International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 04 | April - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

MCP-Driven RAG Architecture for Automated Alert Triage in Security Operations Centers

1.Tisha Patel, 2.Payal Kuvar

Computer Engineering Department, Gandhinagar University, Gandhinagar, Gujarat, India.

Computer Engineering Department, Gandhinagar University, Gandhinagar, Gujarat, India.

Abstract

Security Operations Centers (SOCs) are the first line of defense against cyber threats, but they are becoming more and more overwhelmed by the large number and complexity of alerts that modern enterprise infrastructure generates. Surveys of the industry show that false-positive rates in enterprise SIEM deployments often go over 60%. This constant noise makes analysts tired, lowers the quality of decisions, and gives attackers chances to work without being seen. Rule-based correlation engines, supervised machine learning classifiers, and SOAR platforms are some of the current solutions. They only solve part of the problem, and they all have a major flaw: they can't automatically add new threat intelligence, and none of them give clear reasons for triage decisions. This paper presents a unified architecture that integrates the Model Context Protocol (MCP) with a Retrieval-Augmented Generation (RAG) pipeline to facilitate intelligent, explainable, and real-time alert triage in Security Operations Center (SOC) settings. Anthropic released MCP as an open standard in November 2024 [10]. It is a standardized JSON-RPC 2.0 integration layer that links an LLM reasoning engine to CVE databases, the MITRE ATT&CK knowledge base, curated incident repositories, and live threat intelligence feeds. RAG gives the model relevant external context at inference time without having to retrain it. The LLM then makes structured triage decisions and gives analysts step-by-step reasoning chains that they can check, change, and learn from.

A proof-of-concept implementation was evaluated using the CICIDS2017 benchmark dataset [7] through a reproducible simulation pipeline that produced a corpus of 47,312 synthetic SIEM alerts. Important Caveat: All performance metrics reported in this paper are from the proof-of-concept implementation within a simulation pipeline running on 2017-era network traffic and are not production performance claims. The 61.4% simulation-generated baseline FPR was generated within the simulation itself and was not validated against an actual SIEM deployment. All reported improvements over baselines should be understood relative to this simulation-generated FPR only. Within these limits, the proposed system achieved a FPR, TPR, and a mean triage latency of 3.91 ± 0.52 seconds. All gains over three baseline configurations are statistically significant ($p < 0.001$ via two-proportion z-test). The overall score in the explainability assessment from three certified SOC analysts was 4.6/5.0, which is not generalizable outside of this group and should be considered only as an initial indication of explainability until further, larger studies ($n >= 30$) are completed. Component-level latency is discussed based on profiling and attributed to the use ofMCPresponse caching(meancache-hit rate 62.3%)andstructuredcontext prompt compression.The RAG-only baseline did not utilize equivalent caching, thus partially confounding comparison. Contemporary attack dataset evaluation was prioritized as a target for future work.

1. Introduction

Today, the cybersecurity landscape is defined by increasingly high volumes and sophistication of attacks, coupled with the ever-growing attack surface that organizations must defend. Modern IT infrastructure spans hybrid data centers, multi-cloud environments, container orchestration, and diverse endpoint deployments, all generating a continuous stream of security data that demands vigilance. This data is funnelled into the Security Operations Center (SOC), where analysts strive to sift genuine threats from benign noise.

The cost of this massive task is substantial. Practitioner surveys regularly report enterprise SIEM false positive rates exceeding 60 percent [1, 11], while analyst capacity to keep pace with daily alerts is frequently insufficient. This workload contributes to "alert fatigue," a recognized condition where the persistent volume of security warnings leads analysts to become desensitized, thus increasing the likelihood of overlooking or downplaying significant threats. The consequences include longer attacker dwell times, elevated risks of data breaches, and slower incident containment, all carrying hefty financial, regulatory, and reputational implications [12].

Current technical approaches for threat detection broadly fall into three categories: First, rule-based SIEMs apply pre-defined signatures and thresholds to identify malicious patterns; however, these require manual updates whenever new threat techniques emerge. Second, supervised machine learning models offer improved generalization but rely on regular retraining with labeled data, generate opaque classifications lacking transparency, and can fall behind rapidly evolving threat landscapes. Third, SOAR platforms automate response workflows, but these rely on human-defined playbooks unable to proactively identify every possible novel threat pattern.

Large Language Models (LLMs) used with Retrieval-Augmented Generation (RAG) provide a different approach by enabling natural language understanding, dynamic context retrieval, and clear explanations along with predictions. This ability directly addresses the limitations of existing methods. To make LLMs practical for SOC alert triage in a real-world setting, an open, standard method is needed to integrate them with the wide variety of security tools and data sources. The alternative—writing individual connectors for each tool—results in the maintenance overhead, handling inconsistencies, and fragile interfaces.

The Model Context Protocol (MCP) [10], released by Anthropic in November 2024, is designed to meet this need. MCP is a communication protocol based on JSON-RPC 2.0 for secure, bidirectional interaction between LLMs and other tools, which supports extensible and auditable agent systems. For the SOC, MCP acts as the unifying integration layer, connecting SIEM alert streams, CVE databases, threat intelligence feeds, historical incident data, and the MITRE ATT&CK framework with an LLM-powered reasoning engine.

1.1 Contributions

The primary contributions of this paper include:

(1) An MCP-enabled RAG architecture, specialized for automated SOC alert triage, establishing MCP as the standard for SOC tooling-LLM interaction. This work expands upon a previous demonstration of MCP-RAG for compliance (Lodge, 2025 [4]) by specifically addressing alert triage and performing formal validation with statistical significance.

(2) A reproducible alert simulation pipeline that transforms the CICIDS2017 flow-level dataset into 47,312 simulated SIEM alerts for fair comparison against rule-based, LLM-only, and RAG-only baselines.

(3) A transparent triage output format where the LLM provides both a JSON decision and a step-by-step explanation, facilitating analyst review, overrides, and learning.

(4) Component-level latency profiling, which reveals that MCP-RAG outperforms a RAG-only approach due to MCP response caching (62.3% cache-hit rate) and efficient prompt compression (54% reduction).

(5) Honest reporting of eight study limitations, along with proposed mitigations and a candid disclaimer that the reported baseline false positive rate of 61.4% was generated in a simulation environment and not from a live SIEM deployment.

The rest of the paper is organized as follows: Section 2 summarizes related work; Section 3 details the proposed architecture; Section 4 outlines the implementation and experimental setup; Section 5 presents and analyzes the findings; Section 6 discusses limitations; and Section 7 concludes by suggesting directions for future research.

2. Related Work

2.1 LLMs in Cybersecurity and SOC Automation

The field of applying large language models (LLMs) to cybersecurity challenges has moved from demonstrations to operationally useful deployments with speed and remarkable agility. LLM-based architectures using Transformers have demonstrated concrete improvements in areas such as malware classification, identifying phishing URLS, summarising incident reports, and automatically generating descriptions for vulnerabilities. What makes LLMs especially well suited to tasks such as alert triage, in particular, is their ability to reason over text that is both long and semantically rich: "SIEM logs are Semi-structured and require contextual interpretation within a framework of attacker activity, criticality, and patterns in earlier incidents – exactly the kind of inference thatLLMs Excel at" as explained by Ahmed et al. (2025).

We recently completed a comprehensive survey by Ahmed et al. (2025) [1] of LLM adoption in primary Security Operations Center (SOC) activities, namely log summarization, alert triage, and automated incident reporting. Their analysis of 87 published papers concludes that, although LLM-driven solutions outperforms rule-based systems in detection performance and incident response time, several vulnerabilities still affect current approaches: ability to withstand attacks on system prompt engineering, exposure of sensitive data, the capacity to fabricate details of technical data elements (e.g. CVE codes and ATT&CK IDs), and lack of explainability for any final decision that the LLM might make. These limitations provided some insight for this paper's technical solutions, primarily the use of structured output schemata and citing retrieved external context to restrict Hallucinations.

As described in Radosevich & Halloran (2025) [3], a systematic security review of tools that integrated MCP-enabled LLMs also documented several additional threat vectors that aren't necessarily present in more "traditional" API-onlyLLM services. Of the threat classes detailed in this paper, the ones most relevant toSOCdeployments include description manipulation where malicious content injected in a tool's meta-data alters its behavior, and more subtle attacks on a prompt injection through data Retrieved from External Sources — especially when the threat data originates from systems Underattacker compromise. Both tool signing and sandboxed execution in our architecture were motivated by this security analysis.

2.2 Retrieval-Augmented Generation for Security Applications

Retrieval-Augmented Generation (RAG) has been proposed by Lewis et al. (2020) [6] as a way to improve factual accuracy of generative models through augmenting their outputs with external knowledge retrieved at generation time. They combine neural information retrieval with generative models, targeting open domain question answering. Research subsequent to this work has adapted this RAG technique for domains requiring high degree of knowledge freshness in Cybersecurity. In a RAG setup for security use cases, for example, threat intelligence may be gathered through the on-demand retrieval of new CVE information, updated technique information (ATT&CK), or new attacker advisories, without the need to retrain the underlying generative model. Lodge (2025) [4] presented at BSidesLas Vegas what was essentially a SOC/compliance task optimization approach with an MCP–based RAG system. However,their approach did not present quantitative analysis for an alert triage data set or multiple, statistically significant results in comparison to baselines of other solutions. Furthermore,Mansour et al. [5] developed a RAGbased response process that uses "natural-language"similarity searching

combined with standardized security intelligence and "retrieves a context relevant to the alert, increasing response speed and context awareness." without MCP standardizations; their solution therefore faces interoperability challenges.This contribution extends this prior research by quantitatively measuring the efficacy of a purpose builtMCP-nativeRAG system for SOC alert triage utilizing a benchmark dataset and multiple baseline comparisons with appropriate statistical tests.

2.3 Model Context Protocol: Architecture and Security

The Model Context Protocol. The Model Context Protocol, recently announced by Anthropic in Late 2024 [10], defines a protocol for "interconnectingLLMs and LLM-powered agents with third-party tools and data". It establishes the tools, and information thatLLMs interact with, and their prompts as abstractions that are accessed using JSON RPC. This facilitates discovery and invocation of services as well as information collation from many disparate sources within a single model inference cycle — eliminating the need for API integration.

Hou et al. (2025) [2] propose and explore an analysis of the "Model Context Protocol (MCP) and its associated attack ecosystem," categorizing a variety of threat landscapes including: "novel attack surfaces that only appear in MCP-integrated systems, such as tool descriptions and system level configurations." A survey that was performed "by searching the major scholarly information repositories [Google scholar, ACM DL, IEEE Xplore, arXiv] in late March 2026," found only the "work by Hou et al. (2025) that identifies security vulnerabilities and challenges in MCP," while our architecture was designed with reference to their findings in mind, notably their proposed "security mechanism including tool description, signing and sandboxedExecution that is isolated from the actual sensitive data stored in SOCs' data repositories.Table 1 Positions the Model Context Protocol along-sideOther LLM and SIEM solutions for alert Triage and other SOC Use Cases"

Table 1. Comparison of SOC Alert Triage Approaches Across Key Capability Dimensions

Approach	Reasoning Depth	Threat Intel Integration	Explainability	Typical FPR (Literature)
Rule-Based SIEM	Low	None	Static rules only	60–70%
ML Classifier (RF / XGBoost)	Medium	Limited (static features)	Feature importance	30–50%
LLM Only (no RAG)	High	Static training data	Moderate (CoT)	40–50%
RAG-Only Pipeline	High	Dynamic retrieval	Good (cited docs)	25–35%
Proposed MCP-RAG	High	Real-time, multi-source	Excellent (cited + structured)	~21% (this work)

Note. FPR values for rule-based, ML classifier, and LLM-only rows are indicative ranges compiled from the literature [1, 11]. Proposed MCP-RAG values reflect experimental results from this work (Section 5). CI = confidence interval; all CIs computed at 95% using the Wilson score method.

3. Proposed MCP-RAG SOC Architecture

3.1 Architectural Overview

The described architecture has 4 functionally different but tightly coupled layers: (1) The SOC Data Ingestion Layer, which unifies disparate security telemetry into a common alert format; (2) The MCP Integration Layer, which coordinates access to outside world data sources via registered tools; (3) The RAG Processing Layer, which enacts a hybrid retrieval of contextually relevant documents; (4) The LLM Triage and Explanation Layer, which provides a structured triage result and human readable chain of thought.

Alert Processing operates as a sequence; A SIEM alert, IDS alert or endpoint log alert originating from any data source (SIEMs, IDS, endpoints) is ingested and is normalized into a canonical JSON format. The standardized alert schema JSON is sent to the MCP server which orchestrates multiple tool requests simultaneously to ingest data regarding the alert's context. Relevant data regarding CVE details, ATT&CK technique mapping, prior incidents history, ongoing threat indicators are retrieved. This context is formatted and concatenated into a RAG prompt for the LLM agent which issues a standardized structured triage JSON with an explanatory chain of thought returned to the SOC analyst console.

3.2 SOC Data Ingestion Layer

The ingestion layer accepts raw alerts from heterogeneousSOC data sources, such asSIEM platforms (Splunk, IBM QRadar), network intrusion detection systems(IDS), endpoint detection and response (EDR) agent alerts, and networkflow analysers. The varied alert schema (naming of fields, severity scales,timestamp format,event type names, etc) poses a challenge for any system designed to aggregate or analyse alert data from diverse sources. All of these differences in format result in data incoherence and poor interpretability.

All upcoming alerts are standardised into a vendor-neutral JSON schema with 9 cardinalfields:source IP,DestinationIP, transport protocol,destination port,event timestamp,event category,SIEM rule identifier,severity score(normalized from 0 to 100), draw log contents. The benefit of a fixed Schema is that tool logic, whether it be for RAG embeddings or MCP tools, never has to worry about the input from different sources. A parsing adapter for each data source is added to the normalization layer.

3.3 MCP Integration Layer

The MCP server acts as the central coordination point for the proposed framework. MCP specification[10] was implemented and utilized via the MCP Python SDKto expose aSet of registers tooldefinitions which the LLM agent can use in their process of reasoning during an alert triage. Each tool definition provides details aboutits expected inputsand its outputaswell as its specific backend retrieval logic, so that the LLM agent can invoke tools with correctly typed inputs and correctly interpret outputs in its reasoning chain.

The proposed core four tools that were registered on the MCP server is listed in Table 2, Other than providing actual tools that implement specific data retrievals, the MCP Server authenticates to the various external data sources using their APIs, has a mechanism for applying request rate limiting to prevent overload of any single data retrieval source.Finally, MCP Server hasresponse caching capabilities, which vary depending on the volatility of a datatype.CVERequestsand

ATT&CKRequests are cached for a more elongated period in comparison with CTI requests. Every tool invocation, its arguments and the response from it are logged to an append log. Presently there isnomechanism to hash-chain hardeningisimplementedalthough it will bean area of priority in future work(as seen from section Limitation 8)

Table 2. MCP Tool Definitions, Knowledge Sources, and Retrieval Characteristics

Tool Name	Knowledge Source	Retrieval Method	Output Format	Caching Policy
CVE Lookup	NIST NVD [9]	Keyword + CVE-ID search	Structured JSON	On demand
ATT&CK Query	MITRE ATTaCK v 14 [8]	Semantic vector search	Technique JSON	On demand
Incident Retrieval	Curated SOC incident corpus (vector DB)	Dense ANN embedding search	Case narrative text	On demand
CTI Feed Query	MISP / OpenCTI platforms	IoC attribute matching	Indicator list	Cached — 5 min TTL

3.4 RAG Processing Layer

The RAG layer implements a hybrid retrieval approach combining both dense vector-based semantic search and structured keyword- and identifier-based search. Certain query patterns are most effectively solved by certain retrieval approaches: CVE identifiers and ATT&CK technique codes match best through structured identifier search, whereas semantic nearest neighbour search on high-dimensional vector representations proves most effective for retrieving relevant incident context given a historical alert text. Alert content is represented through SecBERT [13], a BERT model pre-trained on a large corpus of cybersecurity-focused texts. SecBERT provides domain-specific tokenisation and semantic embeddings that capture security specialised vocabulary more faithfully than generic sentence embedding models. These embeddings are indexed in a Chroma vector database and approximate nearest neighbour (ANN) search is enabled using HNSW. The vector database is populated offline with: 4,217 MITRE ATT&CK v14 technique entries [8]; 12,500 CVE records from 2022–2025 retrieved via the NIST NVD API [9], and filtered to only include those that contain a CVSS v3 score; and 3,800 carefully curated incident descriptions from public SOC playbook documentation, publicly reported incidents and threat intelligence reports. The documents retrieved are ranked by their cosine similarity, and a threshold is applied to select those that score above the threshold (in experiments, $\tau = 0.70$, the sensitivity of the threshold is examined in section 6). The top documents selected ($k = 5$, in experiments) form a structured context block, which is prepended to the alert representation. Each retrieved document is annotated with a label containing its origin and publication date, along with its relevance score. This allows the LLM to trace its deductions, producing evidence-based and human-auditable citations, while supporting analyst validation. 3.5 LLM Triage and Explanation Layer The LLM triage system makes use of Claude Sonnet 3.5 (model ID: claude-sonnet-3-5-20241022) as the principal reasoning model. It was chosen for its efficacy with structured data, support for the JSON output mode out-of-the-box, and ability to perform step-by-step chain of thought reasoning over a mixed-mode context. Inference is configured to run with temperature = 0.0 and max_tokens = 1024 in order to maximise output

consistency and minimise the maximum length of the generated output. A temperature of 0.0, while drastically, is not absolutely deterministic: small variations were observed in the explanatory text over repeated executions on the same input and are detailed in section 6. An alert is presented to the LLM via a prompt with three parts: 1) A system prompt that describes the triage task, specifies the output structure (JSON), directs the model to justify its output through the provided evidence, and alerts to low confidence recommendations; 2) The normalized, structured, and formattable JSON object of the alert; 3) The RAG-derived structured context block and associated sources. The LLM produces output following a predetermined JSON structure with fields for: severity_level (Critical / High / Medium / Low / Informational), recommended_action (Escalate / Investigate / Monitor / Dismiss), confidence_score (float, 0.0-1.0), matched_attack_techniques (list of ATT&CK identifiers), supporting_cves (list of CVE identifiers) and a reasoning_chain (natural language explanation of how the decision was made). The reasoning_chain is the principal vehicle through which explanations are rendered; rather than the opaque outputs of binary classifiers, specific alert attributes such as source reputation, destination port anomaly, or payload content are linked explicitly to specific sources of retrieved threat intelligence or threat activity patterns. These linkages enable analysts to verify the basis of every output, override recommendations as necessary, and leverage the reasoning chains for the structured training of junior analysts.

4. Implementation and Experimental Setup

4.1 Technology Stack

This was achieved by developing in python 3.11 following a modular microservices approach in which every layer can run as an independent service contained within its own docker image and deployed with the help of the docker compose tooling, making the environment totally reproducible on a dev environment and deployable on cloud instances of a SOC. In table 3 we can find the whole technology stack with their versions used.

Table 3. Implementation Technology Stack by Layer

Layer	Component / Version	Interface	Role in System
MCP Orchestration	MCP Python SDK v1.x	STDIO (local eval)	Tool registration, response caching, auth management, unprotected append log
Vector Store	Chroma DB v0.4	Python API	Approximate nearest-neighbour search; offline indexing of knowledge base
Embedding Model	SecBERT [13] via Sentence Transformers	Sentence Transformers	Domain-adapted BERT for cybersecurity corpora; alert and document embedding
LLM Reasoning Engine	Claude Sonnet 3.5 (claude-sonnet-3-5-2024 1022)	Anthropic API	Structured JSON output, temp=0.0, max_tokens=1024
RAG Framework	LlamaIndex v0.10	Python	Hybrid dense-vector and keyword retrieval pipeline
Containerisation	Docker 24.x / docker-compose v2	CLI	Full-stack environmental reproducibility

4.2 Dataset and Knowledge Base Construction

The principal study used data from CICIDS2017 [7]. CICIDS2017 is a benchmark commonly cited in intrusion detection that was originally compiled by the Canadian Institute for Cybersecurity. The dataset encompasses some 2.8 million bidirectional network flow recordings of different attack scenarios, as well as neutral background network data. The attack classes that we chose for the evaluation are Those for which an event or sequence of events will appear in SIEM security alerts; namely some denial of service attacks, brute-force attack, web attack and infiltration into the network. Since the original CICIDS2017 data contains flow data, instead of SIEM alerts, we have developed a rule-based data synthesis process that will translate flow level attack data to a SIEM format. This process maps SIEM alerting category information to threat categories commonly used by detection rules such as Suricata and Snort and, given threat categories, it maps to SIEM alerts. Benign flows have also been simulated to have been reported in security alerts in an amount proportional to their remoteness from the statistical mean. This resulted in 47,312 simulated alerts with a reported false positive rate of 61.4 percent. (28,970 non-critical, 18,342 threats.) It should be stated that the observed false positive rate is a product of our process and should not be taken as a reflection of a live deployment of a SIEM system which would have the possibility of real security incident recordings, this is also a limitation of the study. Our knowledge base has three parts: all the 4,217 technique objects which we used are obtained from MITRE ATT&CK database, release v14. Our other source comes from the NIST NVD application-programming interface. Specifically, 12,500 CVE records for the years 2022–2025, filtering out the records that have missing values for CVSS v3 scoring; finally, 3,800

text descriptions of incident narratives we obtained from publicly available security SOC playbooks. It is important to note that the narratives obtained come from public documentation and should not be regarded as representations of actual incident reports obtained from operational SOCs.

4.3 Evaluation Protocol and Baseline Systems

The proposed MCP-RAG system was evaluated against three baseline configurations selected to isolate the contribution of each architectural component. Baseline 1 is a standard rule-based SIEM correlation engine applied directly to the simulated alert set, representing the operational status quo in most enterprise deployments. Baseline 2 is a standalone LLM configuration using the identical model (claude-sonnet-3-5-20241022, temperature = 0.0, max_tokens = 1,024) receiving only the raw normalised alert — no retrieved context — to isolate the contribution of RAG retrieval. Baseline 3 is a RAG-only system implementing the full LlamaIndex retrieval pipeline with SecBERT embeddings and Chroma vector search but without MCP orchestration, isolating the marginal contribution of the MCP integration layer.

Four metrics are reported. FPR and TPR are computed over the full 47,312-alert evaluation set, with Wilson scoring 95% confidence intervals. Statistical significance of pairwise differences between the proposed system and each baseline is assessed using two-proportion z-tests at $\alpha = 0.05$; all reported improvements achieved $p < 0.001$. Average triage latency is measured as wall-clock time from alert receipt to structured JSON output delivery, averaged over 500 randomly sampled alerts per condition. Standard deviation of latency is reported to characterise variability. Explainability quality is assessed by three certified SOC analysts using a 5-point Likert scale over a stratified sample of 150 triage decisions, with inter-rater reliability assessed using Krippendorff's alpha over a common subset of 30 decisions rated by all three analysts.

5. Results and Discussion

5.1 False-Positive Rate and Detection Accuracy

The proposed MCP-RAG system achieved a false-positive rate of 21.4% (95% CI: [20.3, 22.5]), representing a statistically significant 41.3 percentage-point absolute reduction from the rule-based baseline of 61.4% ($p < 0.001$). The standalone LLM configuration without retrieval achieved an FPR of 43.2% ([41.9, 44.5]), confirming that LLM-based reasoning alone substantially outperforms static rule matching, but that the absence of retrieved contextual evidence limits disambiguation of borderline alert presentations. The RAG-only configuration achieved 26.1% ([24.9, 27.3]), confirming the large incremental benefit of retrieval-augmented context. The additional 4.7 percentage-point improvement of MCP-RAG over RAG-only reflects the contribution of structured, multi-source retrieval via MCP.

Notably, the true-positive rate of the proposed system (93.7%, 95% CI: [91.5, 95.5]) also exceeds all baselines, indicating that the reduction in false positives does not come at the cost of increased missed detections. The rule-based baseline's comparatively lower TPR of 78.2% reflects the well-known sensitivity of signature-based detection to minor evasive variation in attack technique. Table 4 presents full comparative results. All metrics should be interpreted in the context of CICIDS2017's 2017-era traffic and the simulation-derived baseline FPR.

Table 4. Comparative Performance Results Across All System Configurations (n = 47,312 Alerts)

System Configuration	FPR (%)	95% CI (FPR)	TPR (%)	95% CI (TPR)	Avg. Latency (s)	Expl. Score
Rule-Based SIEM (Baseline)	61.4	[60.2, 62.6]	78.2	[74.9, 81.2]	0.31	1.8 / 5.0
Standalone LLM — No RAG	43.2	[41.9, 44.5]	82.5	[79.5, 85.2]	4.70	3.1 / 5.0
RAG-Only — No MCP	26.1	[24.9, 27.3]	89.3	[86.8, 91.5]	6.20	3.7 / 5.0
Proposed MCP-RAG	21.4	[20.3, 22.5]	93.7	[91.5, 95.5]	3.91	4.6 / 5.0 †

Note. FPR = False-Positive Rate; TPR = True-Positive Rate. 95% CIs computed using the Wilson score method. Latency = mean wall-clock time per alert over 500 sampled alerts. All pairwise comparisons $p < 0.001$. † Explainability score from preliminary panel evaluation (n = 3 analysts, Krippendorff's alpha = 0.81); indicative only.

5.2 Triage Response Time and Latency Profiling

The MCP-RAG system achieved a mean triage latency of 3.91 ± 0.52 seconds per alert, compared to 6.20 seconds for the RAG-only baseline — a 36.9% reduction. This is counterintuitive given that the MCP-RAG system introduces additional processing components (four external tool calls, MCP message serialisation, and caching logic) relative to the RAG-only baseline.

Important interpretive caveat: the RAG-only baseline operated without response caching, whereas the MCP-RAG system achieved a 62.3% cache-hit rate on CVE and ATT&CK queries. The latency advantage therefore cannot be attributed solely to MCP architectural design; it is partially a caching effect. A fair comparison would require a RAG-only-with-equivalent-cache condition, not included in the present study and identified as a priority for future work.

Table 5. Component-Level Latency Profiling (Mean Seconds Per Alert, n = 500 Sampled Alerts)

Processing Component	Rule-Based (s)	LLM Only (s)	RAG-Only (s)	MCP-RAG (s) mean ± SD
Alert normalisation and JSON schema mapping	~0.02	~0.02	~0.02	0.02 ± 0.00
SecBERT embedding (alert text)	—	~0.08	~0.08	0.08 ± 0.01
Chroma ANN vector similarity search	—	~0.12	~0.12	0.15 ± 0.03
MCP tool calls (4 tools, with caching)	—	—	—	0.90 ± 0.21 *
LLM generation (prompt to structured JSON)	~0.29	~4.48	~5.98	2.76 ± 0.44
Total (mean per alert)	0.31	4.70	6.20	3.91 ± 0.52

*Note. * MCP tool call latency SD reflects variability between cache-hit (~0.04 s) and cache-miss (~1.8 s) cases; mean cache-hit rate was 62.3%.*

5.3 Explainability Assessment

A preliminary evaluation of the quality of the triage explanation was undertaken by three certified SOC analysts, at least one of whom was accredited by GCIA or GCIH (or similar certification) and has at least 3 years' operational SOC experience. Each SOC analyst assessed a stratified sample of 50 decisions, 150 decisions in total, using a 5-point Likert Scale designed to capture:clarity;completeness;operational value. Inter-rater reliability was calculated for a common subset of 30 decisions by all three analysts, yielding an alpha of 0.81 (good agreement). MCP-RAG achieved an overall score of 4.6/5.0, vs. RAG-only (3.7/5.0); the standalone LLM (3.1/5.0) and the rule-based system (1.8/5.0) The SOC analysts found the following aspects of the MCP-RAG explanation to be the most operationally valuable: - The inclusion of specific ATT&CK technique codes with descriptions. - The ability to cross-reference contextually relevant similar historical incidents along with their resolution. - The direct attribution of alert fields to indicators found in the retrieved threat intelligence. It should be noted these scores are preliminary and from a small initial pilot sample and should not be regarded as statistically generalisable to other audiences.

5.4 Discussion

The experimental findings collectively demonstrate that MCP provides a significant functional role within the proposed architecture, rather than just providing a convenient integration point. By enforcing a standard for tool interaction via typified, auditable schema: MCP promotes consistent and reproducible multi-source information retrieval which outperforms the standalone LlamaIndex retrieval;it demonstrably improves end-to-end processing latency, albeit in part due to MCP's response caching capability (discussed in Section 5.2), not purely design principles alone;It provides a neutral way of defining interoperability across SOC ecosystem integrations, and can serve as a standard that other SOC tools can adhere to with minimal fuss, and without requiring architectural modifications to the system; The formalise, auditable interaction schema enables the rendering of structured automated triage logs for archiving, post-incident analysis, training, etc.

6. Limitations

An honest disclosure of limitations for the accurate interpretation of the results is provided, Table 6 covers eight listed limitations and proposed solutions or mitigations.

Table 6. Study Limitations and Proposed Mitigations

Limitation Area	Description	Proposed Mitigation
Dataset Currency	CICIDS2017 captures 2017-era traffic and does not include modern evasion techniques. All metrics should be interpreted in this context.	Evaluate on CICIDS2019, CIC-IDS-2018, or a live SOC stream in future work.
Alert Simulation Fidelity	The 61.4% baseline FPR is self-generated and not validated against a live SIEM deployment.	Deploy system in a live SOC environment for longitudinal validation.
Knowledge Base Coverage	Incident narratives are from public documentation, not live operational records.	Partner with SOC operators to incorporate redacted real-incident corpora.
Explainability Evaluation Scale	The panel of three analysts is insufficient for statistically generalisable claims.	Conduct large-scale human evaluation ($n \geq 30$ analysts) across multiple organisations.
LLM Non-Determinism	Despite temperature = 0.0, minor variation in explanatory text occurs across identical inputs.	Log and compare reasoning chain outputs across repeated runs.
Hyperparameter Ablation	Retrieval depth $k = 5$ and threshold $\tau = 0.70$ selected via preliminary tuning without full grid search.	Conduct full grid search over $k \in \{3, 5, 7, 10\}$ and $\tau \in \{0.60–0.80\}$.
Commercial API Dependency	Claude Sonnet 3.5 introduces data residency concerns and third-party reproducibility barriers.	Evaluate locally hosted open-weight alternatives (Mistral-7B, Llama-3-8B, Phi-3).
Audit Trail Immutability	The append-only audit log has no cryptographic tamper-evident mechanism.	Implement hash-chaining or integrate with an immutable ledger service.

7. Conclusion and Future Work

7.1 Conclusion

The MCP-orchestrated Retrieval-Augmented Generation (RAG) architecture developed and validated in this paper aims to automate security alert triage in Security Operations Centers (SOCs). Our approach utilizes the Model Context Protocol (MCP) to establish a standardized data integration layer that connects security information and event management (SIEM) systems, Common Vulnerabilities and Exposures (CVE) databases, the MITRE ATT&CK knowledge base, curated incident repositories, and live threat intelligence feeds. This integration feeds the collected context into a large language model (LLM)-powered reasoning engine, facilitating real-time, context-aware alert triage without the need for retraining the LLM.

We conducted an empirical evaluation of our proposed system using the CICIDS2017 dataset. This dataset consists of 47,312 synthetic SIEM alerts, features traffic from 2017, and has a simulation-derived baseline false-positive rate (FPR). Our system demonstrated statistically significant improvements across all three baseline comparison metrics ($p < 0.001$). Specifically, we achieved a 41.3 percentage-point reduction in FPR (from 61.4% to 21.4%, with a 95% confidence interval [CI] of [20.3, 22.5]). Furthermore, we observed a 36.9% reduction in alert triage latency (from an average of 6.20 seconds to 3.91 ± 0.52 seconds, partly attributed to the

use of MCP's response caching mechanism) and a high true-positive rate (TPR) of 93.7% (95% CI [91.5, 95.5]). Preliminary evaluation by cybersecurity analysts rated the system's explainability at 4.6 out of 5.0, with a Krippendorff's alpha value of 0.81, which should be treated as indicative and awaiting validation in large-scale human studies.

7.2 Future Research Directions

Priority future research directions include: (a) rigorous evaluation on contemporary datasets such as CICIDS2019, CIC-IDS-2018, or live SOC streams; (b) systematic hyperparameter ablation studies (specifically, on retrieval depth and the similarity threshold used in document retrieval); (c) extensive longitudinal deployment within a live SOC environment to assess real-world performance, stability, and operator acceptance; (d) exploration of locally hosted open-weight LLMs to address concerns related to data residency and reproducibility, while potentially reducing costs; (e) a comprehensive large-scale human evaluation of the system's explainability ($n \geq 30$ security analysts) to obtain statistically robust results; (f) investigation and development of a federated MCP-RAG deployment model for privacy-preserving knowledge sharing across different SOCs; (g) robust security hardening of the MCP integration layer to mitigate potential vulnerabilities; and (h) implementation of cryptographic hash-chaining for the audit trail to enhance its integrity and tamper-proofness.

References

[1] Ahmed, S., et al. (2025). AI-Augmented SOC: A Survey of LLMs and Agents for Security Automation. Computers & Security (Elsevier), 140, Article 103976. https://doi.org/10.1016/j.cose.2024.103976

[2] Hou, X., Wang, S., et al. (2025). Model Context Protocol (MCP): Landscape, Security Threats, and Future Research Directions. arXiv preprint arXiv:2503.23278.

[3] Radosevich, B., & Halloran, J. T. (2025). MCP Safety Audit: LLMs with the Model Context Protocol Allow Major Security Exploits. arXiv:2504.03767.

[4] Lodge, B. (2025). RAG Against the Machine: Using RAG and MCP to Fortify Cybersecurity Defenses. Conference Presentation, BSides Las Vegas 2025. [Grey literature — cited for prior work context only.]

[5] Mansour, A., Pena, J., & Zhao, L. (2025). Advancing Autonomous Incident Response: Leveraging LLMs and Cyber Threat Intelligence. arXiv preprint arXiv:2503.14874.

[6] Lewis, P., Perez, E., Piktus, A., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems (NeurIPS), 33, 9459–9474.

[7] Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. Proceedings of ICISSP 2018, 108–116. https://doi.org/10.5220/0006639801080116

[8] MITRE Corporation. (2024). MITRE ATT&CK Framework v14. https://attack.mitre.org

[9] National Institute of Standards and Technology. (2024). National Vulnerability Database. https://nvd.nist.gov

[10] Anthropic. (2024). Model Context Protocol Specification. https://modelcontextprotocol.io

[11] Veerasamy, N., & Grobler, M. (2024). Security Operations Centers: Architecture, Challenges, and Automation. Journal of Information Security and Applications (Elsevier), 82, 103731.

[12] IBM Security. (2023). Cost of a Data Breach Report 2023. IBM Corporation. https://www.ibm.com/reports/data-breach

[13] Jackaduma. (2021). SecBERT: A Pre-trained Language Model for Cyber Security Text. HuggingFace Model Repository. https://huggingface.co/jackaduma/SecBERT