Re-implementing Agentic Graph RAG for Vulnerability Assessment

Implementation Plan: November 6 - December 4, 2025

Team 7

PIC:
Muhamad Hafiz Saputra (23/518511/PA/22252)

Members:
Syafran Abdillah Erdin (23/521752/PA/22444)
Dzikran Azka Sajidan (23/516665/PA/22110)
Tegar Prasetyo (23/520277/PA/22364)
Vincentius Davin Febrillianagata (23/520016/PA/22330)

1 Project Overview

This project aims to adapt and re-implement the Agentic Graph RAG approach for vulnerability assessment, integrating cybersecurity knowledge graphs to rank vulnerabilities, generate risk assessments, and provide mitigation advice.

1.1 Key Deliverables

- Agentic RAG system for vulnerability assessment
- Comparison with Normal RAG and Traditional CVSS scoring
- Evaluation results (accuracy, ranking quality, speed)
- 3 real vulnerability examples with system outputs
- GitHub repository with code and documentation

1.2 Technical Stack

- **Knowledge Graph:** SEPSES Cybersecurity KG (CVE, CVSS, CWE, CAPEC)
- **Framework:** LangGraph for agent orchestration
- **Database:** Neo4j (LPG) + Virtuoso/SPARQL endpoint
- **LLM:** OpenAI GPT-4 or Claude (for prototyping)
- **Embedding:** all-MiniLM-L6-v2 (HuggingFace)

2 Weekly Implementation Plan

Week 1: November 6-12, 2025 - Setup & Research

Team Member	Tasks	Deliverables
Hafiz (PIC)	• Study both papers thoroughly • Setup project GitHub repository • Define evaluation metrics (accuracy, ranking correlation, response time) • Create project timeline and coordinate team meetings	GitHub repo, Evaluation plan
Dzikran	• Study AgCyRAG architecture (Paper 1) • Explore SEPSES KG SPARQL endpoint • Document available KG queries and data schema • Setup Python environment (LangGraph, Neo4j)	KG documentation, Environment setup
Tegar	• Collect CVE vulnerability dataset (NVD API) • Select 50-100 vulnerabilities with varying severity • Extract 3 detailed case studies for demonstration • Prepare ground truth CVSS scores	Dataset (CSV/JSON), Case studies
Syafran	• Setup Neo4j database locally • Install and configure required dependencies • Test SPARQL queries against SEPSES endpoint • Document API keys and access requirements	Neo4j setup, SPARQL test queries
Davin	• Research embedding models for vulnerability text • Setup vector database (Neo4j vector index) • Test embedding generation pipeline • Prepare baseline RAG implementation outline	Embedding pipeline, RAG baseline design

Week 2: November 13-19, 2025 - Core Implementation

Team Member	Tasks	Deliverables
Hafiz (PIC)	• Implement Guardrail Agent (query validation) • Design agent orchestration workflow • Coordinate integration between team components • Review and merge code from team members	Guardrail Agent, Integration plan

Dzikran	• Implement SPARQL Agent for KG queries • Create MCP functions for SEPSES KG access • Implement vulnerability-to-CAPEC mapping queries • Test retrieval accuracy from KG	SPARQL Agent code, Query functions
Tegar	• Implement Vector Agent (semantic search) • Create vulnerability embeddings from descriptions • Build vector similarity search module • Integrate with Neo4j vector index	Vector Agent, Embedding index
Syafran	• Implement Cypher Agent for Neo4j queries • Model vulnerability relationships in Neo4j • Create graph traversal queries for impact analysis • Test query performance	Cypher Agent, Neo4j graph model
Davin	• Implement Normal RAG baseline (no KG) • Use only vector retrieval over CVE descriptions • Implement Traditional CVSS baseline • Prepare comparison framework	Baseline implementations

Week 3: November 20-26, 2025 - Integration & Enhancement

Team Member	**Tasks**	**Deliverables**
Hafiz (PIC)	• Implement Reflection Agent (quality control) • Implement Synthesis Agent (response generation) • Integrate all agents into LangGraph workflow • End-to-end testing of agentic system	Complete Agentic RAG system
Dzikran	• Implement vulnerability ranking algorithm • Combine CVSS, exploitability, and KG context • Add severity scoring logic • Test ranking against ground truth	Ranking module, Test results

Tegar	• Implement risk assessment generation • Create prompt templates for summaries • Generate mitigation advice from CAPEC/CWE • Test output quality on case studies	Risk assessment module
Syafran	• Optimize query performance (caching, indexing) • Implement parallel agent execution • Profile system response times • Optimize database queries	Performance optimizations
Davin	• Run comparative experiments (Agentic vs Normal RAG vs CVSS) • Collect accuracy metrics (precision, recall) • Measure ranking correlation (Spearman's rho) • Record response times for each approach	Experimental results

Week 4: November 27 - December 3, 2025 - Evaluation & Documentation

Team Member	Tasks	Deliverables
Hafiz (PIC)	• Write comprehensive README documentation • Prepare final presentation slides • Review all deliverables for completeness • Coordinate final testing and bug fixes	Documentation, Presentation
Dzikran	• Generate detailed outputs for 3 case studies • Create comparison tables/charts (accuracy, ranking, speed) • Analyze system strengths and limitations • Write technical report sections	Case study outputs, Analysis report
Tegar	• Create evaluation visualizations (charts, graphs) • Generate confusion matrices and ranking plots • Prepare demo video/screenshots • Write results section for report	Visualizations, Demo materials

Syafran	• Clean and organize codebase • Write API documentation • Create setup/installation guide • Prepare code comments and docstrings	Clean code, API docs
Davin	• Prepare dataset files for GitHub • Create reproducibility instructions • Test installation on fresh environment • Upload final code and data to GitHub	GitHub repository (final)

Final Deliverable: December 4, 2025

- **GitHub Repository:** Complete code, data, and documentation
- **Technical Report:** Methodology, evaluation results, analysis
- **Presentation:** Demo and findings
- **Comparison Table:** Agentic RAG vs Normal RAG vs CVSS
- **3 Case Studies:** Real vulnerability assessments with system outputs

3 Evaluation Metrics

3.1 Quantitative Metrics

- **Accuracy:** Precision, Recall, F1-score for vulnerability detection
- **Ranking Quality:** Spearman's correlation with ground truth CVSS
- **Speed:** Average response time per query (seconds)
- **Coverage:** % of vulnerabilities correctly mapped to attack patterns

3.2 Qualitative Assessment

- Clarity and actionability of risk assessments
- Relevance of mitigation recommendations
- Explanation quality and grounding in KG evidence

4 Risk Mitigation

Risk	Impact	Mitigation
SPARQL endpoint downtime	Cannot access SEPSES KG	Cache KG data locally, use backup endpoints
LLM API rate limits	Slow experimentation	Use local models (Llama, Mistral) as backup
Integration issues	Delays in testing	Weekly integration checkpoints, modular design
Insufficient dataset	Poor evaluation	Collect diverse CVEs early, validate with experts

5 Communication Plan

- **Weekly Meetings:** Schedule TBD (coordinate during Week 1)
- **Code Reviews:** All PRs require 1 review before merging
- **Async Updates:** WhatsApp group for quick coordination
- **Emergency Contact:** Hafiz (PIC) coordinates urgent issues

6 References

1. Kurniawan et al., "AgCyRAG: an Agentic Knowledge Graph based RAG Framework for Automated Security Analysis", RAGE-KG 2025
2. Kiesling et al., "The SEPSES Knowledge Graph: An Integrated Resource for Cybersecurity", ISWC 2019
3. SEPSES SPARQL Endpoint: `https://w3id.org/sepses/sparql`
4. AgCyRAG GitHub: `https://github.com/sepses/multi-agents-cykg-rag`