**Journal of Computer and Electrical Sciences (JCES)- Volume-1 Issue-4, March 2026**
**ISSN: 3049-2602**

# NETWORK LOG ANALYSIS SYSTEM USING RETRIEVAL AUGMENTED GENERATION AND LARGE LANGUAGE MODELS

**K. Madhurya[1],**

**Mr. M. Samuel Sandeep[2]**

[1]Student, Department of Computer Science & Engineering

Andhra Loyola Institute of Engineering and Technology, Vijayawada, Andhra Pradesh, India
[2]*Assistant Professor, Department of Computer Science & Engineering*
*Andhra Loyola Institute of Engineering and Technology*
*Vijayawada, Andhra Pradesh, India*
*Email id:* madhuryakondaveeti@gmail.com

**Abstract:** The increasing reliance on networked systems has led to a surge in cyber threats, producing large volumes of network logs that require continuous analysis. Manual log analysis is time-consuming and complex, while traditional methods struggle with unstructured data. To address this, the project proposes an intelligent Network Log Analysis system using Retrieval-Augmented Generation (RAG) and Large Language Models (LLMs).The system preprocesses log data, converts it into vector embeddings, and stores them in a FAISS vector database for efficient semantic retrieval. When a user submits a query, the RAG pipeline retrieves relevant log context and uses an LLM to generate accurate, context-aware responses.This approach enables interactive log analysis, detection of suspicious activities, and automated generation of security reports. By leveraging RAG and LLMs, the system enhances cybersecurity monitoring, reduces analyst workload, and improves the efficiency of threat investigation.

**Keywords:** Network Log Analysis, Retrieval-Augmented Generation (RAG), Large Language Models (LLMs), Cybersecurity, FAISS Vector Database, Log Investigation, Threat Detection, Security Analytics

## 1. INTRODUCTION

With the rapid growth of digital networks, cyber attacks have become more frequent and complex. Modern systems generate large volumes of log data from sources like firewalls, servers, and intrusion detection systems. These logs are crucial for identifying suspicious activities and maintaining network security. However, manual log analysis is time-consuming and requires expertise, while traditional rule-based tools struggle with complex patterns and unstructured data.Recent advancements in artificial intelligence and natural language processing have introduced Large Language Models (LLMs), which can understand and generate meaningful insights from complex data. Retrieval-Augmented Generation (RAG) enhances this capability by combining retrieval mechanisms with language models to produce more accurate, context-based responses.The proposed system, "Network Log Analysis using Retrieval-Augmented Generation and Large Language Models," processes log data into vector embeddings stored in a FAISS database for efficient retrieval. When users submit queries, the system retrieves relevant log information and uses an LLM to generate accurate responses.The system also provides an interactive interface for analyzing logs, detecting potential threats, visualizing attack timelines, and generating automated security reports. This approach improves the efficiency of log analysis and supports faster, more effective cybersecurity monitoring.

### 2. Literature Survey

The growth of network systems and internet services has increased cybersecurity threats, generating large volumes of log data from devices like firewalls, servers, and intrusion detection systems. Analyzing these logs is essential for detecting threats, but it is challenging due to their size and complexity. Researchers have explored machine learning and advanced language models to improve log analysis. Machine learning approaches use algorithms such as Decision Trees and Support Vector Machines to detect anomalies in network logs. These methods help identify suspicious activities but often lack deep contextual understanding.Recent research focuses on Large Language Models (LLMs) and Retrieval-Augmented Generation (RAG), which combine retrieval with language generation to produce context-aware and accurate responses. This improves the reliability of automated log analysis.The proposed system uses RAG and LLMs to analyze network logs by converting them into vector embeddings and storing them in a FAISS database for efficient retrieval and response generation through a web interface.From the reviewed literature, the following key observations can be made:

- Machine learning techniques can effectively detect anomalies in network logs but lack contextual understanding.
- Retrieval-Augmented Generation improves accuracy by combining retrieval with language models.
- Large Language Models enable better interpretation of complex log data.
- Efficient vector databases improve storage and retrieval performance.
- System performance depends on data quality and retrieval efficiency.

## 3. Proposed System

To overcome the limitations of existing systems, this project proposes a Network Log Analysis system using Retrieval-Augmented Generation (RAG) and Large Language Models (LLMs). The system automates the analysis of network logs and assists security analysts in identifying potential threats more efficiently.In the proposed system, network log files are first preprocessed and divided into smaller text segments. These segments are converted into vector embeddings using an embedding model and stored in a FAISS vector database. When a user submits a query, the system retrieves relevant log entries from the database and passes them to a large language model for analysis. The model then generates meaningful responses based on the retrieved context.The system is integrated into a Flask-based web application that allows users to upload log files, perform chat-based log analysis, visualize attack timelines using graphical representations, and generate automated security reports. This approach improves the efficiency of log investigation and helps analysts quickly identify suspicious activities within large datasets.

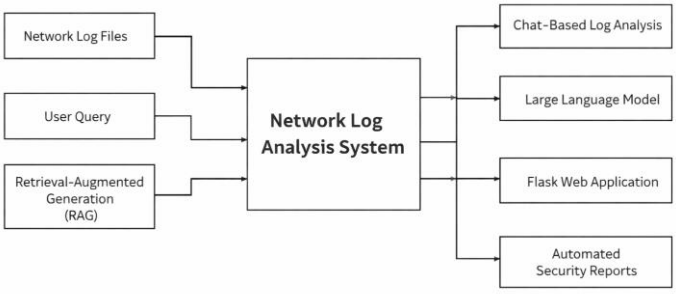


Fig 1: Proposed System

The application provides an interactive interface that allows users to view and control different functionalities such as:

- Automates the analysis of network log files.
- Uses Retrieval-Augmented Generation to improve response accuracy.
- Enables interactive chat-based investigation of log data.
- Provides visualization of attack timelines through graphs.
- Generates automated security analysis reports.
- Improves efficiency and reduces the workload of security analysts.

## 4. Methodology

The methodology of the system is organized into the following steps:

1. Log Data Collection:The system allows users to upload network log files from sources such as firewalls, servers, and intrusion detection systems for analysis.
2. Data Preprocessing:The uploaded log files are cleaned and divided into smaller text segments to make them suitable for further processing.
3. Embedding Generation:Each log segment is converted into vector embeddings using an embedding model, enabling semantic representation of the data.
4. Vector Storage:The generated embeddings are stored in a FAISS vector database, allowing efficient similarity-based retrieval.
5. Query Processing:Users submit queries through a web interface. The system processes the query and converts it into an embedding for comparison.
6. Context Retrieval (RAG):Relevant log segments are retrieved from the vector database based on semantic similarity and passed to the next stage.
7. AI-Based Analysis:A Large Language Model (LLM) analyzes the retrieved data and generates meaningful, context-aware responses to user queries.
8. Visualization and Reporting:The system provides features such as chat-based analysis, attack timeline visualization, and automated security report generation.
9. System Management:The Flask-based web application manages communication between modules, handles user requests, and ensures smooth system operation.

## 5. Proposed System Results

The proposed Network Log Analysis system using Retrieval-Augmented Generation (RAG) and Large Language Models (LLMs) was successfully developed and tested under different scenarios. The system effectively enabled automated log analysis, threat detection, and interactive investigation.

- The system efficiently processed large network log files and converted them into vector embeddings stored in the FAISS database. Retrieval of relevant log data was fast and accurate.

- The user interface allowed users to upload log files and perform chat-based analysis smoothly. The system generated meaningful and context-aware responses to user queries.

- The RAG-based approach improved the accuracy of responses by retrieving relevant log information before analysis. This helped in identifying suspicious activities and potential threats.

- The system successfully visualized attack timelines using graphical representations, making it easier to understand security events.

- Automated security report generation worked effectively, summarizing key findings from the analyzed logs.

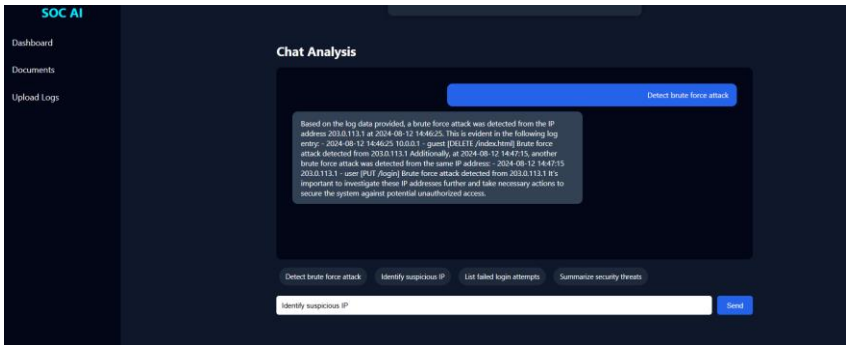


**Fig 2: Chat-Based Log Analysis Interface**

The response time of the system was fast, and query processing with log retrieval and analysis was performed efficiently.

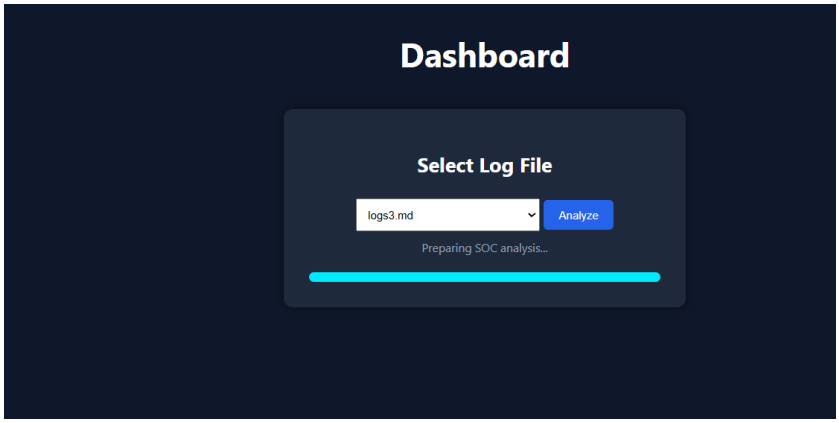


**Fig 3: Log Data Processing**

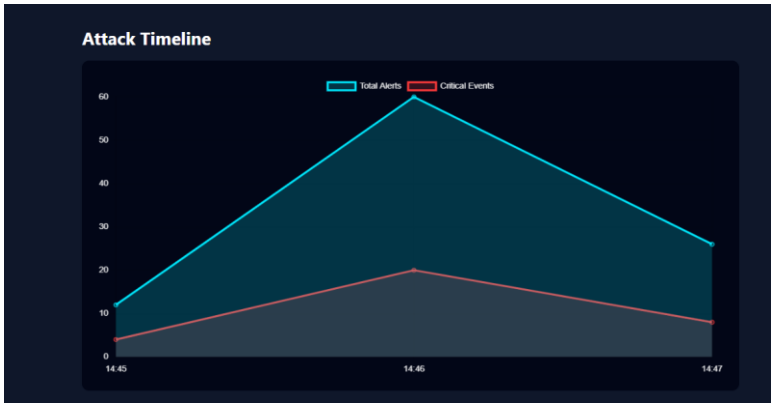


**Fig 4: Attack Timeline Visualization**

Overall, the system demonstrated reliable performance in automated log analysis, interactive investigation, and efficient cybersecurity monitoring.

# 6. CONCLUSION

The Network Log Analysis System using Retrieval-Augmented Generation (RAG) and Large Language Models (LLMs) is designed to assist security analysts in efficiently analyzing large volumes of log data. Traditional methods rely on manual inspection, which is time-consuming and challenging for large datasets. The proposed system automates this process using AI to extract meaningful insights from log files.The application is developed using Python, Flask, HTML, CSS, and JavaScript, providing an interactive interface and efficient backend processing. Log files are converted into vector embeddings using the *nomic-embed-text* model and stored in a FAISS database for semantic retrieval. When a user submits a query, relevant log data is retrieved and analyzed by the Mistral LLM to generate accurate, context-aware responses.

The system supports features such as log file uploading, chat-based analysis, attack timeline visualization, and automated report generation. Testing shows that the system effectively processes logs and generates meaningful insights.Overall, the project demonstrates the practical use of RAG and LLMs in cybersecurity, improving threat detection and enhancing the efficiency of log analysis.

## REFERENCES

[1] P. Lewis, E. Perez, A. Piktus et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," Advances in Neural Information Processing Systems (NeurIPS), 2020.
Available: https://arxiv.org/abs/2005.11401

[2] T. Brown et al., "Language Models are Few-Shot Learners," Advances in Neural Information Processing Systems (NeurIPS), 2020.
Available: https://arxiv.org/abs/2005.14165

[3] J. Johnson, M. Douze, and H. Jégou, "FAISS: A Library for Efficient Similarity Search and Clustering of Dense Vectors," Facebook AI Research, 2017.
Available: https://github.com/facebookresearch/faiss

[4] LangChain Documentation, "LangChain: Building Applications with LLMs, 2023"
Available: https://python.langchain.com

[5] Ollama Documentation, "Running Large Language Models Locally, 2023"
Available: https://ollama.com

[6] Mistral AI, "Mistral 7B: Open Large Language Model, 2023"
Available: https://mistral.ai

[7] Flask Documentation, "Flask Web Framework, 2010"
Available: https://flask.palletsprojects.com

[8] C. Behl and S. Behl, Cybersecurity and Cyberwar: What Everyone Needs to Know, Oxford University Press, 2017.