AgCyRAG: an Agentic Knowledge Graph based RAG Framework for Automated Security Analysis

Kabul Kurniawan[1,2,*], Rayhan Firdaus Ardian[1], Elmar Kiesling[3] and Andreas Ekelhart[4,5]

[1]*Department of Computer Science and Electronics, Universitas Gadjah Mada, Indonesia*
[2]*Center for Cryptography and Cybersecurity Research, Universitas Gadjah Mada, Indonesia*
[3]*WU Wien, Institute for Data, Process and Knowledge Management, Vienna, Austria*
[4]*SBA Research, Vienna, Austria*
[5]*University of Vienna, Vienna, Austria*

Abstract

Cybersecurity analysis is a critical activity that aims to detect threats, respond to incidents, and ensure organizations' resilience. It is a highly complex task where analysts typically navigate and interpret vast amounts of heterogeneous data across structured and unstructured sources, ranging from system logs and network activity logs to threat intelligence and policy documents. Large Language Models (LLMs) can provide simplified access to this data through a natural language interface and have the potential to unlock advanced analytic capabilities. In this paper, we propose to combine a number of Retrieval-Augmented Generation (RAG) techniques to make this diverse and highly dynamic information accessible to LLMs and enable factual grounding of cybersecurity analyses. A key challenge in this context is that RAG approaches typically focus on unstructured text and often overlook symbolic representations and conceptual relations that are essential in cybersecurity – including network structures, IT assets hierarchies and attack patterns. To address this gap, we propose AgCyRAG: a hybrid Agentic RAG framework that integrates Knowledge Graphs (KGs) and vector-based retrieval to enhance the factual accuracy and contextual relevance of security analyses. The framework orchestrates multiple agents that interpret user queries and adaptively select the optimal retrieval strategy according to the analytical context. The agentic workflows enable systems to combine structured semantic reasoning with vector-based retrieval, resulting in more comprehensive and interpretable security analyses. We validate AgCyRAG by means of three real-world use-cases and demonstrate its ability to support advanced, context-aware security analyses.

Keywords

Agentic, RAG, LLM, Cybersecurity, Knowledge Graph

1. Introduction

Cybersecurity analysis is crucial for safeguarding digital infrastructures by enabling timely detection of threats, effective incident response, and sustained organizational resilience. However, as cyber threats become more sophisticated and widespread, interpreting vast volumes of heterogeneous data, such as system logs, network traffic, and threat intelligence reports, to detect and mitigate malicious activities, has become increasingly difficult. Moreover, this data spans both structured and unstructured formats and is often distributed across siloed systems, which makes timely and accurate analysis inherently challenging [1].

Recent advances in Large Language Models (LLMs) have created new opportunities to enhance cybersecurity operations by supporting defensive, offensive, vulnerability, and risk assessment activities [2]. LLMs offer intuitive, natural language interfaces that can simplify the process of querying and synthesizing information from complex datasets [3]. By understanding and generating human-like text, LLMs can assist analysts with tasks such as identifying suspicious behavior, correlating incidents, and reasoning about adversarial behavior. However, LLMs are prone to hallucinations and lack reliable mechanisms for grounding their outputs in verifiable evidence, which poses significant risks in

RAGE-KG 2025: The Second International Workshop on Retrieval-Augmented Generation Enabled by Knowledge Graphs, co-located with ISWC 2025, November 2–6, 2025, Nara, Japan

kabul.kurniawan@ugm.ac.id (K. Kurniawan); rayhan.firdaus@ugm.ac.id (R. F. Ardian); elmar.kiesling@wu.ac.at (E. Kiesling); andreas.ekelhart@univie.ac.at (A. Ekelhart)
0000-0002-0877-7063 (K. Kurniawan); 0000-0002-0877-7063 (E. Kiesling); 0000-0003-3682-1364 (A. Ekelhart)

high-stakes domains like cybersecurity [4].

To mitigate these limitations, Retrieval-Augmented Generation (RAG) has emerged as a promising technique to enrich the generative capabilities of LLMs by retrieving relevant external information from curated knowledge sources before generating a response. This improves factual grounding and reduces the likelihood of hallucinated or misleading outputs [5]. Despite these advances, current RAG approaches primarily focus on unstructured data and typically process it as disconnected chunks of text[1]. Security analytics, on the other hand, typically requires reasoning about complex cybersecurity knowledge in structured representations [6].

For instance, to understand whether a *PowerShell* command indicates an *illegitimate login*, it is necessary to contextualize it within a broader sequence of system events, user behavior, and known attack patterns. These forms of reasoning often involve multiple dispersed knowledge sources such as event sequences, network topologies, hierarchical asset models and attack patterns to be integrated for the purpose of, e.g., identifying the root cause of an attack and assessing the potential impact. Traditional RAG approaches are typically designed for document retrieval and do not account for neither symbolic and relational structures, nor temporal ones, hence, missing the deeper context behind relationships that are fundamental to cybersecurity.

To address this gap, we introduce AgCyRAG, a hybrid Agentic RAG framework specifically designed for cybersecurity analysis. AgCyRAG provides advanced retrieval paradigms that combine Knowledge Graph (KG)-based reasoning with vector based retrieval in an agentic framework to provide accurate, context-aware, and explainable analytical support. Compared to prior work [7] that introduced RAG techniques in a cybersecurity analytic context, our contribution in this paper is an agentic architecture that combines specialized agents responsible for different tasks such as interpreting user queries, selecting appropriate retrieval strategies (e.g., Cypher, SPARQL, or vector search), and coordinating the synthesis of agents output information into coherent responses.

The agentic design enables AgCyRAG to simulate multi-step reasoning processes typical of human analysts. Structured graph queries can capture precise relationships and causality, while vector search supports flexible retrieval from unstructured sources. By orchestrating these modalities, AgCyRAG delivers comprehensive outputs grounded in both symbolic semantics and empirical evidence.

We demonstrate the effectiveness of AgCyRAG by means of three real-world cybersecurity use cases covering a variety of scenarios such as general cybersecurity analytic question answering and security log analysis within a Cybersecurity Knowledge Graph (CSKG) context. These evaluations highlight the system's ability to improve the precision and interpretability of analyses, as well as enhancing analysts' situational awareness in complex cyber defense tasks.

Overall, our contributions in this paper can be summarized as follows: *(i)* we propose an agentic framework for cybersecurity analysis, *(ii)* we integrate RAG techniques for log-structured data, *(iii)* we provide the LLMs performing the analyses with attack knowledge grounded in MITRE's Att&ck framework, *(iv)* we assess the capabilities of LLM-based log analysis on synthetically generated data in common analytic scenarios (i.e., analyzing login data for suspicious behavior), and *(v)* we demonstrate how the proposed framework can support human analysts and provide and effective and efficient natural language interface to initiate and perform agentic cybersecurity analyses.

The remainder of this paper is organized as follows: Section 2 provides an overview of RAG, GraphRAG, and Agentic RAG approaches in the cybersecurity domain. Section 3 details our proposed methodology, including the conceptual framework and system architecture of our agentic AI framework; Section 4 presents the implementation details of our approach, accompanied by a discussion of the selected use-case applications. Section 6 summarizes our work and outlines potential directions for future work.

[1] http://graphwise.ai/event/webinar-graph-rag-why-your-rag-needs-a-graph/

2. Related Work

The field of cybersecurity analysis has seen a shift towards leveraging knowledge-based systems and LLMs to handle the complexity and sheer volume of data. This section reviews key developments in two areas: *(i)* Cybersecurity Knowledge Graphs and Semantic Security Analysis, and *(ii)* LLMs in cybersecurity, RAG, and particularly the emerging field of agentic RAG for cybersecurity.

Cybersecurity Knowledge Graphs and Semantic Analysis The foundation for structured security analysis was laid by early research into cybersecurity ontologies and semantic approaches. These efforts aimed to create formal vocabularies and models to represent security knowledge in a machine-readable format. Foundational work proposed core ontologies to formalize security terminologies [8, 9]. Other research focused on more specific applications, such as modeling attacks for intrusion detection [10], annotating security resources [11], and describing security incidents [12] and requirements [13]. The use of vocabularies for semantic analysis of log data further advanced this area, providing a more structured way to interpret heterogeneous information [1].

Building on these early ontologies, the field shifted toward creating comprehensive cybersecurity knowledge graphs (CKGs) that could integrate and correlate diverse data sources. The development of an ontology for building these KGs was discussed in [14]. [6] addressed the challenge of logs and threat intelligence integration via the SEPSES Knowledge Graph. Subsequent work introduced more specialized KGs, such as the ICS-SEC KG for industrial control systems [15] and the ATT&CK-KG for linking attacks to adversary tactics [16]. Recent surveys have extensively reviewed the applications of CKGs, highlighting their potential for threat intelligence and automated reasoning by revealing connections that are otherwise difficult to detect [17, 18, 19, 20].

LLM, Retrieval-Augmented Generation and Agentic Frameworks The advent of LLMs has introduced new possibilities for cybersecurity by enabling natural language interfaces and advanced reasoning capabilities [2]. Broader surveys on LLMs discuss core concepts, architectures, and challenges like hallucination, which is a significant concern when using LLMs for fact-sensitive tasks like security analysis [3, 21]. To mitigate such issues and ground analyses in factual data, researchers have explored methods to combine LLMs with external knowledge. To overcome the limitations of relying solely on LLMs, RAG has emerged as a promising technique. Lewis et al. [5] introduced the foundational concept of RAG, which combines a retriever (to find relevant information) with a generator (the LLMs) to produce more accurate, grounded responses. This approach has been adapted for cybersecurity. Kurniawan et al. [7] introduced CyKG-RAG, which specifically leverages a knowledge graph for the retrieval step to provide LLMs with structured cybersecurity knowledge.

More recently, the concept of agentic RAG has gained traction. This involves orchestrating multiple specialized agents to perform different tasks, such as querying databases, retrieving documents, and synthesizing information, in a dynamic workflow [22, 23]. Specific agentic frameworks for cybersecurity have been proposed, such as ARCeR [24], an agentic RAG for defining cyber ranges, and CRAKEN [25], an LLM agent with knowledge-based execution for cybersecurity. Paduraru et al. [26] also explored integrating LLMs and agentic AI for network security, demonstrating the growing interest in this paradigm. Our proposed AgCyRAG framework builds upon these ideas by creating a hybrid agentic system that intelligently combines structured knowledge graph reasoning (via Cypher and SPARQL agents) with vector-based retrieval to provide a more comprehensive and accurate approach to automated security analysis.

3. AgCy-RAG Framework

In this section, we discuss the overall architecture of the AgCy-RAG Framework as well as retrieval mechanism in the context of security log analysis.

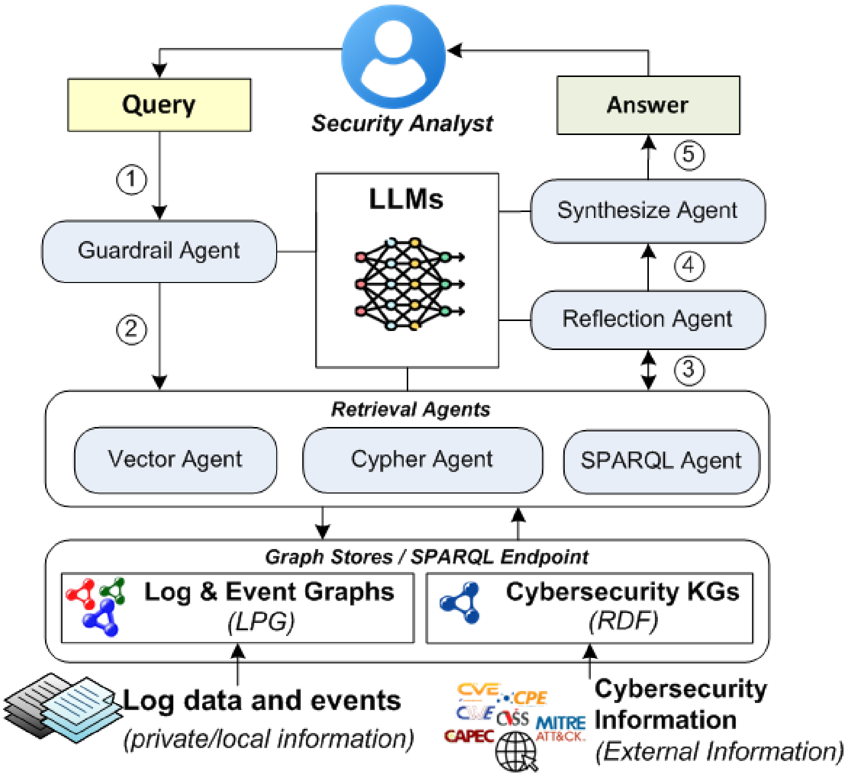


Figure 1: AgCy-RAG Architecture

3.1. The Agentic KG-RAG Architecture

This section details the architecture and operational workflow of the AgCyRAG framework. The framework has a modular architecture that provides separate agents based on their functionality i.e., task and role.

The workflow orchestrates five specialized agents, each designed to handle specific aspects of the query pre-processing, query execution & retrieval and response generation processes. Query pre-processing is managed by a *guardrail* agent that filters irrelevant queries. Query execution is handled by three specialized *retrieval* agents that coordinate retrieval from both local sources such as event logs [1] and external cybersecurity knowledge graphs (KGs) [6]. Response generation is performed by *reflection* and *synthesize* agents, which evaluate the generated response and integrate it into a coherent and relevant answer. Figure 1 depicts an overview of AgCyRAG architecture. In the following, we explain each step of the workflow:

Guardrail Agent The workflow starts with a guardrail agent that filters incoming queries to ensure they address cybersecurity topics (1). Once the query passes the relevance check, the agent routes the query to the appropriate set of *Retrieval Agents* (2). Internally, the *Guardrail Agent* also performs entity extraction through Named Entity Recognition (NER) to identify critical parameters such as usernames, system names, dates, or IP addresses, etc. which become search tokens for subsequent retrieval steps. Based on both the intent and the entities found, the *Guardrail Agent* determines the appropriate routing, e.g. whether dispatching the query to the *Vector Agent* for semantic similarity search to capture unstructured context, to the *Cypher Agent for structured log graph retrieval*, as well as to the *SPARQL Agent* for CSKG related queries. This selective routing prevents unnecessary resource usage and ensures that only the relevant retrieval agents are triggered, maintaining efficiency and precision throughout the pipeline.

Retrieval Agents The retrieval agents provide three types of retrieval services: (i) Vector-based semantic search handled by the Vector Agent; (ii) Log graph retrieval via the Cypher Agent; (ii) CSKG query via the SPARQL Agent. This hybrid approach allows the system to handle both symbolic and sub-symbolic information.

- **Vector Agent** This agent specializes in performing semantic similarity retrieval over log graphs. This is achieved by mapping the processed query and extracted entities into a high-dimensional vector space (embeddings). The agent then searches an embedding index stored within a Labelled Property Graph (LPG) database to identify relevant log chunks. The Vector Agent contributes a crucial task when symbolic graph queries alone cannot capture nuanced or context-rich patterns that exist in unstructured raw log data.
- **Cypher Agent** The *Cypher Agent* operates on a LPG, which represents structured relationships extracted from raw log data, which are constructed from log data and events in local environments (e.g., System logs). Using the Cypher query language, this agent retrieves structured local insights about system events, network behaviors, and user activity.
- **SPARQL Agent** The SPARQL Agent focuses on enriching the analysis with external cybersecurity intelligence. It connects to the CSKG, represented in Resource Description Framework (RDF) and accessed via a SPARQL endpoint. This KG integrates a variety of structured threat intelligence from standardized sources such as CVE, CWE, CAPEC and MITRE ATT&CK. The agent operates through Model Context Protocol (MCP), which allows the LLM to invoke predefined SPARQL query functions hosted on a MCP server.

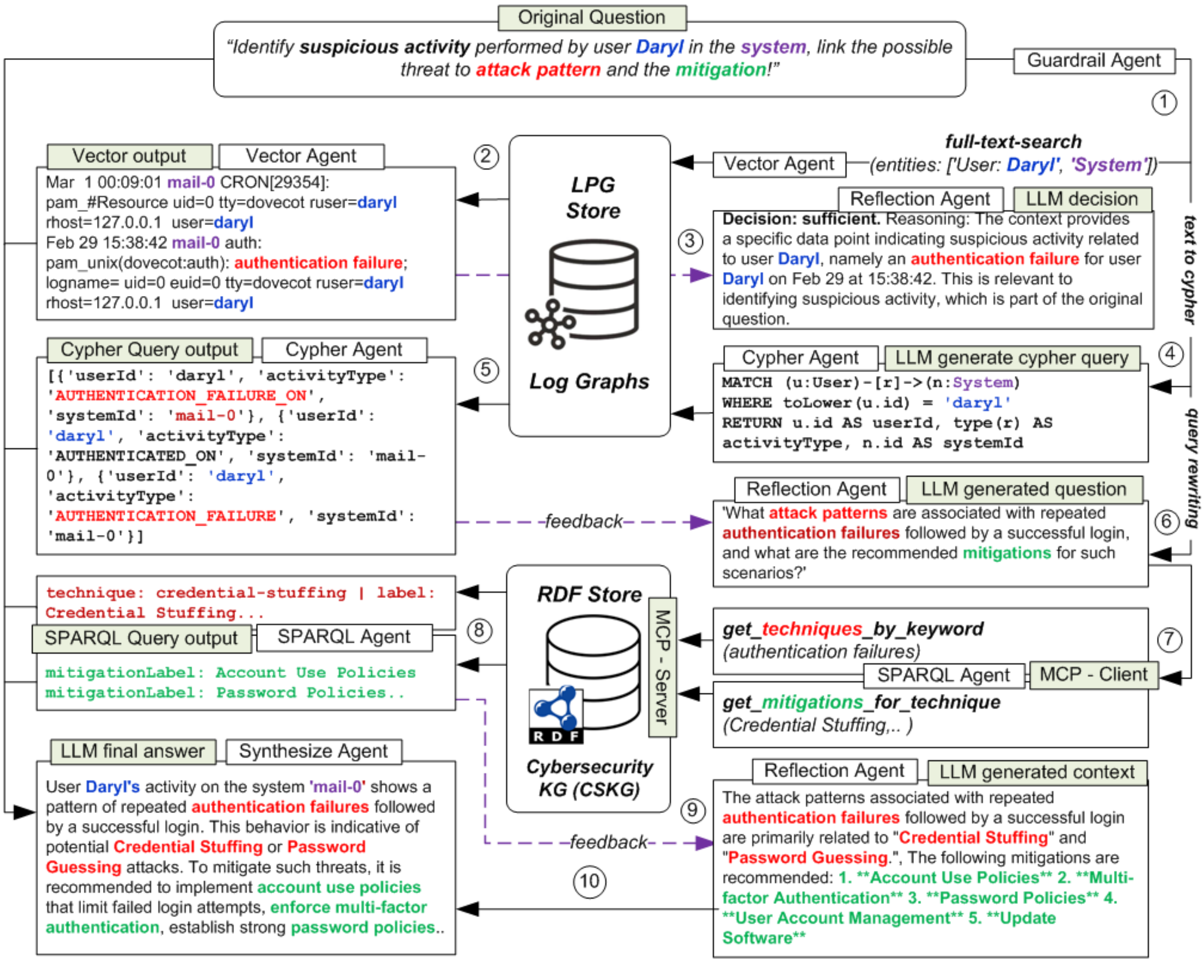
Reflection Agent receives all retrieved evidence and conducts a reasoning pass to evaluate its coherence and relevance. This agent assesses the sufficiency of retrieved information, and decides whether additional refinement or retrieval rounds are necessary. It acts as a quality control loop — if log data shows suspicious activity but lacks contextual threat intelligence, it will trigger another retrieval from external log graphs such as CSKG. For example, once the *Reflection Agent* detects that the log and semantic search results point to suspicious activity but lack an explicit mapping to known attack techniques, it triggers the *SPARQL Agent*. Finally, when the contextual output is deemed satisfactory, it passed the output to the *Synthesis Agent* (4).

Synthesis Agent integrates and summarizes the findings into a coherent and human-readable answer. This synthesis process is grounded in RAG, using prompt templates and evidence injection strategies to generate structured, explainable responses. It merges structured log data, semantic search results, and threat intelligence into a single comprehensive answer (5).

3.2. Retrieval Mechanism

In this section, we explain the retrieval process of our AgCyRAG framework by means of security analysis use-cases (demonstrated in more detail in Section 4). It specifically aims to identify suspicious activity, map it to known attack patterns (i.e., combinations of attack techniques) and recommend potential mitigations. As shown in Figure 2, the retrieval mechanism involves a use-case of detecting suspicious activity performed by a user named *Daryl* in an authentication log (*auth.log*).

The workflow starts with a natural language query from the security analyst, i.e., *"Identify suspicious activity performed by user Daryl in the system, link the possible threat to attack patterns and the mitigation!"*. The first step in this process involves validating the relevancy, which is performed by the *Guardrail Agent*. Once relevance is confirmed, the framework performs a full-text search (1) against the log graph stored in a LPG database (e.g., Neo4J database). Based on the original query, the LLM performs entity extraction though NER, such as "User: Daryl" and *"System"* and uses them as search tokens. The *Guardrail Agent* then routes the query to the *Vector Agent* for a semantic similarity search.



"Identify *suspicious activity* performed by user *Daryl* in the *system*, link the possible threat to *attack pattern* and the *mitigation*!"

```
Mar  1 00:09:01 mail-0 CRON[29354]:
pam_#Resource uid=0 tty=dovecot ruser=daryl
rhost=127.0.0.1  user=daryl
Feb 29 15:38:42 mail-0 auth:
pam_unix(dovecot:auth): authentication failure;
logname= uid=0 euid=0 tty=dovecot ruser=daryl
rhost=127.0.0.1  user=daryl
```

Decision: sufficient. Reasoning: The context provides a specific data point indicating suspicious activity related to user Daryl, namely an authentication failure for user Daryl on Feb 29 at 15:38:42. This is relevant to identifying suspicious activity, which is part of the original question.

```
MATCH (u:User)-[r]->(n:System)
WHERE toLower(u.id) = 'daryl'
RETURN u.id AS userId, type(r) AS
activityType, n.id AS systemId
```

```
[{'userId': 'daryl', 'activityType':
'AUTHENTICATION_FAILURE_ON',
'systemId': 'mail-0'}, {'userId':
'daryl', 'activityType':
'AUTHENTICATED_ON', 'systemId': 'mail-
0'}, {'userId': 'daryl',
'activityType':
'AUTHENTICATION_FAILURE', 'systemId':
'mail-0'}]
```

'What attack patterns are associated with repeated authentication failures followed by a successful login, and what are the recommended mitigations for such scenarios?'

```
technique: credential-stuffing | label:
Credential Stuffing...
```

```
mitigationLabel: Account Use Policies
mitigationLabel: Password Policies..
```

The attack patterns associated with repeated authentication failures followed by a successful login are primarily related to "Credential Stuffing" and "Password Guessing.", The following mitigations are recommended: 1. **Account Use Policies** 2. **Multi-factor Authentication** 3. **Password Policies** 4. **User Account Management** 5. **Update Software**

User Daryl's activity on the system 'mail-0' shows a pattern of repeated authentication failures followed by a successful login. This behavior is indicative of potential Credential Stuffing or Password Guessing attacks. To mitigate such threats, it is recommended to implement account use policies that limit failed login attempts, enforce multi-factor authentication, establish strong password policies..

Figure 2: Agentic Graph RAG Retrieval Mechanism

The *Reflection Agent* then evaluates the retrieved context (log data) from the *Vector Agent* (2) and determines whether the vector search results provide sufficient information. In this case, the logs reveal repeated authentication failures for user *Daryl* on February 29 at 15:38:42, a potential indicator of malicious activity.

Next, the *Cypher agent* generates a Cypher query by means of an LLM to extract structured, security-relevant relationships from the log graph (3). The query searches for nodes representing the user and connected systems, retrieving activity types and system identifiers. This results in a structured JSON output (5) that shows patterns of both authentication failures and successful logins for *Daryl* on the *mail-0* system.

From this structured data, the *Reflection Agent* determines whether further investigation with cybersecurity knowledge is required, and if so, the agent generates a follow-up question based on the previous findings and the original query (6). In this instance, the generated follow-up question is *"What attack patterns are associated with repeated authentication failures followed by a successful login, and what are the recommended mitigations for such scenarios?"* This follow-up question is then passed to the *SPARQL Agent*, which leverages MCP to identify suitable tools (7) for querying the CSKG. Here, the tool *get_technique_by_keywords* is selected and executed on the MCP server. It performs a SPARQL query via the CSKG SPARQL Endpoint to retrieve relevant attack techniques related to the keyword "*authentication failure*". In this case, the retrieved techniques includes *Credential Stuffing* and *Password Guessing*. The results are evaluated, and the second tool i.e., *get_mitigations_for_technique* is run to query the CSKG to find corresponding mitigations related to the identified attack technique. The SPARQL query returns relevant mitigations such as *Account Use Policies*, *Password Policies*. Since the response is complete and satisfactory, the MCP operation concludes at this point.

Finally, the *Synthesis Agent* merges and summarizes the information with the context from the SPARQL query output (9) and other data gathered from the Vector and Cypher query. The LLM-based synthesis agent produces a final interpretation, compiling all findings into a clear, human-readable security analysis (10).

4. Implementation and Use-Case Application

This section presents the implementation of our approach and its application in the context of security log analysis.

4.1. Implementation

Our prototype implementation is available at https://github.com/sepses/multi-agents-cykg-rag. We implemented the prototype using a Python-based pipeline with LangGraph[2] as the primary multi-agent orchestration framework. This framework coordinates a dynamic workflow that includes query validation and routing, query rewriting and refinement through reflection loops, tool invocation, and cross-agent coordination. The backend architecture employs a dual-graph approach: *(i)* LPGs are used to represent log graphs, stored in a Neo4j database[3]; *(ii)* RDF graphs are used to represent cybersecurity knowledge, stored it in a triple store (specifically, Virtuoso[4] in our prototypical implementation) and provide it via a SPARQL endpoint. For this experiment, we used log data derived from the AIT Log Dataset [27] and employed Neo4J Graph Builder[5]—an LLM-based graph construction framework—to automatically transform raw log data into Neo4J graphs. For cybersecurity knowledge, we leveraged our previous work, the SEPSES CSKG [28], a continuously updated cybersecurity knowledge graph that integrate various cybersecurity information and threat intelligence such as CVE, CWE, CPE, CVSS, CAPEC and ATT&CK. The CSKG is publicly accessible via the SPARQL endpoint[6]. Information retrieval over the log graphs uses a hybrid search mechanism that combines Cypher graph query and semantic vector search within the Neo4j database. This vector search uses Hugging Face[7] embeddings (all-MiniLM-L6-v2) selected for compatibility with the 384-dimensional vector space produced by the Neo4j LLM Graph Builder. Communication between agents and the CSKG is facilitated via MCP[8], a client–server architecture in which LLM-based agents (MCP clients) invoke tools exposed by independent backend services (MCP server) to execute SPARQL queries. For this experiment, we implemented several predefined SPARQL queries in functions and exposed them via MCP, which allows the agents to retrieve MITRE ATT&CK knowledge from the CSKG. Listing 1 shows an example SPARQL Query used in the MCP function "get_techniques_by_keyword" from the CSKG.

Listing 1: SPARQL Query for the MCP function `get_techniques_by_keyword` from the CSKG

```
PREFIX attack: <http://w3id.org/sepses/vocab/ref/attack#>
PREFIX dcterm: <http://purl.org/dc/terms/>

SELECT ?technique ?label WHERE {{
    ?technique a attack:Technique .
    ?technique dcterm:title ?label .
    OPTIONAL {{ ?technique dcterm:description ?description }}
    FILTER( CONTAINS(LCASE(?label), LCASE("{keyword}")) ||
            CONTAINS(LCASE(?description), LCASE("{keyword}")))
}}  ORDER BY ?label
    LIMIT 50
```

[2] https://www.langchain.com/langgraph
[3] https://neo4j.com/
[4] https://vos.openlinksw.com/owiki/wiki/VOS/VOSDownload
[5] https://llm-graph-builder.neo4jlabs.com/
[6] https://w3id.org/sepses/sparql
[7] https://huggingface.co/
[8] https://modelcontextprotocol.io/docs/getting-started/intro

4.2. Use Case Application

We evaluate our approach by means of three distinct use cases. The first two use cases were adopted from previous work [7]. The question answering process of each use case is divided into four parts represented by colored boxes: *(i)* The black box contains the original question – a natural language query posed by a security analyst. *(ii)* The green box shows agents involved in the workflow (i.e., Vector, Cypher, or SPARQL agent). The specific agent used depends on the variety and complexity of the question. *(iii)* The red box contains the output generated by each agent, which serves as contextual information fed to the LLM. *(iv)* The blue box contains the final answer – a synthesized response produced by the LLM, using context derived from all agent outputs.

(i) General Cybersecurity Question In this demonstration, we show how our agentic graph-RAG system can be used to answer direct questions related to cyber threat intelligence – in this case, questions related to MITRE ATT&CK.

Question

Which MITRE ATT&CK techniques are used by attackers to escalate their privileges within a network?

Agent Executions

Agents	Use	Query/Function and Parameter	Iteration (n)
Vector	✗	-	-
Cypher	✗	-	-
SPARQL (MCP)	✓	get_techniques_by_tactic ('tactic_name': 'Privilege Escalation')	1

Agent Output/Context

Vector Output: -

Cypher Output: -

SPARQL Output:

```
technique: abuse-elevation-control-mechanism | techniqueLabel: Abuse Elevation Control Mechanism..
technique: access-token-manipulation | techniqueLabel: Access Token Manipulation..
technique: accessibility-features | techniqueLabel: Access Token Manipulation.. ...
```

Final Answer

Attackers use a variety of techniques to escalate their privileges within a network, as outlined in the MITRE ATT&CK framework. These techniques include manipulating access tokens, exploiting accessibility features, modifying accounts, and bypassing user account controls, among others.

(ii) Security Log Analysis without CSKG Context This scenario demonstrates the use of our agentic graph-RAG system to query information from private or local information (e.g., log sources).

Question

Who is Daryl? What did he do on the system? What time he did it?

Agent Executions

Agents	Use	Query/Function and Parameter	Iteration (n)
Vector	✓	Extracted Entities: [’Daryl’, ’system’]	1
Cypher	✓	"MATCH (u:User)-[r]->(n) WHERE toLower(u.id) = ’daryl’ RETURN u.id AS user, type(r) AS action, n.id AS entity, n.createdAt AS actionTime"	1
SPARQL (MCP)	✗	-	-

Agent Output/Context

Vector Output:

```
Feb 29 08:17:02 mail-0 CRON[4087]: pam_unix(#Resource uid=0 tty=dovecot ruser=daryl rhost=127.0.0.1 user=daryl
Feb 29 15:38:42 mail-0 auth: pam_unix(dovecot:auth): authentication failure; logname= uid=0 euid=0 tty=dovecot
ruser=daryl rhost=127.0.0.1 user=daryl
Feb 29 15:39:01 mail-0 CRON[15907]: pam_unix(cron:session): session opened for user root by (uid=0) ...
```

Cypher Output:

```
[{’user’: ’daryl’, ’action’: ’AUTHENTICATION_FAILURE_ON’, ’entity’: ’mail-0’, ’actionTime’: None}, {’user’: ’daryl’, ’action’:
’AUTHENTICATED_ON’, ’entity’: ’mail-0’, ’actionTime’: None}, {’user’: ’daryl’, ’action’: ’AUTHENTICATION_FAILURE’,
’entity’: ’mail-0’, ’actionTime’: None}] ...
```

SPARQL Output: -

Final Answer

Daryl is a user who attempted to authenticate on the system ’mail-0’. According to the logs, Daryl experienced an authentication failure at 15:38:42 on February 29, as recorded in the ’auth.txt’ document. Subsequently, there was another authentication failure recorded at 15:39:01 on the same day. The logs do not provide a successful authentication timestamp for Daryl, indicating that his attempts to access the system may not have been successful.

(iii) Security Log Analysis with CSKG Context Similar to use-case (ii) above, this scenario demonstrates the use of the framework to facilitate question answering against private or local information (e.g., log sources) as well as to contextualize them to cybersecurity knowledge. In this case, the identified suspicious activities are associated with attack patterns (technique and mitigation) from MITRE ATT&CK.

Question

Identify suspicious activity performed by user Daryl in the system, link the possible threat to attack pattern and the mitigation!

Agent Executions

Agents	Use	Query/Function and Parameter	Interation (n)
Vector	✓	Extracted Entities: [’Daryl’, ’system’]	1
Cypher	✓	"MATCH (u:User)-[r]->(n:System) WHERE toLower(u.id) = ’daryl’ RETURN u.id AS userId, type(r) AS activityType, n.id AS systemId"	1
SPARQL (MCP)	✓	- get_techniques_by_keyword(’keyword’: ’authentication failure’) - get_mitigations_for_technique(technique_name’: ’Credential Stuffing’) ...	2

Agent Output/Context

Vector Output:

```
Feb 29 08:17:02 mail-0 CRON[4087]: pam_unix(#Resource uid=0 tty=dovecot ruser=daryl rhost=127.0.0.1 user=daryl
Feb 29 15:38:42 mail-0 auth: pam_unix(dovecot:auth): authentication failure; logname= uid=0 euid=0 tty=dovecot ruser=daryl rhost=127.0.0.1 user=daryl
Feb 29 15:39:01 mail-0 CRON[15907]: pam_unix(cron:session): session opened for user root by (uid=0) ...
```

Cypher Output:

```
[{'userId': 'daryl', 'activityType': 'AUTHENTICATION_FAILURE_ON', 'systemId': 'mail-0'}, {'userId': 'daryl', 'activityType': 'AUTHENTICATED_ON', 'systemId': 'mail-0'}, {'userId': 'daryl', 'activityType': 'AUTHENTICATION_FAILURE', 'systemId': 'mail-0'}] ...
```

SPARQL Output:

```
- technique: credential-stuffing | label: Credential Stuffing ...
- technique: password-guessing | label: Password Guessing...
- technique: credential-stuffing | mitigationLabel: Account Use Policies..
- technique: credential-stuffing | mitigationLabel: Multi-factor Authentication ..
- technique: password-guessing | MitigationLabel: Password Policies ..
- technique: password-guessing | MitigationLabel: Update Software .. ...
```

Final Answer

User Daryl's activity on the system 'mail-0' shows a pattern of repeated authentication failures followed by a successful login. This behavior is indicative of potential Credential Stuffing or Password Guessing attacks, where adversaries attempt to gain unauthorized access by trying multiple credentials. To mitigate such threats, it is recommended to implement account use policies that limit failed login attempts, enforce multi-factor authentication, establish strong password policies, regularly review user accounts, and keep software updated to protect against vulnerabilities. These measures can help prevent unauthorized access and enhance the security of user accounts.

5. Discussion

The three scenarios we investigated demonstrate the ability and flexibility of the agentic framework to compose and orchestrate workflows for cybersecurity question answering and analysis. We found that the generated agentic workflows successfully retrieve and integrate (i) information stored in labeled property graphs including information retrieved through vector embeddings, with (ii) structured knowledge stored in a triple store and synthesize it with (iii) LLM-internal knowledge to produce a coherent diagnosis.

The first scenario illustrates how the agentic framework enables LLM-based access to structured cybersecurity knowledge. Whereas the ability to explore the knowledge in the graph is constrained by the implemented MCP methods, this approach lowers the adoption barrier for using a CSKG. Comparing the results of the analytic scenarios (ii) and (iii), we further observe that granting the LLM access to structured knowledge about particular attack techniques and mitigations improves the answer from a plain description of the suspicious behavior to explanations and actionable recommendations. The structured knowledge on attack patterns and techniques may also provide a framework to guide more complex multi-step analytic workflows in future work.

In terms of limitations, although the RAG-based approach and the addition of a reflection agent successfully eliminated hallucinations in our experiments, potential risk such as hallucination in the analyses remains. Hence, a human analyst is still required to validate the results. However, the agentic framework can complement analyst workflows and lower their cognitive load, allowing them to focus on higher-level aspects.

Analytic capabilities are constrained by the reliance on predefined Cypher and SPARQL queries exposed via MCP, which currently prevent LLM agents from executing arbitrary queries on the two graph data sources. While this restriction helps manage complexity and guide the analytic process—significantly improving the quality of answers, it also limits the ability to freely explore available knowledge. In future work, we plan to investigate agents with extended access that can directly query the knowledge graph.

Finally, our results on relatively straightforward tasks are highly encouraging. We plan to expand beyond isolated analytic tasks—such as detecting suspicious login attempts—to more complex scenarios. This will allow us to evaluate how agentic architectures reason over extended attack chains and to assess the scalability of such analyses. Finally, attackers might exploit inherent vulnerabilities of the agentic framework and manipulate it to their advantage. We leave adversarial attacks to future work.

6. Conclusions

In this paper, we presented AgCyRAG, a hybrid Agentic Retrieval-Augmented Generation framework that integrates vector-based retrieval with knowledge graph based querying mechanism to enhance the accuracy, contextual relevance, and interpretability of security log analyses. We demonstrated use cases that highlight AgCyRAG's ability to orchestrate multiple retrieval mechanisms in a dynamic, agent-driven workflow. By combining LPG-based log graph queries via Cypher with semantic vector retrieval and RDF-based CSKGs via SPARQL, the framework bridges the gap between low-level security data and high-level cybersecurity knowledge and threat intelligence. The framework automatically refines queries based on intermediate findings, enabling context-aware mapping of suspicious activities to corresponding attack patterns from the CSKG. This heterogeneous data integration through specialized agents not only enhances accuracy and interpretability but also produces evidence-grounded, human-readable analyses that directly support security analysts in operational decision-making. For future work, we aim to conduct comprehensive evaluations of the framework using a variety of LLM models (including local and specialized models), datasets, and use cases to assess its robustness, scalability, and adaptability under evolving threat landscapes. In addition we want to test the framework in complex scenarios, including multi-hop question answering. While this paper focuses on security log analysis, we also plan to explore how the architecture can be applied to other domains such as vulnerability scanning and intrusion detection. By combining symbolic and sub-symbolic retrieval within agentic orchestration, AgCyRAG takes an important step toward more reliable, interpretable, and operationally effective AI-assisted cybersecurity analysis.

Acknowledgements. This work was supported by the Department of Computer Science and Electronics, Universitas Gadjah Mada under the Publication Funding Year 2025. This work was supported by the Austrian Research Promotion Agency (FFG) under the "Digitale Technologien 2023" programme (FFG Project No. FO999915293 – LLM4CTI). SBA Research (SBA-K1 NGC) is a COMET Center within the COMET – Competence Centers for Excellent Technologies Programme and funded by BMIMI, BMWET, and the federal state of Vienna. The COMET Programme is managed by FFG. This research was also funded in part by the Austrian Science Fund (FWF) 10.55776/COE12.

Declaration on Generative AI

During the preparation of this work, the author(s) used ChatGPT in order to: Grammar and spelling check. After using this tool, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] A. Ekelhart, E. Kiesling, K. Kurniawan, et al., Taming the logs-vocabularies for semantic security analysis., in: SEMANTiCS, 2018, pp. 109–119.

[2] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, Y. Zhang, A Survey on Large Language Model (LLM) Security and Privacy: The Good, The Bad, and The Ugly, High-Confidence Computing 4 (2024) 100211. URL: https://www.sciencedirect.com/science/article/pii/S266729522400014X. doi:10.1016/j.hcc.2024.100211.

[3] S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain, J. Gao, Large language models: A survey, 2024. URL: https://arxiv.org/abs/2402.06196. arXiv:2402.06196.

[4] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, T. Liu, A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions, 2023. URL: https://arxiv.org/abs/2311.05232. arXiv:2311.05232.

[5] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, D. Kiela, Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks, 2021. URL: http://arxiv.org/abs/2005.11401, arXiv:2005.11401 [cs].

[6] E. Kiesling, A. Ekelhart, K. Kurniawan, F. Ekaputra, The SEPSES Knowledge Graph: An Integrated Resource for Cybersecurity, in: C. Ghidini, O. Hartig, M. Maleshkova, V. Svátek, I. Cruz, A. Hogan, J. Song, M. Lefrançois, F. Gandon (Eds.), The Semantic Web – ISWC 2019, volume 11779, Springer International Publishing, Cham, 2019, pp. 198–214. doi:10.1007/978-3-030-30796-7_13, series Title: Lecture Notes in Computer Science.

[7] K. Kurniawan, E. Kiesling, A. Ekelhart, Cykg-rag: Towards knowledge-graph enhanced retrieval augmented generation for cybersecurity, in: RAGE-KG 2024 Workshop at ISWC 2024, 2024.

[8] V. Raskin, C. Hempelmann, K. Triezenberg, S. Nirenburg, Ontology in information security: A useful theoretical foundation and methodological tool, in: Proceedings of the 2001 Workshop on New Security Paradigms, 2001. doi:10.1145/508171.508180.

[9] M. Schumacher, Toward a Security Core Ontology, 2003. doi:10.1007/978-3-540-45180-8_6.

[10] J. Undercoffer, A. Joshi, J. Pinkston, Modeling computer attacks: An ontology for intrusion detection, in: Recent Advances in Intrusion Detection, 2003.

[11] A. Kim, J. Luo, M. Kang, Security ontology for annotating resources, in: On the Move to Meaningful Internet Systems 2005: CoopIS, DOA, and ODBASE, 2005.

[12] A. Martimiano, E. d. S. Moreira, An owl-based security incident ontology, in: Proceedings of the Eighth International Protege Conference, 2005.

[13] A. Souag, C. Salinesi, I. Comyn-Wattiau, Ontologies for security requirements: A literature survey and classification, in: Advanced Information Systems Engineering Workshops, 2012.

[14] M. Iannacone, S. Bohn, G. Nakamura, J. Gerth, K. Huffer, R. Bridges, E. Ferragut, J. Goodall, Developing an ontology for cyber security knowledge graphs, 2015. doi:10.1145/2746266.2746278.

[15] K. Kurniawan, E. Kiesling, D. Winkler, A. Ekelhart, The ics-sec kg: An integrated cybersecurity resource for industrial control systems, in: International Semantic Web Conference, Springer, 2024, pp. 153–170.

[16] K. Kurniawan, A. Ekelhart, E. Kiesling, An att&ck-kg for linking cybersecurity attacks to adversary tactics and techniques, in: The Semantic Web – ISWC 2021, 2021, p. 5. URL: https://ceur-ws.org/Vol-2980/paper363.pdf.

[17] K. Liu, F. Wang, Z. Ding, S. Liang, Z. Yu, Y. Zhou, Recent Progress of Using Knowledge Graph for Cybersecurity, Electronics 11 (2022) 2287. doi:10.3390/electronics11152287.

[18] K. Liu, F. Wang, Z. Ding, S. Liang, Z. Yu, Y. Zhou, A review of knowledge graph application scenarios in cyber security, 2022. URL: http://arxiv.org/abs/2204.04769, arXiv:2204.04769 [cs].

[19] L. F. Sikos, Cybersecurity knowledge graphs, Knowledge and Information Systems 65 (2023) 3511–3531. doi:10.1007/s10115-023-01860-3.

[20] X. Zhao, R. Jiang, Y. Han, A. Li, Z. Peng, A survey on cybersecurity knowledge graph construction, Computers & Security 136 (2024) 103524. doi:10.1016/j.cose.2023.103524.

[21] K. Huang, Y. Wang, B. Goertzel, Y. Li, S. Wright, J. Ponnapalli (Eds.), Generative AI Security: Theories and Practices, Future of Business and Finance, Springer Nature Switzerland, Cham, 2024. URL: https://link.springer.com/10.1007/978-3-031-54252-7. doi:10.1007/978-3-031-54252-7.

[22] A. Singh, A. Ehtesham, S. Kumar, T. T. Khoei, Agentic Retrieval-Augmented Generation: A Survey on Agentic RAG, 2025. URL: http://arxiv.org/abs/2501.09136. doi:10.48550/arXiv.2501.09136, arXiv:2501.09136 [cs].

[23] N. Kshetri, Transforming Cybersecurity with Agentic Ai to Combat Emerging Cyber Threats, 2025. URL: https://www.ssrn.com/abstract=5159598. doi:10.2139/ssrn.5159598.

[24] M. Lupinacci, F. Blefari, F. Romeo, F. A. Pironti, A. Furfaro, ARCeR: an Agentic RAG for the Automated Definition of Cyber Ranges, 2025. URL: http://arxiv.org/abs/2504.12143. doi:10.48550/arXiv.2504.12143, arXiv:2504.12143 [cs].

[25] M. Shao, H. Xi, N. Rani, M. Udeshi, V. S. C. Putrevu, K. Milner, B. Dolan-Gavitt, S. K. Shukla, P. Krishnamurthy, F. Khorrami, R. Karri, M. Shafique, CRAKEN: Cybersecurity LLM Agent with Knowledge-Based Execution, 2025. URL: http://arxiv.org/abs/2505.17107. doi:10.48550/arXiv.2505.17107, arXiv:2505.17107 [cs].

[26] C. Paduraru, C. Patilea, A. Stefanescu, CyberGuardian 2: Integrating LLMs and Agentic AI Assistants for Securing Distributed Networks:, in: Proceedings of the 20th International Conference on Evaluation of Novel Approaches to Software Engineering, SCITEPRESS - Science and Technology Publications, Porto, Portugal, 2025, pp. 660–667. URL: https://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0013406000003928. doi:10.5220/0013406000003928.

[27] M. Landauer, F. Skopik, M. Wurzenberger, W. Hotwagner, A. Rauber, Have it your way: Generating customized log datasets with a model-driven simulation testbed, IEEE Transactions on Reliability 70 (2021) 402–415. doi:10.1109/TR.2020.3031317.

[28] E. Kiesling, A. Ekelhart, K. Kurniawan, F. Ekaputra, The sepses knowledge graph: An integrated resource for cybersecurity, in: C. Ghidini, O. Hartig, M. Maleshkova, V. Svátek, I. Cruz, A. Hogan, J. Song, M. Lefrançois, F. Gandon (Eds.), The Semantic Web – ISWC 2019, Springer International Publishing, Cham, 2019, pp. 198–214. URL: https://doi.org/10.1007/978-3-030-30796-7_13.