2024 IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) | 979-8-3315-0620-9/24/$31.00 ©2024 IEEE | DOI: 10.1109/TrustCom63139.2024.00098

RAG-based Cyber Threat Tracing Graph Modeling Method

Jong-Hee Jeon
Security Engineering Lab., Dept. of Computer and Information Security,
Sejong University
Seoul, Republic of Korea
jonghee.jeon530@gmail.com

Jahoon Koo
Security Engineering Lab., Dept. of Computer and Information Security,
Sejong University
Seoul, Republic of Korea
sigmao@sejong.ac.kr

Young-Gab Kim[†]
Security Engineering Lab., Dept. of Computer and Information Security, and Convergence Engineering for Intelligent Drone
Sejong University
Seoul, Republic of Korea
alwaysgabi@sejong.ac.kr

Abstract— **Cyber-attacks have become increasingly complex and sophisticated, occurring frequently through various paths. Numerous attack techniques, such as ransomware, advanced persistent threat (APT) attacks, and viruses facilitated by generative artificial intelligence (GAI), are threatening systems. Attackers often remain hidden for extended periods, using networks to infiltrate and spread within internal systems, aiming to steal sensitive information. These stealthy lateral movements are difficult to detect and defend against, necessitating a graph-based approach to effectively track and analyze threats. In this paper, we propose a retrieval-augmented generation (RAG) based threat behavior tracing method that derives more meaningful results through the information retrieval functions in the large language model (LLM). We present a suitable graph model for threat tracing by comparing label property graph (LPG) and resource description framework (RDF) as graph methods for knowledge extraction in RAG.**

Keywords— retrieval-augmented generation (RAG), cyber threat tracing, graph modeling, resource description framework (RDF), label property graph (LPG)

I. INTRODUCTION

The modern cyber environment is more complex than ever and is constantly threatened by various attack techniques. Recently, the development of generative artificial intelligence (GAI) has further diversified cyber-attack techniques, leading to the evolution of methods for virus, ransomware, and advanced persistent threat (APT) attacks. These attacks are becoming increasingly secretive and can remain latent for a long period, leading to the continued spread of internal networks. MITRE [1] provides a knowledge-based ATT&CK framework for design attack tactics and techniques to build awareness and understanding of how complex and diverse cyberattacks operate. It defines the act of a malicious actor intruding into a network system to achieve a goal and gain access to the next target on the network through lateral movement. Internal lateral movement is challenging to detect and respond to until the attack surfaces, making it difficult to block or defend with existing security systems alone completely. In addition, since existing lateral movement detection focuses on protecting a single endpoint, it is difficult to detect or identify the provenance of threats moving laterally through multiple endpoints.

A delayed response to threats can exponentially increase the damage caused by cyber threats, making the need for rapid detection and response methods more critical than ever. As cyber threats become more sophisticated and widespread, the ability to quickly track and respond to threat behaviors is essential for minimizing potential damage. Without timely intervention, the impact of these threats can escalate, leading to severe consequences for organizations and individuals alike. As a result, graph-based approaches have been studied to effectively track and analyze such behaviors [2, 3]. For example, HOLMES [4] conducted a study aimed at generating detection signals representing APT campaign information by proposing a graph-based detection system that generates high-level graphs that summarize attacker behaviors in real time. Similarly, Latte [5] conducted a study on discovering potential malicious lateral movement paths by proposing a graph-based detection system designed to address the problems caused by large-scale data and lack of attacker knowledge. In addition, with the recent advancement of GAI, retrieval-augmented generation (RAG) technology is utilized to prevent answer errors, and graphs serve as knowledge repositories. Representative data structures for constructing graphs include resource description framework (RDF) and labeled property graph (LPG). Graph modeling is important for graph-based cyber threat tracing, but there is currently no standardized or widely accepted method for graph modeling.

In this paper, we propose an efficient graph modeling method through a comparison of graph models for RAG-based threat behavior tracing. Our approach involves constructing and comparing LPG and RDF graph models to identify the most effective structure and query performance for cyber threat tracing. The graph models are constructed on a graph database and integrated with RAG, we aim to enhance detection and response capabilities against sophisticated threats. We also focus on optimizing how LLMs understand and query these graph structures to improve the overall effectiveness of the tracing system. Ultimately, our goal is to provide a practical, scalable solution that advances cyber threat analysis capabilities. This paper is organized as follows. Section II presents the background and related work. Section III describes the RAG structure and graph modeling. Section IV evaluated the experiments and Section V concludes this paper.

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024-00335672). [†] Corresponding author: Young-Gab Kim.

II. BACKGROUND AND RELATED WORK

This section presents the background and research on graph-based cyber threat tracing, RAG-based cyber security, and lateral movement source detection. We also present the limitations of these studies and describe the problem definition.

A. Graph-based Cyber Threat Tracing

Many studies have recently recognized the importance of graph-based cyber threat tracing. Representative models for this type of research include LPG and RDF. LPG is a graph data model that incorporates additional properties into nodes and edges. Each node can have a label and properties, with the label defining the domain or group information about the type of data the node represents. RDF, by the W3C, is a data model used to describe resources. It expresses the relationships between data in the form of subject-predicate-object triples within the semantic web and is identified through URIs, structuring data with a schema. MITRE proposed CyGraph [6], an LPG-based framework for ATT&CK, which is a framework for attacks and tactics. CyGraph adopts an integrated graph cyber security model related to cyber-attacks and identifies relationships in complex cyber security domains by associating various vulnerabilities and security events with vulnerability paths within the network environment. Chetwyn et al. [7] simulated threat behaviors using MITRE's Caldera, and describing the relationships between security events based on graphs. They proposed supporting threat hunting through connections with threat behaviors and tracing and analyzing endpoint detection and response threat behaviors using provenance graphs that identify data generation sources. KRYSTAL [8] demonstrates how attack graphs and scenario reconstruction can achieve threat detection, tracing, and data expansion. It also proposes an RDF-based provision graph framework for this purpose.

B. RAG-based Cybersecurity

Along with the development of LLM, research to utilize them is actively underway in the cybersecurity field. Fine-tuning is a representative method for optimizing the accuracy of LLM. LogPrompt [9] uses prompt technology and a pre-trained language model to enable good detection with a small amount of training data, thereby overcoming the limitations of existing deep learning-based log anomaly detection techniques (e.g., class imbalance, data training cost). However, it is a slow and costly method due to the significant computing resources required. Data quality and performance issues can also cause hallucinations and inaccurate responses. To address these challenges, RAG technology has been introduced. By linking the search mechanism with a large language model, more accurate answers that match the question and situation are derived. RAGLog [10] proposed an approach that integrated the RAG model into abnormal behavior detection analysis of system logs. Using the RAG framework, detection capabilities were improved by searching for information related to abnormal behavior through massive log data using a vector database. It was confirmed that significant improvement in extracting relevant context occurred when RAG was used for large language models. Singh et al. [11] shows how RAG can be leveraged to gain a deeper understanding and strengthen a system's overall cybersecurity measures through data from various sources, such as system logs and threat reports.

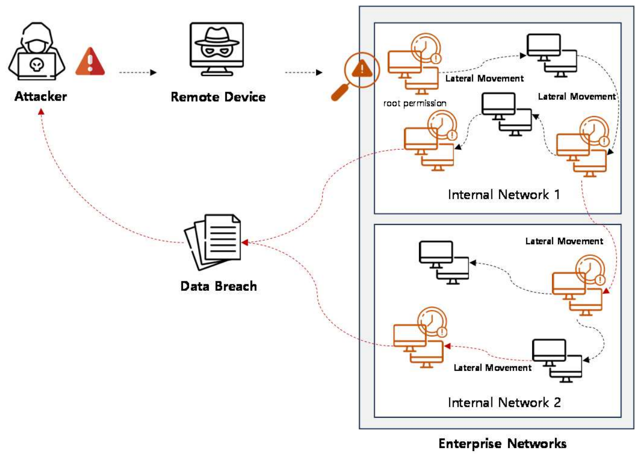


Fig. 1. Lateral movement attack

C. Lateral Movement Provenance Detection

In the field of cybersecurity, APT attacks are those that take control of a system and move laterally through an internal network, remaining hidden for an extended period of time to steal the desired information. As shown in Fig. 1, the attacker gains access to the internal network through a vulnerability, moves laterally to maintain control of the network over a long period, and ultimately obtains and exfiltrates the targeted data. This lateral movement is also defined as a core tactic by the MITRE framework. LMTracker [12] proposed a heterogeneous graph-based attack path detection method. This method organizes event logs into heterogeneous graphs, generates vectors for lateral movement paths, and detects threat lateral movement paths using an unsupervised algorithm. Rabbani et al. [13] proposed a method to detect lateral movement paths by organizing event logs into a graph and using DeepWalk, a graph algorithm, for node embedding. They studied a technique that can effectively and dynamically analyze and detect event logs within a connected structure. Smiliotopoulos et al. [14, 15] proposed a tool for generating and evaluating data by utilizing the MITRE ATT&CK framework and implementing nine lateral movement methods, along with a methodology for building machine learning models for lateral movement detection. They also described the processes for feature selection, data preprocessing, and assessing feature importance in lateral movement detection.

D. Problem Definition

Despite advancements in graph-based cyber threat tracing, RAG-based cybersecurity, and lateral movement detection, several limitations persist in existing research. Graph models like LPG and RDF, while effective in structuring data, often struggle with efficient querying and integration with LLMs, leading to delays in real-time threat detection in complex network environments. RAG-based approaches, although promising for improving LLM accuracy, are hindered by high computational costs and issues such as hallucinations and inaccuracies due to data quality challenges. Additionally, current methods for detecting lateral movement paths in networks face scalability and adaptability issues, making them less effective in real-time scenarios, particularly against APT. To address these challenges, we propose a graph modeling

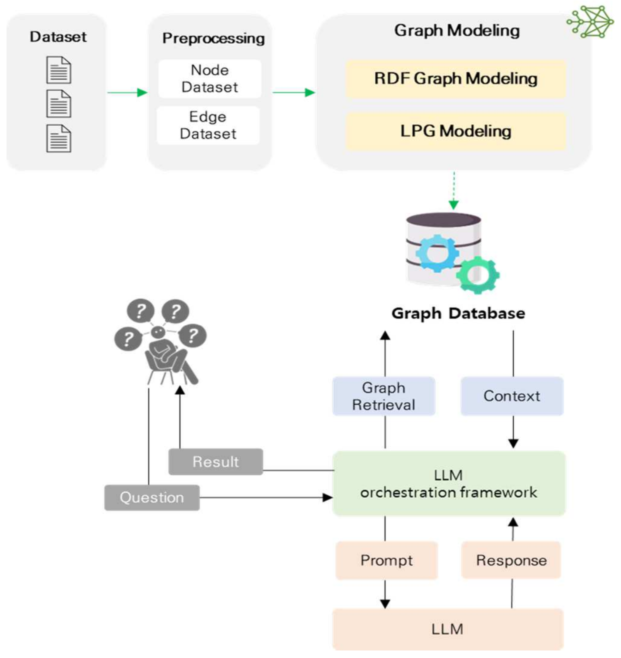


Fig. 2. RAG-based graph modeling framework for cyber threat tracing

method that enhances RAG-based cyber threat tracing by optimizing graph structures and comparing query performance. This approach provides a more scalable and practical solution for advanced cyber threat analysis, which is detailed in Section III.

III. PROPOSED APPROACH

In this section, we outline the core components and prompt design of our RAG-based graph modeling approach for cyber threat tracing. We focus on the integration of LPG and RDF technologies with an LLM orchestration framework to enhance threat detection and analysis. Additionally, we propose an application method that leverages these components for more effective cyber threat tracing.

A. Components of RAG-based Cyber Threat Tracing Framework

The overall structure of the graph modeling method for RAG-based cyber threat tracing is shown in Fig. 2. This structure primarily consists of graph modeling, a graph database, an LLM orchestration, LLM, and a prompt. LLMs include OpenAI's ChatGPT, Google's Gemini, and Mistral. The LLM orchestration is designed to simplify application development by integrating large language models. The components of this framework are designed to structure data through graph modeling for effective threat tracing.

B. Graph Modeling

LPG and RDF are modeled to effectively represent and analyze the complex relationships within cyber threat environments, allowing for detailed exploration and understanding of data structures. LPG modeling includes node, edge, and property components. Additionally, labels that classify Nodes by type are included in the modeling elements. LPG graphs are modeled as follows.

- Node: Nodes consist of processes, network endpoints, and file descriptors. The main identifiers of processes are process name and process id. The main identifiers of network endpoints are client and server IP and port, respectively. The file descriptor is defined by the corresponding process file name.
- Edge: Edges consist of inter-process relationships (i.e., parent-child), process-to-network relationships, process-to-file relationships, and inter-network relationships (i.e., server-client).
- Property: Each node and edge has properties such as DateTime, process arguments, ports, and file type.

RDF Graph is a data structure consisting of a subject, predicate, and object. Each triple describes a specific node and its properties and relationships. The subject, predicate, and object parts of each triple are as follows.

- Subject: This is the starting point of the triple, usually an event node. In the example, Event Node is the subject.
- Predicate: The predicate represents the relationship between the subject and the object. In the example, HAS_TYPE, HAS_PROC_NAME, HAS_PROC_PNAME, HAS_FD_TYPE, HAS_FILE_DESCRIPTOR, HAS_PROPERTY, HAS_CLIENT_IP, HAS_SERVER_IP, HAS_CLIENT_PORT, HAS_SERVER_PORT are predicates.
- Object: An object is a node that is connected to a subject through a predicate. In the example, eventType, procName, procPname, fdType, fileDescriptor, Property, clientIP, serverIP, clientPort, serverPort are objects.

Therefore, LPG is a data structure that adds properties to each node and edge, allowing for efficient representation of intuitive modeling and complex processes. RDF, on the other hand, is structured as subject-predicate-object triples, making it suitable for ontology-based reasoning and demonstrating strengths in semantic expression and interoperability. These characteristics result in distinct approaches to threat detection and tracking between the two graph models.

C. LLM Ochestration

To effectively build and deploy a cyber threat tracing system that leverages LLM, a robust orchestration is essential. An LLM orchestration manages the workflow of LLM tasks and provides scalable solutions tailored to various applications. This framework supports LLM in efficiently retrieving and analyzing cyber threat information. In this study, we adopt LangChain [16] as the LLM orchestration. LangChain offers agent functionalities that support the dynamic selection of LLM and facilitate user interaction. Through LangChain, LLM can process relevant data according to specific cyber threat scenarios, enabling them to produce optimal results. LangChain is utilized to perform the following roles in the proposed framework.

- Modular Architecture: Through LangChain's modular structure, our provides the ability for each component to operate independently while still being seamlessly integrated. This capability enables flexible configuration of the LLM-based cyber threat tracing system, allowing for the addition of new modules or adjustments to existing ones as needed, enhancing adaptability and responsiveness to evolving threats.
- Prompting and Context Management: Leveraging LangChain's prompting and context management features, our framework ensures that the language model generates accurate, question-specific answers while maintaining a natural and continuous interaction flow. This functionality significantly improves the user experience and the precision of threat analysis in complex cyber-attack tracing scenarios.
- Scalability and Integration: Leveraging LangChain's flexible architecture, our framework seamlessly integrates with applications and systems. As the system scales, it facilitates the incorporation of additional LLM models or data sources without disrupting ongoing operations. This scalability ensures that the LLM-based cyber threat tracing solution can adapt to increasing data volumes and evolving cybersecurity challenges, providing continuous improvement and alignment with the latest security strategies.

Through these features, LangChain serves as a powerful tool for cyber threat detection and response. It facilitates the efficient management of LLM models and supports the derivation of optimal results in complex data environments. The characteristics of this LLM orchestration play a critical role in maximizing the potential of LLM within cybersecurity contexts.

D. *Cyber Threat Tracing Method*

Cyber threat tracing utilizes data from a graph database constructed through graph modeling (i.e., LPG, RDF). When a user poses a question to the RAG-based LLM, the system extracts the necessary data using a graph database query language, based on the analyzed question information and the defined prompt, and provides an answer to the user. To effectively utilize LLM for cyber threat tracing, prompt engineering plays a crucial role. Prompt engineering involves designing prompts that enable LLM to generate accurate and meaningful responses based on the provided data. This process is essential for addressing complex questions related to cyber threats effectively. We utilize few-shot prompting and zero-shot prompting as key techniques in our prompt engineering.

- Few-Shot Prompting: Our framework can provide the LLM with a few examples through few-shot prompting, enabling it to effectively solve similar problems. In the context of cyber threat tracing, this allows the LLM to identify and analyze new threats with similar patterns by supplying representative attack scenarios.
- Zero-Shot Prompting: By leveraging this approach, our framework makes it possible for the LLM to solve problems without prior examples. It enables the LLM to respond effectively to novel threats, using its existing knowledge and data to address emerging challenges.

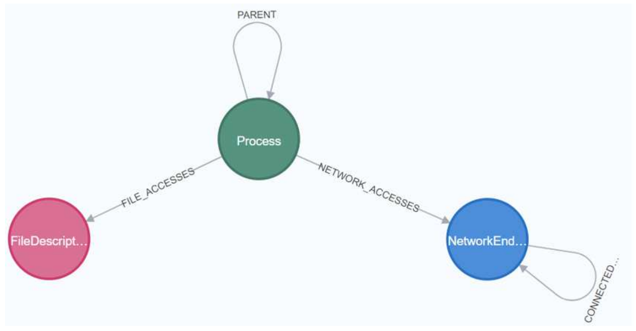


Fig. 3. LPG graph model schema

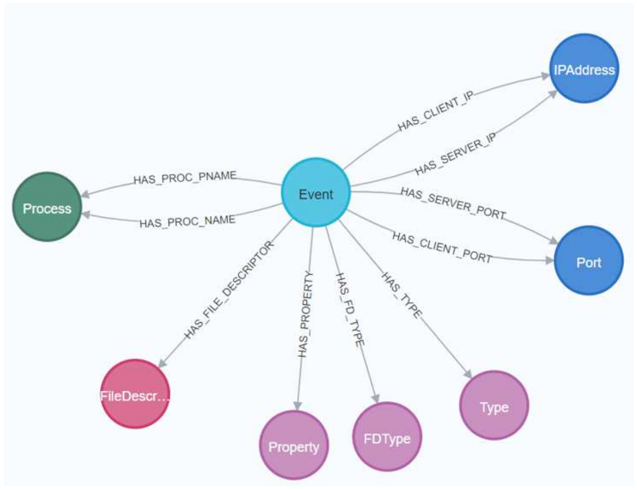


Fig. 4. RDF graph model schema

Graph modeling is used as a key component of RAG-based cyber threat tracing. LLMs use data from these graph models to generate answers to questions related to cyber threats. Although LLMs can understand basic query syntax, constructing a query structure for exploring graph database schemas can be challenging. Therefore, it is essential to design prompts that include graph exploration scenarios to facilitate effective graph database exploration. It is crucial to design effective prompt queries so that LLMs can understand the database structure and contents, enabling them to quickly extract the necessary information. RAG-based cyber threat tracing structures cyber threat data through graph modeling and graph databases, analyzing the relationships between various entities in the network. The LLM orchestration coordinates the work of LLM to enable efficient data processing and user interaction. Prompt engineering instructs the LLM to accurately interpret cyber threat data and generate reliable answers to user queries.

IV. Experimental Result and Evaluation

This section presents a comprehensive evaluation of RAG-based threat-tracing graph modeling. It describes experimental environment and evaluates key metrics, including graph size (i.e., nodes, edges), physical capacity, and query speed.

A. *Dataset*

The dataset for the provenance graph modeling for RAG-based cyber threat tracing uses the ProvSec [17] dataset. The

ProvSec dataset is a cybersecurity provenance dataset for event tracing generated through a Docker container and sysdig event tracer on Ubuntu 20.04. The entire dataset generated through 11 attack scenarios contains approximately 1.3 million events. The dataset includes fields such as system call, process name, parent process name, process ID, parent process ID, server IP, and client IP. It is organized into 22 files and divided into 11 normal and abnormal cases.

The ProvSec dataset, designed to mimic real-world environments for cybersecurity provenance, is formatted in JSON and enables complex analysis across various real-world attack scenarios.

B. Graph Modeling

Generated LPG Graph Model Schema is shown as Fig. 3. The green node is a process and has a self-relationship because it has a connection relationship with the parent process. The red node is a file, and the blue node is a network event. Since the process is the subject, they have an outgoing connection relationship from the process. The network node shows a self-relationship because actions occur with other hosts. Fig. 4 is the RDF Graph Model Schema. The green node is a process, the red file node, the blue IP and Port node, and the purple are attribute information that occurs in one event, centered on the action called an event.

C. RAG

To configure the RAG environment, we utilized the LangChain framework in Python as the main framework and GPT-4o as the LLM. LangChain is a sub-framework that facilitates easy integration with various AI models and plays an important role in the RAG-based system. The database for RAG includes Neo4j, which is used for graph construction and provides robust functions for managing graph data. For the graph query language, we used Cypher, which is compatible with Neo4j, to enable efficient querying and data exploration. This environment configuration focuses on analyzing and tracing cyber threats by leveraging RAG technology.

D. Evaluation Metrics

The following key metrics were used to evaluate the constructed LPG and RDF.

- Graph Schema Size: Compare the number of nodes, properties, and edges of the LPG and RDF graphs.
- Graph Query Speed: Evaluate the results derived for the same information queried in the LPG and RDF graphs.
- Prompt Response: Evaluate the results for the same prompts in the constructed RAG and RDF graphs.

E. Evaluation Results

We converted the ProvSec dataset to fit the LPG and RDF graph models, loaded them into the Neo4j graph database. Next, we compared the graph schema sizes of the LPG and RDF models, the query performance of the cyber threat tracing graph, and the results of user questions prompted in RAG. Table I compares the graph schema size of LPG and RDF generated in the graph database.

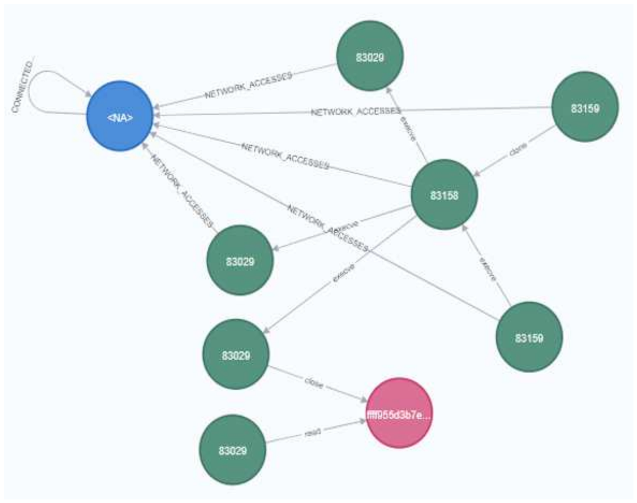
TABLE I. LPG AND RDF GRAPH SCHEMA SIZE COMPARISON

Type	Graph Schema Size		
	Node	**Edge**	**Property**
LPG	1,310,134	16,003,852	9,122,431
RDF	28,784,340	33,199,265	14,392,170

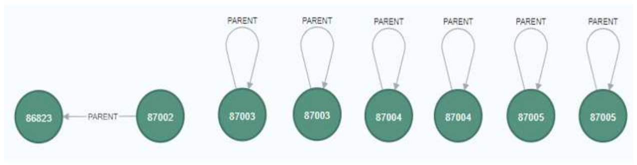
We compared the performance of each graph using three queries on LPG and RDF. Queries 1 and 2 retrieve vulnerability information from the ProvSec dataset, while Query 3 performs a network connectivity search. Query 1 identifies a path traversal and file disclosure vulnerability in Apache HTTP Server, which allows the root of an expected document to be mapped to an external file via a path traversal attack in Apache 2.4.49 (i.e., CVE-2021-41773).

- Query 2 identifies a vulnerability in Python PIL/Pillow for remote shell command execution via Ghostscript, specifically in versions prior to 9.24 (i.e., CVE-2018-16509).
- Query 3 compares the connectivity of network events and assesses the performance of deriving connected relationships from 1 to 4 hops.

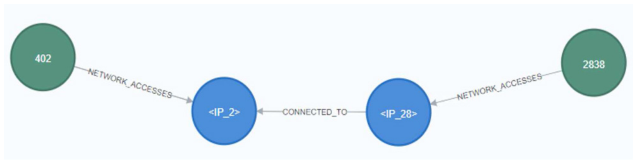
Figs. 5 and 6 present the graphs of LPG and RDF corresponding to Queries 1, 2, and 3. These figures include: (1) a simplified backtrack graph for Query 1, which encompasses Apache HTTP Server path traversal and file disclosure vulnerabilities included in the ProvSec dataset. (2) a vulnerability graph for Python PIL/Pillow, illustrating remote shell command execution via Ghostscript corresponding to Query 2. (3) a network event relationship graph corresponding to Query 3. In Fig. 5 (a), the LPG graph corresponding to Query 1 intuitively illustrates the path of the top-level process 83029, which generates network events and file accesses, along with processes 83158 and 83159. The LPG graph highlights its effectiveness in visually representing the flow and interactions within the event, making it easier to trace the provenance and propagation of events in cyber threat tracing. In contrast, Fig. 6 (a) displays the RDF graph, which shows the network and file events related to process 83029 that occur in the remaining processes and the httpd process. These relationships are expressed in a more complex and difficult-to-interpret manner than the LPG, which can hinder the rapid identification of threat paths and complicate analysis. The RDF model's complexity in visual representation, while powerful for detailed semantic relationships, may pose challenges in scenarios requiring quick decision-making and straightforward visualization. When examining Fig. 5 (b) and Fig. 6 (b) graphs corresponding to Query 2, each process is executed sequentially, starting a shell and creating a demo file. The parent-child relationship is not represented in the data, resulting in a disjointed appearance. For Query 3, the LPG graph in Fig. 5 (c) and 5 (d) shows an intuitive network connection relationship, flowing from the process to the network and back to the process. This clear depiction of network interactions underscores the LPG model's advantage in scenarios where understanding the flow of information is crucial for effective threat detection and response.



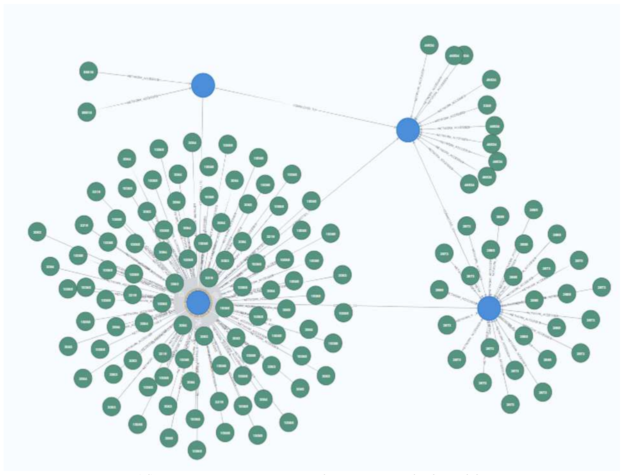
(a) LPG Graph Path traversal and file disclosure vulnerability in Apache HTTP Server



(b) LPG Graph Python PIL/Pillow Remote Shell Command Execution via Ghostscript

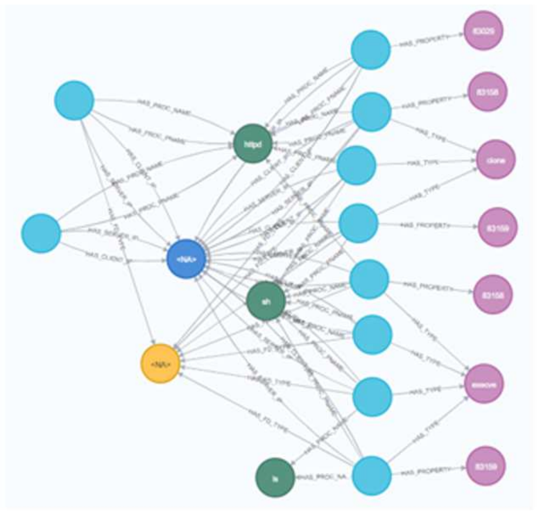


(c) LPG Network Event Relationship

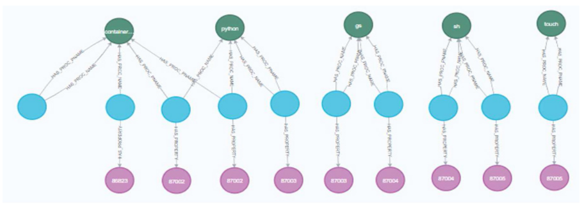


(d) LPG 3-Hop Network Event Relationship

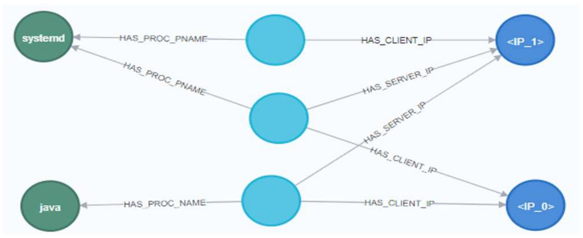
Fig. 5. LPG graph query visualization



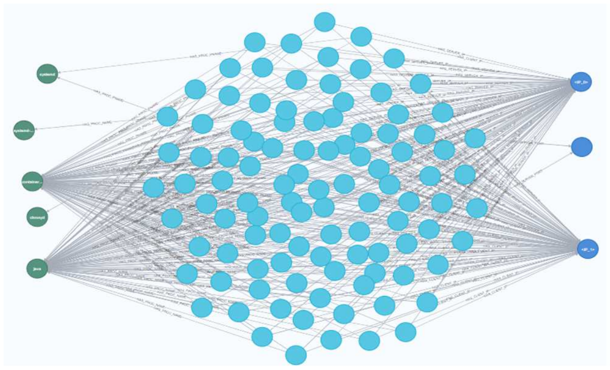
(a) RDF Graph Path traversal and file disclosure vulnerability in Apache HTTP Server



(b) RDF Graph Python PIL/Pillow Remote Shell Command Execution via Ghostscript



(c) RDF Network Event Relationship



(d) RDF 3-Hop Network Event Relationship

Fig. 6. RDF graph query visualization

The RDF graph in Fig. 6 (c) and 6 (d) presents an event connection relationship that is not intuitive, making the relationship difficult to understand and more complex in structure. The complexity of the RDF graph illustrates the potential drawbacks of using RDF for certain types of cyber threat analysis, where the added complexity may obscure critical connections and delay the identification of threats. Fig. 5 and 6 demonstrate that while both LPG and RDF have their strengths, LPG's more straightforward and intuitive visualizations can be particularly advantageous in situations requiring quick, clear interpretations of complex cyber threat tracing. This is because LPG effectively utilizes the properties of nodes and edges to enable direct and rapid data access. In contrast, RDF, with its detailed and intricate relationships, may be better suited for in-depth analyses where the semantic richness of data is crucial. However, in the case of multi-hop exploration, performance degradation occurs, making it less suitable for tracing complex network propagation paths. Graph Query Performance: A comparison of three queries is provided below and in Table II.

- Query 1 Performance Comparison Results: In a query on Apache HTTP Server path traversal and file exposure vulnerabilities, LPG showed a response time approximately 5 times faster than RDF. LPG demonstrated high query efficiency due to its structure, which allows for direct connection and exploration of relationships.
- Query 2 Performance Comparison Results: In a query on Python PIL/Pillow remote shell command execution vulnerabilities, RDF returned results faster than LPG. RDF's clear schema and ontology enable it to efficiently process complex property queries.
- Query 3 Performance Comparison Results: In a query comparing the connectivity of network events LPG performed well when exploring multiple hops, but RDF's performance deteriorated sharply as the number of hops increased, and it failed to return results in multi-hop exploration.

The basic query for exploring relationships between networks of LPG and RDF is executed as follows:

LPG:

```
MATCH path = (p:Process)-[]-(n:NetworkEndpoint)-[]-(nn:NetworkEndpoint)-[]-(pp:Process)
WHERE n.ServerIP <> '<NA>' and nn.ClientIP <> '<NA>' and p.ProcPID <> pp.ProcPID
RETURN path LIMIT 1
```

RDF:

```
MATCH path = (p:Process)-[]-()-[]-(i:IPAddress)-[]-()-[]-(ii:IPAddress)-[]-()-[]-
(:Process) WHERE i.ip <> '<NA>' and ii.ip <> '<NA>' and i.i p <> ii.ip  RETURN path LIMIT 1
```

For multi-hop search, the number of edges is increased and searched.

TABLE II. GRAPH QUERY PERFORMANCE BETWEEN LPG AND RDF MODELS

Graph Query Response Time (*ms*)		
Graph Query	**LPG**	**RDF**
Query1	3,367	15,316
Query2	1,428	656
Query3	13,238	4-Hop / No Results

TABLE III. PROMPT QUERY RESULTS LPG AND RDF MODELS

Prompt Results				
Query	**zero-shot prompting**		**few-shot prompting**	
	LPG	**RDF**	**LPG**	**RDF**
1	✓	✓	✓	✓
2	✓	✗	✓	✓
3	✗	✗	✗	✗

We compared RAG prompt results based on LPG and RDF models. In addition, we performed zero-shot prompting and few-shot prompting queries on the LPG and RDF models. The queries were: (1) "Find events related to httpd and return 10 results for process name and type," (2) "Return 10 results for network event behaviors that involve 2-3 hops or more," and (3) "Return 5 results for events that could be considered as httpd or ghostscript vulnerabilities." Table III shows the comparison results between LPG and RDF for prompt-based queries. First, zero-shot prompting:

- Query 1: Both LPG and RDF returned correct results without example queries.
- Query 2: LPG returned results from 2 hops, while RDF returned no results. Neither LPG nor RDF returned results from 3 hops or more.
- Query 3: LPG returned some httpd-related results, but no vulnerabilities, and RDF returned no results.

The prompt evaluation of LPG and RDF through few-shot prompting is as follows:

- Query 1: Returned results accurately for both LPG and RDF.
- Query 2: Neither LPG nor RDF returned results well beyond 3 hops.
- Query 3: The query guide was used to find vulnerabilities, but while LPG attempted to find similar results, RDF did not return any results.

We evaluated RAG-based cyber threat tracing using prompted questions, which enabled even non-expert threat analysts to effectively engage in the process by asking relevant questions. Through this approach, we demonstrated the potential of tracing threat behaviors using prompted queries, specifically for vulnerability detection and analysis within RAG-based systems. In this study, we encountered challenges in consistently achieving accurate tracing and detection outcomes when querying for vulnerability detection, particularly in complex scenarios. Future work will focus on overcoming these limitations by exploring and extending advanced graph modeling techniques and refining prompt engineering strategies, including the incorporation of a comprehensive vulnerability knowledge base and improved query structures.

Limitations

Although this study utilizes the ProvSec dataset, which emulates a real cybersecurity environment, it may not fully capture the diversity of real-world cyberattacks. Additionally, the generalizability of the findings may be limited by the restricted number of tested queries. Future studies should enhance the reliability of the results by incorporating diverse datasets and a broader range of queries.

V. Conclusion

This study compared and evaluated graph modeling methods for RAG-based cyber threat tracing. The graph sizes varied across modeling methods, with LPG proving to be more efficient. As the depth of the graph connections increased, query performance also favored LPG. In terms of RAG-based prompt performance, LPG performed well in both zero-shot prompting and few-shot prompting for Query 1 and Query 2, while RDF required Few-Shot Prompting to answer Query 2. However, neither LPG nor RDF provided satisfactory answers for Query 3 (i.e., threat tracing) when prompted. LPG may be more suitable when flexibility and query performance are prioritized in RAG-based cyber threat tracing, whereas RDF may be more appropriate when standardization and interoperability are important. Our study not only contributes to the understanding of RAG-based graph modeling methods in cybersecurity but also lays the foundation for future research on cyber threat tracing. These findings highlight the need for continued exploration of hybrid approaches that could leverage the strengths of both LPG and RDF models. Additionally, further research should focus on enhancing graph query mechanisms and improving the accuracy of RAG-based prompts for complex threat scenarios.

References

[1] MITRE, https://attack.mitre.org/tactics/TA0008/

[2] S. Wang et al., "THREATRACE: Detecting and Tracing Host-Based Threats in Node Level Through Provenance Graph Learning," in IEEE Transactions on Information Forensics and Security, vol. 17,2022, pp. 3972-3987

[3] W. U. Hassan, A. Bates and D. Marino, "Tactical Provenance Analysis for Endpoint Detection and Response Systems," 2020 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2020, pp. 1172-1189

[4] Milajerdi, S.M., Gjomemo, R., Eshete, B., Sekar, R., Venkatakrishnan, V.N., et al., "HOLMES: real-time APT detection through correlation of suspicious information flows," IEEE Symposium on Security and Privacy (SP), 2019, pp. 1137–1152

[5] Q. Liu, J. W. Stokes, R. Mead, T. Burrell, I. Hellen, J. Lambert, A.Marochko, W. Cui, "Latte : Large-Scale Lateral Movement Detection," IEEE Military Communications Conference (MILCOM), 2018, pp. 1–6

[6] S. Noel, E. Harley, K.H. Tam, M. Limiero and M. Share, "CyGraph: Graph-Based Analytics and Visualization for Cybersecurity," Handbook of Statistics, 2016, Volume 35, pp. 117–167

[7] R. Chetwyn, M. Eian, A. Jøsang, "Modelling Indicators of Behaviour for Cyber Threat Hunting via Sysmon," In European Interdisciplinary Cybersecurity Conference, 2024, pp. 104

[8] K. Kurniawana, A. Ekelhart, E. Kieslinga, G. Quirchmayr, A. M. Tjoa, "KRYSTAL: Knowledge graph-based framework for tactical attack discovery in audit data," Computers & Security 121, 2022, Volume 121, Issue C , pp. 102828

[9] T. Zhang, X. Huang, W. Zhao, S. Bian and P. Du, "LogPrompt: A Log-based Anomaly Detection Framework Using Prompts," 2023 International Joint Conference on Neural Networks (IJCNN), Gold Coast, Australia, 2023, pp. 1–8

[10] J. Pan, W. S. Liang and Y. Yidi, "RAGLog: Log Anomaly Detection using Retrieval Augmented Generation," 2024 IEEE World Forum on Public Safety Technology (WFPST), Herndon, VA, USA, 2024, pp. 169–174

[11] J. P. Singh, S. Agrawal, "Automating Threat Intelligence Analysis with Retrieval Augmented Generation RAG for Enhanced Cybersecurity Posture," International Journal of Science and Research (IJSR), 2024, Volume 13 Issue 5, pp. 251–255

[12] Y. Fang, C. Wang, Z. Fang and C. Huang, "LMTracker: Lateral movement path detection based on heterogeneous graph embedding," Neurocompoing 474, 2022, p. 37–47

[13] M. Rabbani, L. Rashidi, Ali A. Ghorbani, "A Graph Learning Based Approach for Lateral Movement Detection," IEEE Transactions on Network and Service Management, 2024, pp. 1–1

[14] C. Smiliotopoulos, K. Barmpatsalou and G. Kambourakis, "Revisiting the Detection of Lateral Movement through Sysmon," Applied Sciences, 2022, Volume 12 Issue 15, pp. 7746

[15] C. Smiliotopoulos, G. Kambourakis, K. Barbatsalou, "On the detection of lateral movement through supervised machine learning and an open-source tool to create turnkey datasets from sysmon logs," International Journal of Information Security, 2023, Volume 22, pp. 1893–1919

[16] Langchain, https://www.langchain.com/

[17] M. Shrestha *et al*., "ProvSec: Cybersecurity System Provenance Analysis Benchmark Dataset," 2023 IEEE/ACIS 21st International Conference on Software Engineering Research, Management and Applications (SERA), Orlando, FL, USA, 2023, pp. 352–357