, Z., et al.: Unifying intrusion detection and forensic analysis via provenance awareness. *Future Gener. Comput. Syst.* **61**(C), 26–36 (2016)
13. Xie, Y., Wu, Y., Feng, D., et al.: P-gaussian: Provenance-based Gaussian distribution for detecting intrusion behavior variants using high-efficiency and real-time memory databases. *IEEE Transactions on Dependable and Secure Computing* **18**(6), 2658–2674 (2021)
14. Sun, X., Dai, J., Liu, P., et al.: Using Bayesian networks for probabilistic identification of zero-day attack paths. *IEEE Transactions on Information Forensics and Security* **13**(10), 2506–2521 (2018)
15. Li, Z., Cheng, X., Sun, L., et al.: A hierarchical approach for advanced persistent threat detection with attention-based graph neural networks. *Sec. and Commun. Netw.* (2021)
16. Ayoade, G., Akbar, K. A., Sahoo, P., et al.: Evolving advanced persistent threat detection using provenance graph and metric learning. In: 2020 IEEE Conference on Communications and Network Security (CNS), pp. 1–9. IEEE (2020)
17. Crowdstrike: "Why Dwell Time Continues to Plague Organizations." (2019) https://www.crowdstrike.com/blog/why-dwell-time-continues-to-plague-organizations/
18. Gartner Peer Insights: "Endpoint Detection and Response Solutions Market." (2019) https://www.gartner.com/reviews/market/endpoint-detection-and-response-solutions
19. Hiroki, T., Yoshiaki, S., Koji, K., and Takayoshi, A.: "Automated Security Intelligence (ASI) with Auto Detection of Unknown Cyber-Attacks," *NEC Technical Journal*, vol. 11 (2016)
20. Fireeye: "Incident Investigation." (2019). https://www.fireeye.com/solutions
21. swimlane: "Automated Incident Response: Respond to Every Alert." (2019). https://swimlane.com/blog/automated-incident-response-respond-every-alert/
22. Malwarebytes Inc.: "Malwarebytes." (2022). https://www.malwarebytes.com/
23. Splunk Inc.: "splunk." (2018). https://www.splunk.com
24. S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," Journal of Computational Science, vol. 25, (2018)
25. 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)
26. Z. K. Maseer, R. Yusof, N. Bahaman, S. A. Mostafa, and C. F. M. Foozy, "Benchmarking of machine learning for anomaly based intrusion detection systems in the cicids2017 dataset," IEEE access, vol. 9, (2021)
27. M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: Anomaly detection and diagnosis from system logs through deep learning," inACM Conference on Computer and Communications Security (CCS), (2017)
28. Y. Shen and G. Stringhini, "Attack2vec: Leveraging temporal word embeddings to understand the evolution of cyberattacks," in USENIX Security Symposium, (2019)
29. M. Ur Rehman, H. Ahmadi and W. Ul Hassan, "Flash: A Comprehensive Approach to Intrusion Detection via Provenance Graph Representation Learning," 2024 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2024, pp. 3552-3570, (2024) doi: 10.1109/SP54263.2024.00139.
30. M. Hucka, "Nostril: A nonsense string evaluator written in Python,"Journal of Open Source Software, vol. 3, no. 25, p. 596, (2018)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18

31. P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching Word Vectors with Subword Information," Transactions of the Association for Computational Linguistics, vol. 5, pp. 135–146, (2017)
32. Cho, K., Van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y.: Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: *Computer Science* (2014)
33. Li, G., Xiong, C., Thabet, A., and Ghanem, B.: DeeperGCN: All you need to train deeper GCNs. In: *ICLR 2022 Conference Withdrawn Submission* (2022)
34. Michael Zipperle, Florian Gottwalt, Elizabeth Chang, and Tharam Dillon.: Provenance-based Intrusion Detection Systems: A Survey. ACM Comput. Surv. 55, 7, Article 135 (2022) https://doi.org/10.1145/3539605
35. Yang, Y., Tang, J., Xia, L., Zou, X., Liang, Y., and Huang, C., "GraphAgent: Agentic Graph Language Assistant". arXiv preprint arXiv:2412.17029. (2024)

ICCS Camera Ready Version 2025
To cite this paper please use the final published version:
DOI: 10.1007/978-3-031-97632-2_18