Efficient Query Answering over Threat Intelligence Knowledge Graphs

Emmanuel Innocent Umoh Sanju Tiwari Kusum Lata

May 28, 2025

Abstract

In the evolving landscape of cybersecurity, access to accurate and timely cyber threat intelligence is critical to defend against modern attacks. Knowledge Graphs (KGs) provide a powerful means to structure and interrelate complex cyber data, but querying them effectively, especially at scale using languages like Cypher, remains a major challenge. This work presents a lightweight and efficient framework for querying Threat Intelligence Knowledge Graphs (TIKGs) using manually generated Cypher queries. We review key research on Cypher query generation, analyze limitations in current approaches, and design a structured methodology tailored to threat analysis. Our system, implemented using Neo4j, processes real-world cyber data, including activities of threat actors such as SilverTerrier, and demonstrates strong performance in retrieving relevant, semantically rich intelligence. The results confirm that well-structured manual Cypher queries, when aligned with optimized graph schemas, can significantly enhance the effectiveness and precision of threat analysis tasks for analysts and researchers.

keywords: Threat Intelligence, Knowledge Graphs, Cypher Query generation, Natural Language Model, Neo4j, question answering.

1 Introduction

The increasing sophistication, speed, and scale of modern cyberattacks demand intelligent systems that can rapidly interpret complex security data. Threat intelligence knowledge graphs (TIKG) have emerged as a robust solution to represent cyber entities and their relationships, linking threat actors, malware, attack techniques, indicators of compromise (IoC), and other intelligence artifacts in a semantically rich structure [14, 8, 19]. These graphs facilitate advanced reasoning and support key tasks such as actor profiling, infrastructure tracking, and campaign correlation.

Despite their expressive power, querying TIKGs remains a significant challenge. Most cybersecurity knowledge graphs are stored in graph databases like

Neo4j, where Cypher serves as the primary query language. However, writing Cypher queries manually requires in-depth knowledge of the graph schema and query syntax, making it difficult for analysts to interact with the data efficiently [20]. Fully automated approaches to translating natural language to Cypher have emerged [13, 16], but often suffer from schema misalignment, limited domain adaptation, or lack of contextual understanding, particularly in security-specific use cases [4].

The key objectives of this study are as follows:

1. To develop a hybrid Cypher query generation framework capable of transforming user queries mostly manual into executable graph queries for TIKG environments.

2. To design and implement a modular architecture that supports knowledge extraction, graph construction, and scalable query processing in Neo4j.

3. To evaluate the effectiveness of the framework using real-world cybersecurity scenarios, measuring the accuracy, relevance, and performance of the generated queries in tasks such as actor tracking, malware linkage, and campaign investigation.

   This work presents a practical and scalable framework for querying Threat Intelligence Knowledge Graphs using manually constructed Cypher queries. Unlike automated methods, our approach emphasizes precision, domain alignment, and interpretability, critical qualities in cybersecurity operations that focus on prominent actors such as *SilverTerrier* which is a major threat actor in the Nigerian cybersecurity landscape. The framework is designed to support threat analysis tasks such as actor profiling, malware tracing, and campaign attribution within a Neo4j property graph environment.

   The remainder of this paper is organized as follows. Section 2 reviews related work on Cypher query generation and threat knowledge modeling. Section 3 introduces the proposed manual Cypher query framework and its architectural components. Section 4 presents the experimental setup, use cases, and results. Finally, Section 5 summarizes the contributions and outlines the directions for future work.

2 Literature Survey

Recent advances in cyber threat intelligence have seen the integration of knowledge graphs (KG) and machine learning techniques to improve understanding, extraction, and querying of cybersecurity data. Numerous studies have contributed unique approaches to this domain:

Table 1: Refined Summary of Literature on Cypher Query Generation

Paper	**Year**	**Methodology**	**Findings**	**Limitation**	**Query Gen.**
[12]	2025	Used schema filtering to guide Cypher translation	Boosted Cypher precision using schema awareness	Depends on schema quality	Automatic
[15]	2025	Created manual Cypher templates for NL dataset	High-quality synthetic Cypher query pairs	Manual process is time-consuming	Manual
[16]	2024	Generated synthetic Cypher queries with LLMs	40% benchmark improvement on test set	Limited real-world coverage	Automatic
[13]	2024	Fine-tuned LMs for NL-Cypher translation	Improved accuracy on NL–Cypher pairs	Scalability issues with large graphs	Automatic
[10]	2024	Evaluated Cypher prompts for LLMs	Reported LLM performance on Cypher tasks	Limited to cybersecurity domain	Manual
[2]	2024	Transformed RDF queries to Cypher	Benchmarked 10K questions, 7.8M entities	Complex Cypher mappings	Automatic
[17]	2023	Used BERT + GraphSAGE + Transformer	Robust NL to Cypher conversion	High computational demand	Automatic
[3]	2023	Manually created schema-driven queries	Highlighted Cypher syntax issues	No benchmark dataset provided	Manual
[5]	2023	Fuzz tested Cypher engines	Discovered engine parsing flaws	Not analyst-focused	Automatic
[1]	2023	Unified graph path handling	Worked well on large path queries	Focused only on path queries	Automatic
[6]	2023	Surveyed query strategies for KGs	Mapped Cypher among emerging standards	No new model proposed	Automatic
[11]	2022	Unified graph query parsing via IR	Improved parsing accuracy by 11%	Syntax mapping complexity	Automatic
[4]	2022	Released 10K NL–Cypher pairs	Exposed translation challenges in Cypher	Hard for models to generalize	Manual
[14]	2021	Built CTI graph in Neo4j	Modeled 10K+ nodes, Cypher-based QA supported	Limited scope and data variety	Manual
[20]	2019	Used query patterns over KG	Improved structural matching	Requires pattern sketching	Manual
[21]	2015	Template-based QA over RDF	Enhanced RDF query accuracy	Low scalability in RDF to Cypher	Manual

Recent advances in query answering over knowledge graphs have emphasized the use of *Cypher*, the native query language of property graph databases such as Neo4j, especially in cybersecurity and semantic web domains. Several studies have focused on improving the translation of queries from natural language (NL) to Cypher using neural models and large language models (LLM) [12, 16, 7]. Ozsoy et al. [12] and Tiwari et al. [16] proposed frameworks like *Text2Cypher* and *SynthCypher*, achieving notable improvements in query translation accuracy through schema filtering and synthetic data generation.

In parallel, schema-aware approaches, such as schema filtering techniques [12] and domain-adaptive NL interfaces [10], have improved the precision of Cypher query generation. However, these methods often rely on the availability of well-defined graph schemas. Hybrid models that combine language and structural embeddings, such as SPCQL [4], have demonstrated improved performance but remain computationally expensive.

On the manual generation front, studies such as Guo et al. [4], Sarhan and Spruit [14], and Zheng and Zhang [20] have emphasized human-generated Cypher queries that were crafted manually. These efforts underscore the importance of human insight in capturing the complexity of graph structures, particularly in domain-specific contexts.

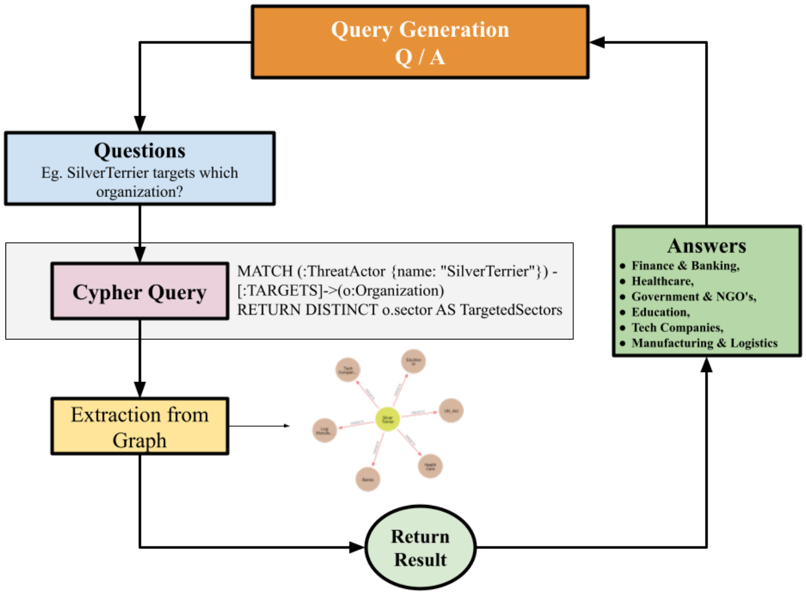
Beyond Cypher, other works addressed SPARQL and RDF-based querying, such as *WDAqua-core0* [9], *GAnswer2* [21], and Semantic Web QA systems [18] provide transferable techniques for query interpretation and entity-relation modeling. Although these are not based on property graphs, their design informs entity linking and structural query generation applicable to Cypher environments.

A major challenge in the literature is the lack of generalization to real-world data, the dependency on template-based systems [20], or poor performance with complex and ambiguous queries [21]. Despite the trend toward automatic query generation, manual efforts remain crucial in domains requiring high precision and semantic clarity.

Collectively, these contributions provide a comprehensive understanding of the current landscape in cipher-based query-response systems. The field continues to evolve toward automatic, scalable, and context-aware solutions capable of supporting real-time applications in threat intelligence and beyond.

3 Proposed Methodology

In the dynamic cybersecurity environment, timely access to contextualized and accurate threat intelligence is essential for proactive defense. Yet, keyword-based search strategies or document-based information retrieval systems tend to be inadequate when it comes to addressing complex and interconnected cyber threat information. This problem has motivated interest in Question Answering (QA) systems based on Threat Knowledge Graphs (Threat KGs), semantic graphs that organize cyber threat information into entities (e.g., actors, malware, IPs) and relationships (e.g., uses, targets, exploits). Question Answering on Threat KGs is the technique of transforming the user queries expressed either in natural or structured forms, into formal graph traversal queries (usually in query languages such as Cypher for Neo4j or SPARQL for RDF-based KGs), and fetch accurate, understandable, and actionable answers. Instead of manually checking reports or feeds, a user might query. This will reduce analyst workload considerably, improves situational awareness, and facilitates real-time decision-making.



```
MATCH (:ThreatActor {name: "SilverTerrier"}) -
[:TARGETS]->(o:Organization)
RETURN DISTINCT o.sector AS TargetedSectors
```

generation and result extraction for identifying organizations targeted by SilverTerrier

3.1 Query Generation and Execution over Threat Intelligence Knowledge Graphs

Figure 1 illustrates the pipeline for question answering over a Threat Intelligence Knowledge Graph (KG), specifically querying targets of a threat actor using Cypher in Neo4j. Each stage of the process plays a crucial role in transforming natural language into meaningful insights derived from structured cyber threat data.

3.1.1 Questions (User Input)

The process begins with a natural language query posed by a threat analyst. For instance:

> *"SilverTerrier targets which organization?"*

Table 2: Components of Question Answering over Threat Knowledge Graphs

Stage	**Function**
Question Interpretation	Analyzes natural language queries to identify user intent and extract key elements such as entities and relations.
Entity Recognition	Detects and highlights relevant cyber entities like threat actors, malware names, IP addresses, or organizations.
Relation Detection	Determines the semantic relationships between entities, such as “uses”, “targets”, or “associated with”.
Entity Linking	Maps identified entities to their corresponding nodes in the knowledge graph using labels or external references.
Query Generation	Translates the interpreted question into a structured query language (e.g., Cypher or SPARQL) based on graph schema.
Query Execution	Executes the generated query on the graph database to retrieve matched subgraphs or data points.
Answer Retrieval	Filters and formats the output of the query execution into a human-readable answer or a result set.
Visualization	Displays results visually through nodes and relationships in a graph view to support further exploration and reasoning.

Here, `SilverTerrier` is a known Nigerian cyber threat actor. The question's intent is to determine the organizations, classified by sector, that have been previously targeted by this actor. This forms the basis of query interpretation and subsequent graph traversal.

3.1.2 Cypher Query (Translation Layer)

The system converts the user's question into a Cypher query:

```
MATCH (:ThreatActor {name: "SilverTerrier"})-[:TARGETS]->(o:Organization)
RETURN DISTINCT o.sector AS Sector
```

This query searches for all `Organization` nodes linked to the `ThreatActor` node named “SilverTerrier” via the `TARGETS` relationship, returning a distinct list of sectors.

3.1.3 Extraction from Graph (Query Execution)

The query is executed on the Neo4j graph database, and the system retrieves all matching subgraphs that correspond to the defined pattern. For example, nodes labeled as `Organization` may contain a `sector` property, which can be used to classify targets such as:

- Finance & Banking
- Healthcare
- Government & NGOs

- Education
- Tech Companies
- Manufacturing & Logistics

3.1.4 Return Result (Answer Presentation)

The final step is returning the result in a human-readable format. The output could be a list of sectors or an interactive dashboard. The results can also be visualized in the form of a graph, helping users to understand the relationships and patterns in the threat landscape.

3.1.5 Visualization

To demonstrate the effectiveness of the proposed Cypher query generation framework, we provide both graphical and tabular visualizations of the query results.

The visual representation in Figure 1 shows the central node *SilverTerrier* and other related *Threat Actors* connected to several organization nodes through relationships such as `TARGETS`, `Used_Tool`, and `ORIGINATES_FROM` etc. Each organization is labeled with its sector (e.g., finance, telecom, government), enhancing semantic clarity. This graph-based visualization allows analysts to explore patterns of targeting behavior and actor-tool interactions.

This methodology illustrates how complex cyber intelligence questions can be mapped to structured graph queries, enhancing the efficiency of threat analysis and facilitating automated reasoning over threat knowledge graphs.

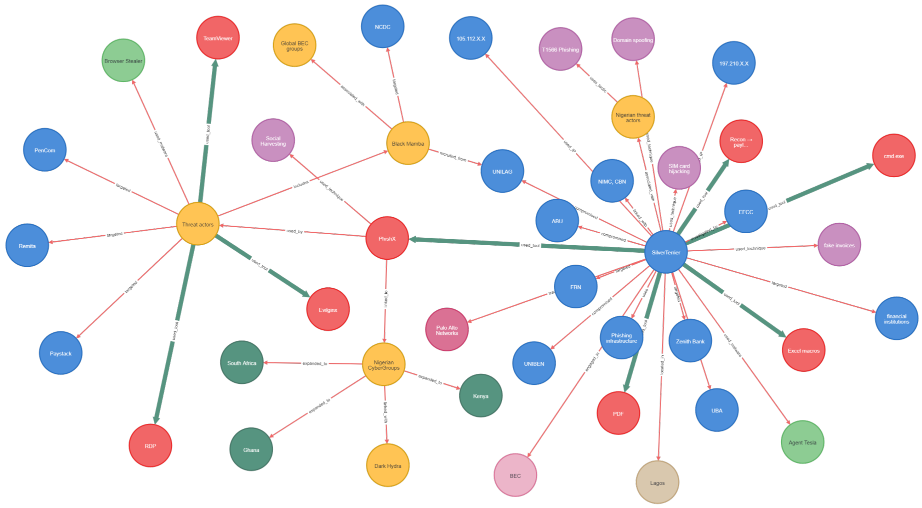


Figure 1: Graph visualization of SilverTerrier and its relationships with organizations and tools.

To complement the visual graph, Table 3 presents examples of natural language (NL) queries issued by analysts, the corresponding Cypher queries is also generated to give the desired results returned from the graph database. This dual-format visualization supports both structural exploration and query-based investigation of threat intelligence.

Table 3: Examples of NL queries, Cypher translations, and output results from the TIKG

NL Query	**Cypher Query**	**Result Returned**
Which Nigerian organizations has SilverTerrier targeted between 2020 and 2024?	MATCH (a:Entity {name: "SilverTerrier"})-[:targeted]-> (c:Entity) RETURN c.name	FBN, Zenith Bank, GTB
What tools has SilverTerrier used in BEC attacks targeting Nigerian banks?	MATCH (a:Entity {name: "SilverTerrier"})-[:used_tool]-> (t:Tool) RETURN t.name	PhishX, Excel macros
Which Nigerian organizations have been attacked by SilverTerrier using Mimikatz?	MATCH (attacker:Entity {name: "SilverTerrier"})-[:linked_with]-> (org:Entity) RETURN DISTINCT org.name	NIMC, CBN
What phishing techniques are often deployed by SilverTerrier in Nigeria?	MATCH (a:Entity {name: "SilverTerrier"})-[:used_technique]-> (b:Technique) RETURN b.name	Domain spoofing, Fake invoices
What tools do Nigerian threat actors use for credential harvesting?	MATCH (a:Group {name: "Threat actors"})-[:used_tool]-> (b:Tool) RETURN b.name	Evilginx, BrowserStealer

This multi modal output linking graph exploration with query response and inspection enhances the interpretability of cyberthreat data and supports better data-driven decisions by the cybersecurity analysts.

3.2 Applications of QA over Threat Knowledge Graphs

Question Answering (QA) over Threat Intelligence Knowledge Graphs (TIKGs) offers transformative capabilities across a wide range of cybersecurity domains. By embedding QA into TIKGs, security operations become not only more intelligent but also automated, interactive, and strategically responsive. These systems enable analysts to go beyond keyword search and engage in structured reasoning over threat knowledge.

Table 4 summarizes several practical application areas, example questions, and the operational benefits of QA over threat knowledge graphs.

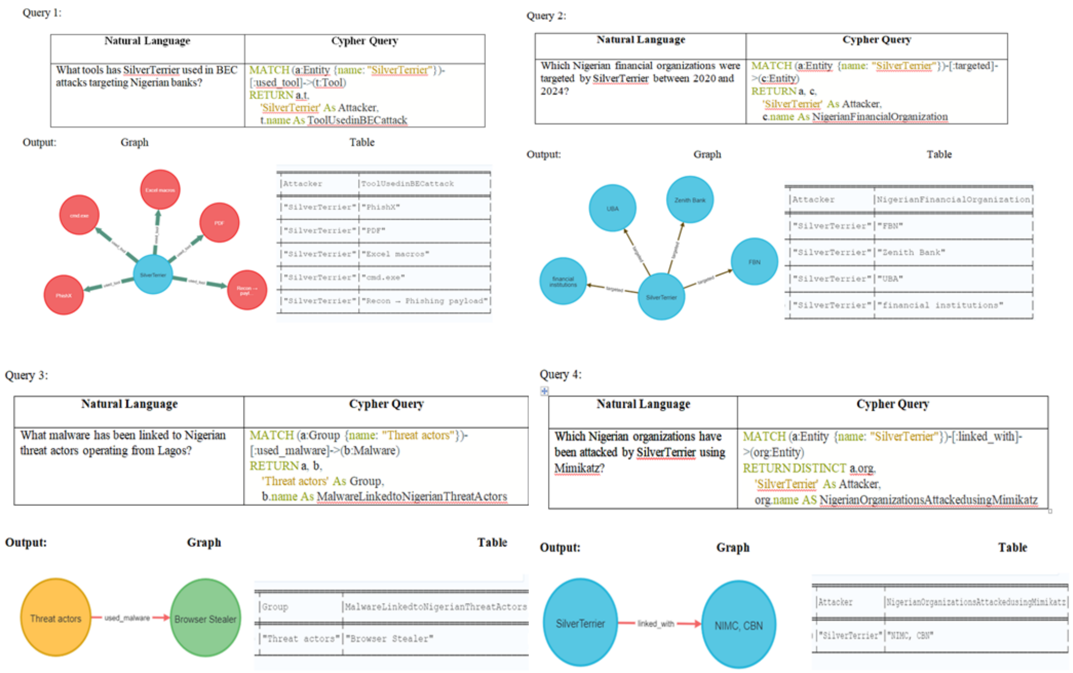
Table 4: Applications of Question Answering over Threat Knowledge Graphs

Application Area	**Example Question**	**Purpose / Benefit**
Threat Actor Profiling	What are the known attack techniques used by SilverTerrier?	Builds a detailed profile of threat actors and their TTPs for intelligence and monitoring.
Vulnerability Assessment	Which vulnerabilities are exploited in Nigerian telecom infrastructure?	Identifies exploitable vulnerabilities in critical sectors to guide patch management and risk mitigation.
Incident Investigation	Which IP addresses were used in the recent malware campaign?	Traces attacker infrastructure and related threat entities to support forensic analysis.
Attack Prediction	What sectors might SilverTerrier target next based on historical patterns?	Supports proactive defense and early warning through graph-based behavioral analysis.
Real-time Threat Monitoring	What are today's top threats affecting African financial institutions?	Enables real-time awareness by linking live feeds with structured threat intelligence.
Security Awareness Training	What methods did SilverTerrier use in past BEC attacks?	Delivers context-rich, scenario-based learning for training security personnel.
Threat Intelligence Reporting	Summarize the major threat actor activities in the past 30 days.	Automates reporting tasks using graph queries for rapid, data-driven threat summaries.

4 Results

To evaluate the efficiency and expressiveness of the proposed framework for query answering over Threat Intelligence Knowledge Graphs (TIKGs), we conducted several experiments using a Neo4j-based implementation. The graph schema included entities such as `ThreatActor`, `Tools`,`Malware`, `Organization`, and `Sector`, connected through semantic relationships such as `USES`, `TARGETS`, and `ASSOCIATED_WITH`.

We generate a test suite of four representative security-related questions extracted from the graph above, as shown in figure 2. Each question was translated into a Cypher query and executed on the knowledge graph. The framework returned correct and contextually relevant answers, demonstrating a strong alignment between the graph structure and the query semantics.



Query 1:

Natural Language	Cypher Query
What tools has SilverTerrier used in BEC attacks targeting Nigerian banks?	MATCH (a:Entity {name: "SilverTerrier"})- [:used_tool]->(t:Tool) RETURN a,t, 'SilverTerrier' As Attacker, t.name As ToolUsedinBECattack

Output: Graph Table

```
|Attacker        |ToolUsedinBECattack         |
|"SilverTerrier"|"PhishX"                     |
|"SilverTerrier"|"PDF"                        |
|"SilverTerrier"|"Excel macros"               |
|"SilverTerrier"|"cmd.exe"                    |
|"SilverTerrier"|"Recon → Phishing payload"|
```

Query 2:

Natural Language	Cypher Query
Which Nigerian financial organizations were targeted by SilverTerrier between 2020 and 2024?	MATCH (a:Entity {name: "SilverTerrier"})-[:targeted]- >(c:Entity) RETURN a, c, 'SilverTerrier' As Attacker, c.name As NigerianFinancialOrganization

Output: Graph Table

```
|Attacker        |NigerianFinancialOrganization|
|"SilverTerrier"|"FBN"                         |
|"SilverTerrier"|"Zenith Bank"                 |
|"SilverTerrier"|"UBA"                         |
|"SilverTerrier"|"financial institutions"      |
```

Query 3:

Natural Language	Cypher Query
What malware has been linked to Nigerian threat actors operating from Lagos?	MATCH (a:Group {name: "Threat actors"})- [:used_malware]->(b:Malware) RETURN a, b, 'Threat actors' As Group, b.name As MalwareLinkedtoNigerianThreatActors

Output: **Graph** **Table**

```
|Group           |MalwareLinkedtoNigerianThreatActors
|"Threat actors"|"Browser Stealer"
```

Query 4:

Natural Language	Cypher Query
Which Nigerian organizations have been attacked by SilverTerrier using Mimikatz?	MATCH (a:Entity {name: "SilverTerrier"})-[:linked_with]- >(org:Entity) RETURN DISTINCT a,org, 'SilverTerrier' As Attacker, org.name AS NigerianOrganizationsAttackedusingMimikatz

Output: **Graph** **Table**

```
|Attacker        |NigerianOrganizationsAttackedusingMimikatz|
|"SilverTerrier"|"NIMC, CBN"                                |
```

Figure 2: Result of Question Answering over TIKG

The Cypher-based graph constructed for this work contains a total of **38 nodes** and **47 directed edges**, corresponding to **47 subject–predicate–object (SPO) triples**. These nodes represent various threat intelligence entities such as actors, tools, techniques, organizations, malware, and countries. Relationships among these entities were modeled using domain-specific predicates such as `used_tool`, `targeted`, `associated_with`, and `investigated_by`. Each edge in the graph captures a directed and semantically meaningful interaction, forming the core structure of the Threat Intelligence Knowledge Graph (TIKG).

Table 5 shows representative SPO triples extracted from the graph:

Table 5: Examples of Subject–Predicate–Object Triples in the TIKG

Subject	**Predicate (Relationship)**	**Object**
SilverTerrier	used_tool	PhishX
SilverTerrier	engaged_in	BEC
Threat actors	used_malware	Browser Stealer
Black Mamba	associated_with	Global BEC groups
Nigerian CyberGroups	expanded_to	Kenya

This structured representation of the threat landscape supports downstream tasks such as manual query answering using Cypher, threat actor profiling, infrastructure tracing, and incident investigation. The triple-based model also ensures that queries return contextually relevant and interpretable results, aligned with the semantics of the knowledge graph.

5 Conclusion and Future Work

In this work, we developed a scalable and efficient framework for query answering over Threat Intelligence Knowledge Graphs (TIKGs), with a focus on Cypher-based querying in property graph databases. By integrating manual query generation strategies. This system will enable timely and semantically rich access to actionable cybersecurity insights. A comprehensive survey of recent advancements in Cypher query generation and semantic parsing was conducted, forming the foundation for the design of a multi-layered architecture. The proposed solution demonstrated its effectiveness in real-world use cases, such as threat actor profiling and attack pattern detection, using structured CTI data within Neo4j.

Future work will focus on the following directions:

- **Multilingual Natural Language Query Support:** Expanding the system front-end interface to support NL queries in multiple languages to serve a larger global cybersecurity community.
- **Real-time Threat Feed Integration:** Incorporating streaming data sources (e.g. threat intelligence APIs) to continuously update the knowledge graph and allow live querying of evolving cyber incidents.

Overall, this research contributes a robust foundation for interactive, intelligent querying of cyber threat knowledge, with strong potential to support both research and operational defense activities.

References

[1] Benjamín Farías, Wim Martens, Carlos Rojas, and Domagoj Vrgoč. Pathfinder: A unified approach for handling paths in graph query languages. *arXiv preprint arXiv:2306.02194*, 2023.

[2] Yanlin Feng, Simone Papicchio, and Sajjadur Rahman. Cypherbench: Towards precise retrieval over full-scale modern knowledge graphs in the llm era. *arXiv preprint arXiv:2412.18702*, 2024.

[3] Adam Forsberg and Andreas Lepik. Random generation of semantically valid cypher queries. *LU-CS-EX*, 2023.

[4] Aibo Guo, Xinyi Li, Guanchen Xiao, Zhen Tan, and Xiang Zhao. Spcql: A semantic parsing dataset for converting natural language into cypher. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 3973–3977, 2022.

[5] Ziyue Hua, Wei Lin, Luyao Ren, Zongyang Li, Lu Zhang, Wenpin Jiao, and Tao Xie. Gdsmith: Detecting bugs in cypher graph database engines. In *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*, pages 163–174, 2023.

[6] Arijit Khan. Knowledge graphs querying. *arXiv preprint arXiv:2305.14485*, 2023.

[7] Hongyi Li, Ze Shi, Chengwei Pan, Di Zhao, and Nan Sun. Cybersecurity knowledge graphs construction and quality assessment. *Complex & Intelligent Systems*, 10(1):1201–1217, 2024.

[8] Zhenyuan Li, Jun Zeng, Yan Chen, and Zhenkai Liang. Attackg: Constructing technique knowledge graph from cyber threat intelligence reports. In *European Symposium on Research in Computer Security*, pages 589–609. Springer, 2022.

[9] Vanessa Lopez, Christina Unger, Axel-Cyrille Ngonga Ngomo, and Elena Cabrio. Wdaqua-core0: A question answering component for the research community. In *International Semantic Web Conference*, pages 84–89. Springer, 2017.

[10] Siraj Munir and Alessandro Aldini. Towards evaluating large language models for graph query generation. *arXiv preprint arXiv:2411.08449*, 2024.

[11] Lunyiu Nie, Shulin Cao, Jiaxin Shi, Jiuding Sun, Qi Tian, Lei Hou, Juanzi Li, and Jidong Zhai. Graphq ir: Unifying the semantic parsing of graph query languages with one intermediate representation. *arXiv preprint arXiv:2205.12078*, 2022.

[12] Makbule Gulcin Ozsoy. Enhancing text2cypher with schema filtering. *arXiv preprint arXiv:2505.05118*, 2025.

[13] Makbule Gulcin Ozsoy, Leila Messallem, Jon Besga, and Gianandrea Minneci. Text2cypher: Bridging natural language and graph databases. *arXiv preprint arXiv:2412.10064*, 2024.

[14] Injy Sarhan and Marco Spruit. Open-cykg: An open cyber threat intelligence knowledge graph. *Knowledge-Based Systems*, 233:107524, 2021.

[15] Saber Soleymani, Nathan M Gravel, Krzysztof Kochut, and Natarajan Kannan. Task splitting and prompt engineering for cypher query generation in domain-specific knowledge graphs. *bioRxiv*, pages 2025–04, 2025.

[16] Aman Tiwari, Shiva Krishna Reddy Malay, Vikas Yadav, Masoud Hashemi, and Sathwik Tejaswi Madhusudhan. Synthcypher: A fully synthetic data generation framework for text-to-cypher querying in knowledge graphs. *arXiv preprint arXiv:2412.12612*, 2024.

[17] Quoc-Bao-Huy Tran, Aagha Abdul Waheed, and Sun-Tae Chung. Robust text-to-cypher using combination of bert, graphsage, and transformer (cobgt) model. *Applied Sciences*, 14(17):7881, 2024.

[18] Christina Unger, Andre Freitas, and Philipp Cimiano. Question answering over the semantic web. *Web Semantics: Science, Services and Agents on the World Wide Web*, 30:39–58, 2014.

[19] Jian Wang, Tiantian Zhu, Chunlin Xiong, and Yan Chen. Multikg: Multi-source threat intelligence aggregation for high-quality knowledge graph representation of attack techniques. *arXiv preprint arXiv:2411.08359*, 2024.

[20] Weiguo Zheng and Mei Zhang. Question answering over knowledge graphs via structural query patterns. *arXiv preprint arXiv:1910.09760*, 2019.

[21] Lei Zou, Jin Huang, Heng Wang, Jeffery Xu Yu Zhao, and Ling Qin. Ganswer2: Natural language question answering over rdf. *The VLDB Journal*, 24:589–613, 2015.