International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

ALUSKORT: A HIERARCHICAL MULTI-AGENT COGNITIVE ARCHITECTURE FOR AUTONOMOUS SECURITY OPERATIONS

Abhinav Sharma[1*], Jawahar Thakur[2], T.P. Sharma[3], Rahul Monie[4], Sachit Shivam[5]

[1*]Department of Computer Science, Himachal Pradesh University, India E-mail: aasvi2006@gmail.com

[2]Department of Computer Science, Himachal Pradesh University, India E-mail: jawahar.thakur@hpuniv.ac.in

[3]Department of Computer Science and Engineering, National Institute of Technology, Hamirpur, India E-mail: teek@nith.ac.in

[4]Aluskort E-mail: rahul@aluskort.com

[5]Independent Researcher E-mail: sachit.shivam@gmail.com

Abstract:

Security Operations Centers (SOCs) are overwhelmed by escalating alert volumes, creating a critical need for autonomous workflows that accelerate incident response. We present *ALUSKORT, a hierarchical multi-agent framework designed to automate the full incident investigation lifecycle by uniquely combining deterministic guardrails with a sequence of LLM-driven reasoning agents.* The framework proves how a smaller, domain-specific open-source model—in this case, a quantized 8B-parameter Foundation-Sec-8B—can be successfully steered by a structured, guardrail-driven pipeline to perform sophisticated reasoning on commodity hardware (a single GPU). A comprehensive, multi-faceted evaluation—testing for factual accuracy, context-aware prioritization, safety, and reasoning quality—confirmed the framework's effectiveness. The system produced high-quality investigative artifacts, with a panel of three LLMs assigning consensus scores of 13.33–16.33/20. Critically, our work reveals key insights for practical implementation. Performance profiling measured the core reasoning pipeline's average latency (from IOC extraction to question generation) at 428.78 seconds (approx. 7 minutes), and identified a clear optimization target in the IOC extraction phase (78.1% of this time). Furthermore, our analysis of prompt architecture showed that a base model's attention can be more than doubled through structural optimization. Ultimately, ALUSKORT provides a reproducible blueprint for building effective, safe, and accessible autonomous capabilities to address the escalating challenges faced by modern Security Operations Centers.

Keywords: AI Agent, Security Operation Centre(SOC), AI Driven Cyber Security, Incident Response, Large Language Models(LLMs)

1 Introduction

The modern cybersecurity landscape places unprecedented pressure on Security Operations Centers (SOCs). With incident rates rising by 65%, mean detection times lingering at 196 days (2018), and U.S. cybercrime losses exceeding $12.5 billion in 2023, the need for effective security is undeniable. While SOC adoption is now near-universal and the services market continues to expand [4,6,7], their operations are plagued by persistent bottlenecks. Analysts face overwhelming alert volumes (24k–130k/day), where false-positive rates approach 99% and only ~0.01% of alerts represent a successful

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

attack. This, combined with opaque toolchains and metadata loss, erodes analyst trust and critically slows investigations [7-9].

Large Language Models (LLMs) promise a paradigm shift, offering a path from reactive detection to proactive, autonomous operations through security-specific tuning, contextual reasoning, and direct tool interaction [11]. LLMs have already demonstrated value in knowledge extraction, reasoning over heterogeneous logs, and orchestrating multi-step threat investigations when grounded in high-quality data [11,13,26-28]. However, their widespread adoption is hindered by significant barriers. Prior work with structured pipelines like SHIELD highlights practical constraints such as token limits and prompt design, while core challenges in data quality, robustness, and explainability remain unsolved [12].

To address these gaps, we present ALUSKORT (Agentic Learning Unified Security Knowledge Operation Response Threat System), a hierarchical multi-agent framework that harnesses the power of LLMs while mitigating their weaknesses. The system integrates deterministic guardrails—such as noise reduction and fuzzy validation—with a structured reasoning pipeline that guides an alert through seven phases: IOC extraction, investigation prioritization, alert contextualization, hypothesis generation, question generation, tool calling and finally, verdict and remediation synthesis.

Crucially, unlike AI approaches focused on raw telemetry analysis, ALUSKORT operates post-SIEM. It assumes an alert has already been generated by a SIEM(Security Information and Event Management) like Wazuh or Splunk and focuses exclusively on the subsequent investigative, contextualization, and remediation stages. This bridges the critical gap between alert generation and actionable incident response, allowing for seamless integration into existing SOC detection stacks.

Our contributions are threefold:

- A novel, unified architecture that systematically integrates deterministic guardrails with AI-driven reasoning.
- A practical, multi-phase workflow that mirrors and automates real-world SOC investigative processes.
- A pilot evaluation demonstrating the framework's operational feasibility using an open-source LLM on real-world security alerts.

2 Related Work

Our work is situated at the intersection of Large Language Models in cybersecurity and the architecture of multi-agent AI systems.

2.1 LLMs in Cybersecurity

The application of Large Language Models (LLMs) across security domains is rapidly expanding, driven by strategies like security-oriented training, tool integration, and retrieval-augmentation [11,13]. Prior work confirms that LLMs excel at knowledge extraction, contextual reasoning, and orchestrating threat intelligence, showing the potential to shift SOC operations from reactive to autonomous modes when properly grounded [26-28]. A survey of 185 papers identifies three primary methods for adapting LLMs to security tasks: prompt engineering, fine-tuning, and external augmentation with tools or retrieval systems [11]. These techniques are applied to a wide array of problems, including threat intelligence analysis, malware detection, log analysis, program repair, and incident response, though their dual-use offensive capabilities are also acknowledged [13].

2.2 Multi-Agent and Agentic Frameworks

To manage complex, multi-step tasks, research is increasingly turning to agentic AI—systems composed of autonomous, adaptive agents that can plan, reason, and act [16]. The potential for these

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

frameworks to drive efficiency gains in SOC operations is evident in commercial offerings like Darktrace, Charlotte AI, and Microsoft Security Copilot, which have demonstrated improvements in autonomous detection and response[16,18].

Methodologically, academic research provides key architectural insights[25]. The Mixture-of-Agents (MoA) approach, for instance, shows that coordinating between proposer and aggregator agents improves output quality [19]. Similarly, task-specialized, sequentially planned multi-agent systems in adjacent domains, such as VulnBot, have proven effective at mitigating the context limitations and brittle long-horizon reasoning of single models [20]. The success of these specialized, sequential architectures directly forms ALUSKORT's design, which relies on role-specific agents operating in a clearly defined pipeline.

2.3 Gap Analysis

Despite these advances, a critical gap exists between current capabilities and the needs of a modern SOC. The vast majority of existing AI applications focus on upstream detection tasks—anomaly detection, malware classification, or log analysis—rather than the complex investigation and contextualization that must follow a SIEM alert [6,11-13.]. While effective at flagging suspicious activity, they rarely provide structured, explainable pipelines for the post-alert analysis.

Furthermore, existing agentic systems that attempt to bridge this gap are limited by two key factors: Commercial offerings are powerful but operate as proprietary, opaque systems. They often rely on closed-source models via commercial APIs, which prevents organizations from independently assessing, customizing, or reproducing their decision-making processes [16].

Academic prototypes, to date, have been limited to narrow scenarios, often tackling independent, separate tasks such as IOC extraction or vulnerability analysis in isolation. They rarely address the holistic, end-to-end workflow required to investigate the high-volume, noisy alert streams found in a real SOC [19,20].

This reveals a clear need for an open, integrated, post-SIEM framework capable of systematically transforming raw alerts into actionable, explainable incident responses, while incorporating robust guardrails to suppress hallucinations and ensure factual grounding.

3 The ALUSKORT Architecture

ALUSKORT (Agentic Learning Unified Security Knowledge Operation Response Threat System) is a hierarchical multi-agent framework designed to automate post-SIEM incident response. Its architecture is guided by three core principles:

1. **Tool-Agnostic Design:** The framework is architected to be independent of any specific vendor's toolchain. While this work demonstrates its capabilities using the open-source Wazuh SIEM, its modular integration points are designed to connect with various SIEMs, EDRs, or other data sources, preventing vendor lock-in.

2. **Guardrail-Driven Design:** It integrates deterministic checks, such as schema validation and fuzzy matching, to mitigate LLM hallucinations and ground all outputs in factual data.

3. **Operational Alignment:** The workflow mirrors a real-world SOC investigation, focusing on the entire process from alert to remediation, rather than on isolated AI tasks.

3.1 The Architectural Workflow

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

The framework operates as a coordinated pipeline of ten sequential phases(Fig. 1). Specialized agents are responsible for executing the logic at each stage, transforming a raw security alert into a fully investigated incident report with actionable remediation steps.

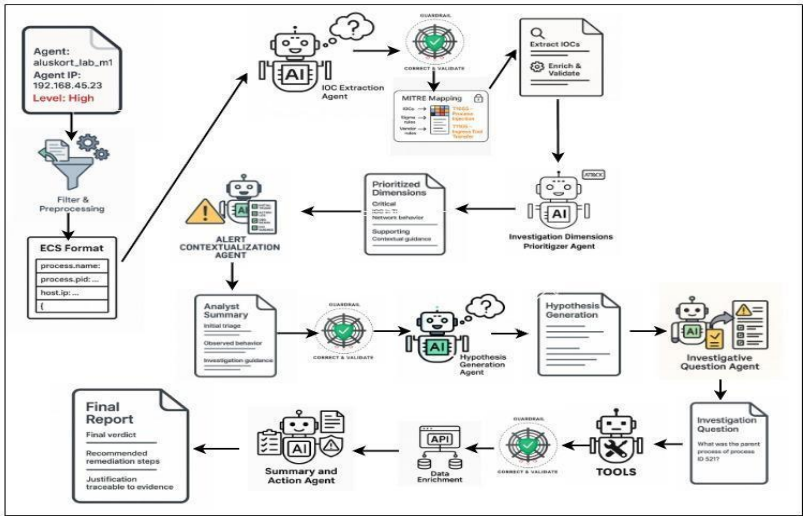

Fig. 1: Architecture of ALUSKORT (Agentic Learning Unified Security Knowledge Operation Response Threat System)

3.2 Agents Roles and Workflow

Phase 0 – Alert Preprocessing: The raw alert a∈A is cleaned using a deterministic function g_{clean} to remove irrelevant metadata such as compliance fields, nested logs, and excess whitespace. This ensures that downstream LLM operations operate on a normalized and noise-free alert representation a_{clean}.

$$a_{\text{clean}} = g_{\text{clean}}(a)$$

- **Optional ECS Mapping((Proposed):** To make ALUSKORT portable across SIEMs and data sources, we define an optional mapper h_{ecs}: A → E that converts the cleaned alert to Elastic Common Schema [ECS Schema]with strict type/shape checks : $e = h_{ecs}(a_{clean})$, $e \in E$[22]. While this component was not implemented in the current pilot study, it is envisioned as a key feature to ensure future portability across diverse data sources.

Phase 1 – IOC Extraction

Using prompt-driven inference via F_{LLM}, the cleaned alert a_{clean} is passed to a fine-tuned extraction template $P_{extract}$ and few-shot examples $E_{extract}$. The LLM produces E_{raw}, a structured multi-line text listing potential Indicators of Compromise (IOCs) and relevant context fields.

$$E_{\text{raw}} = F_{\text{LLM}}(a_{\text{clean}} \mid P_{\text{extract}}, E_{\text{extract}})$$

Phase 2 – IOC Validation & Enrichment

The raw extractions are validated against the original alert using a grounded function $g_{validate}$, which performs fuzzy matching and key confirmation. The validated dictionary Ev is then enriched using g_{enrich} to classify IP addresses and annotate fields with semantic context, resulting in $E_{enriched}$

$$E_v = g_{\text{validate}}(E_{\text{raw}}, a_{\text{clean}})$$
$$E_{\text{enriched}} = g_{\text{enrich}}(E_v)$$

Phase 3 – Investigation Prioritization

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

To guide investigative reasoning, an LLM computes dimension-wise weights $\{w_i\}$based on the enriched IOCs and MITRE technique present. The prompt P_{prio} and examples E_{prio} help generate P, a prioritization schema constrained by $\sum w_i=100, w_i \geq 5$. This allows for strategic focus across six investigative dimensions viz. Process Behavior, IOC in Threat Intelligence, Host Vulnerability, User Attribution, Network Behavior, and Contextual Guidance

$$P = F_{\text{LLM}}(E_{\text{enriched}} \mid P_{\text{prio}}, E_{\text{prio}}) \quad \text{s.t.} \quad \sum_{i=1}^{6} w_i = 100,\ w_i \geq 5$$

- **MITRE Technique Handling(Proposed)**: To handle alerts that lack MITRE technique IDs, the architecture proposes a deterministic resolver that can infer them from other data. As the Wazuh alerts used in our evaluation already contained these IDs, this module was not required and therefore not implemented in the pilot.

In this study, Wazuh already supplied MITRE IDs; we used them to steer prioritization via an **ATT&CK-aligned prior** and a **telemetry check** that down-weights unsupported dimensions, and we disabled the technique-inference module.

Phase 4 – Alert Contextualization & Correction

An initial four-bullet summary S is generated by the LLM using ABLE-style prompts. This summary includes triage context, actor/location, behavior observed, and guidance. A correction function $g_{correct}$ matches quoted entities in SSS to the enriched IOC dictionary $E_{enriched}$, producing a hallucination-free version S′.

$$S = F_{\text{LLM}}(E_{\text{enriched}}, P \mid P_{\text{summary}}, E_{\text{summary}})$$
$$S' = g_{\text{correct}}(S, E_{\text{enriched}})$$

Phase 5 – Hypothesis Generation

The corrected summary S′ is transformed into a formal investigative hypothesis H through prompt-driven synthesis. This phase encourages a proactive analytic stance, capturing the likely root cause or objective of the alert in a declarative format grounded in observed behavior.

$$H = F_{\text{LLM}}(S' \mid P_{\text{hypo}}, E_{\text{hypo}})$$

Phase 6 – Question Generation (CoT)

A chain-of-thought prompt is used to generate natural language questions $Q=\{q_1,\ldots,q_m\}$that target evidence gaps across prioritized dimensions[15]. The inputs include validated IOCs, summary, prioritization, and hypothesis, enabling the LLM to ask targeted, investigative queries.

$$Q = F_{\text{LLM}}(E_v, P, S', H \mid P_{\text{CoT}}, E_{\text{CoT}})$$

Phase 7 – Question Parsing & Tool Mapping

Each question in Q is parsed deterministically via f5 to extract intent and relevant parameters. A tool mapping table M_{map} is used to map each question q_i to a tuple (t_i,θ_i), resulting in an execution plan

$$Plan = f_5(Q) = \{(t_i, \theta_i)\}_{i=1}^{m}$$

Phase 8 – Tool Execution & Evidence Gathering

For each mapped question-tool pair, f_6 executes the appropriate investigative function and collects evidence. The result is a set of question-evidence pairs, which provides objective observations to answer each query generated earlier.

$$E_{\text{new}} = f_6(Plan) = \{(q_i, e_i)\}_{i=1}^{m}$$

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Phase 9 – Final Synthesis, Verdict & Remediation

All prior context—hypothesis, validated IOCs, and new evidence—is synthesized using a specialized prompt P_{final}. The LLM returns a triple (V,R,J) denoting the final verdict (True/False Positive or Needs Human Review), the recommended remediation steps, and a justification traceable to evidence.

$$(V, R, J) = F_{\text{LLM}}^{\text{final}}(H, E_v, E_{\text{new}} \mid P_{\text{final}})$$

Final decision, remediation, and justification.

3.3 LLM Selection Rationale: Foundation Sec 8B:

The engine powering ALUSKORT's agentic workflow is Foundation-Sec-8B, an open-weight model chosen for its unique balance of domain-specific expertise, proven performance, and practical deployability.

First and foremost, the model is purpose-built for cybersecurity. It was trained by Cisco's Foundation AI team on a curated 5.1B token corpus of security-focused data. With a training cutoff of April 10, 2024, its knowledge base includes recent vulnerabilities, updated MITRE ATT&CK mappings, and contemporary defensive practices. This domain-specific pretraining directly enhances key ALUSKORT tasks like IOC extraction and threat prioritization. Its effectiveness is demonstrated on domain benchmarks, where it achieved 67.39% on CTI-MCQA and 75.26% on CTI-RCM, outperforming larger, general-purpose models in security-specific reasoning[2,3].

Second, the choice of Foundation-Sec-8B enables a fully on-premises and controllable deployment, which is a critical advantage over commercial API-based models. Recent studies of API-driven SOC tools (e.g., GPT-3.5, GLM-4) report significant operational hurdles, including high costs (up to $0.21 per alert), substantial token consumption (6k–20k per task), high latencies (10–50s), and integration friction due to indirect tool invocation [10]. In contrast, our local deployment of Foundation-Sec-8B offers:

- **Zero Recurring Costs:** Eliminating per-alert API fees.
- **Data Sovereignty:** The Apache 2.0 license permits on-premises or air-gapped deployment, ensuring full control over sensitive alert data.
- **Tight Integration:** It enables direct, low-latency coupling with SOC tools like Wazuh and MISP, which is essential for responsive automation.

Finally, built on the Llama 3.1 architecture with 8B parameters, the model is highly efficient, capable of running on a single high-memory GPU. This makes the ALUSKORT framework not only powerful but also accessible for deployment in typical enterprise environments without requiring extensive, specialized hardware.

3.4 Execution Parser for Tool Grounding

Given generated questions $Q = \{q_1, \ldots, q_m\}$ and an allow-listed tool registry $\mathcal{T}$, the parser f_5 uses a few-shot CoT planner (ValidatedCoTPlanner) to select valid tools for each q_i (supports single/multi/none). For each selected tool t_i, an ArgumentPlanner asks the LLM only to infer arguments θ_i from the alert context (final JSON assembled in code). Steps are retained only if their non-system parameters are grounded against the validated IOC set E_v (system fields such as agent_id/date are exempt). Per-question plans Π_{q_i} yield:

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$\Pi = \bigsqcup_{i=1}^{m} \Pi_{q_i}$.

Executor f_6 runs fully grounded pairs (t_i, θ_i) via the tool interface (integration layer) and returns normalized evidence $E = \{(q_i, e_{ij})\}$.

Algorithm 1 — Phase 7: f_5 (Tool Selection → Argument Inference → Grounding)

Input: Q, $\mathcal{T}$, E_v and alert context

Output: $\Pi = \bigsqcup_{i} \Pi_{q_i}$ with $\Pi_{q_i} = \{(t_i, \theta_i)\}$

1. Initialize: Π←∅.
2. For each $q_i \in Q$:
3. a) Use **ValidatedCoTPlanner** to select tools from $\mathcal{T}$ (parse **[TOOLS TO USE]**; supports {single | multi | none}).
4. b) For each selected tool t_i, prompt **ArgumentPlanner** to infer θ_i from the alert context (parse **[ARGUMENTS]**; JSON assembled in code).

c) If all non-system parameters in θ_i are grounded in E_v, update:

$\Pi \leftarrow \Pi \cup \{(t_i, \theta_i)\}$.

5. Return: Π

(Hallucinated tool names are discarded by the allow-list; ungrounded steps are dropped.)

Algorithm 2 — Phase 8: f_6 (Execution & Evidence Gathering)

Input: Π

Output: $E = \{(q_i, e_{ij})\}$

1. Initialize: E←∅.
2. For each $(t_i, \theta_i) \in \Pi$:

a) Execute via the integration layer: $e_{ij} \leftarrow$ execute_tool(t_i, θ_i).

b) Normalize to a standard record {tool,args,ts,rows,digest} and add (q_i, e_{ij}) to E.

3. Return: E.

Here,

$\mathcal{T}$: allow-listed tool registry; E_v: validated IOCs;

q_i: question; t_i: tool; θ_i: arguments;

Π: execution plan; E: evidence set.

3.5 Guardrails & Hallucination Control

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

A core design principle of ALUSKORT is to balance the reasoning capabilities of LLMs with a strict requirement for factual accuracy. To achieve this, we apply targeted guardrails exclusively to verifiable, IOC-bearing outputs (e.g., IPs, domains, hashes, file paths), while allowing generated prose (summaries, hypotheses) to retain the benefits of the model's domain training.

These controls are implemented at three critical points in the pipeline:

1. **Initial IOC Validation:** Immediately after extraction, every potential IOC is validated. This involves removing placeholders, using regex to confirm MITRE ID formats, and performing fuzzy token matching (threshold 90) against the original alert text. Only verified IOCs proceed.

2. **Summary Grounding:** After the initial summary is generated, a post-hoc reconciliation process aligns all quoted entities with the validated IOC list. Mismatches are replaced with the nearest valid IOC (threshold 85) or marked with an <IOC.UNRESOLVED> tag.

3. **Tool Argument Verification:** Before execution, all parameters for a tool call are checked to ensure they are traceably linked to the validated IOCs from Phase 1 (system parameters like agent_id are exempt).

The underlying mechanism for these checks is a token-based string similarity algorithm using the RapidFuzz library. This method compares sets of tokens rather than raw string sequences, making it resilient to minor formatting variations while maintaining high matching precision. Thresholds were optimized to balance false positives and negatives.

Crucially, ALUSKORT employs a correction-over-rejection policy. Instead of discarding an entire output that contains an ungrounded value, the system retains the reasoning but corrects or redacts the specific unverified indicator. This preserves valuable analytic context for a human reviewer while preventing the propagation of hallucinated data. This entire process is logged, ensuring every grounding decision is reproducible and auditable.

3.6 Notation & Symbols

Notation Glossary

$a \in \mathcal{A}$: The raw alert object (e.g., a JSON dictionary from Wazuh).

a_{clean}: The cleaned alert after noise removal.

E_{raw}: Raw extracted Indicators of Compromise (IOCs) and contextual fields, as a text block.

E_v: Validated IOCs, in dictionary form, confirmed present in a_{clean}.

E_{enriched}: Enriched IOCs, where each IP is classified (Public/Private/Loopback, etc.).

P: The prioritization plan, consisting of:

- A set of six weights $\{w_i\}$ over dimensions

- Three sets: Critical, Supporting, Low-Priority dimension groups

S: Initial ABLE-style summary.

S': Self-corrected summary after hallucination removal.

H: Investigation hypothesis.

$Q = \{q_1, \ldots, q_m\}$: Generated investigative questions.

$Plan = \{(t_i, \theta_i)\}$: Mapping of each question q_i to a tool t_i and parameters θ_i.

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$E_{\text{new}} = \{(q_i, e_i)\}$: New evidence answers for each question.

$V \in \{\text{TP}, \text{FP}, \text{NHR}\}$: Final verdict—True Positive, False Positive, or Needs Human Review.

R: Remediation steps if $V = \text{TP}$.

J: Final natural-language justification tracing evidence to verdict.

F_{LLM}: Generic LLM inference function.

f_i, g_i: Phase-specific functions.

P_*, E_*: Prompt templates and few-shot examples for LLM calls.

3.7 Formalization

Mathematically, the entire process is modeled as a nested composition of deterministic and LLM-driven phases:

$$(V, R, J) = f_9(f_8(f_7(f_6(f_5(f_4(g_{\text{enrich}}(g_{\text{validate}}(f_1(g_{\text{clean}}(a))))), P, S', H)))))$$

This formulation enables transparency, modularity, and explainability, while balancing LLM flexibility with rule-based guardrails throughout the autonomous security operations pipeline.

4. Experimental Evaluation

This section presents a multi-faceted evaluation of the ALUSKORT framework. Our goal was to assess the performance of each critical reasoning phase, from initial data extraction to the final generation of investigative queries. The evaluation is structured around five key experiments: (1) factual accuracy in IOC extraction, (2) context-aware investigation prioritization, (3) overall quality of the reasoning pipeline, (4) the impact of prompt architecture, and (5) the safety of the tool-mapping pipeline.

4.1 Experimental Setup

- **Evaluation Scope:** To isolate the LLM's cognitive performance, we evaluated the core reasoning pipeline from Phase 1 (IOC Extraction) through Phase 7 (Tool Mapping). The final synthesis and tool execution phases were excluded as they depend on external tool availability and latency. All experiments were conducted on cleaned, raw alert data (gclean enabled).
- **Dataset:** The evaluation was conducted on a pilot dataset of four representative Wazuh alerts, each containing pre-mapped MITRE ATT&CK technique IDs (T1105, T1059.007, T1055, T1055.001).
- **Inference Environment:** All experiments used the Foundation-Sec-8B base model deployed locally in a Kaggle environment on a single NVIDIA T4 GPU (16 GB VRAM) with 8-bit quantization. Greedy decoding was used to ensure deterministic and reproducible outputs.

4.2 Experiment 1: Factual Accuracy of IOC Extraction

Objective: To measure the system's ability to extract IOCs with perfect precision (zero hallucinations) while maximizing recall.

Methodology: We performed an ablation study comparing 2-shot, 3-shot, and 5-shot prompt configurations. After generation, a validation script performed fuzzy matching to ensure every extracted IOC was verifiably present in the source alert.

Results: The guardrails were 100% effective, achieving perfect precision across all configurations. Recall scaled directly with the number of examples, increasing from 0.700 (2-shot) to 0.760 (5-shot),

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

demonstrating the importance of diverse examples for coverage(Fig. 2). The 5-shot configuration, which yielded the best F1-score of 0.864, had an average inference time of 335.1 seconds(Table 1).

Few-Shot Setting	Precision	Recall	F1-score	Avg. Inference Time(s)
2-shot	**1.000**	**0.700**	**0.823**	**174.8**
3-shot	**1.000**	**0.740**	**0.851**	**173.1**
5-shot	**1.000**	**0.760**	**0.864**	**335.1**

able 1: Extraction Accuracy and Inference Time per configuration (Avg over 4 alerts)

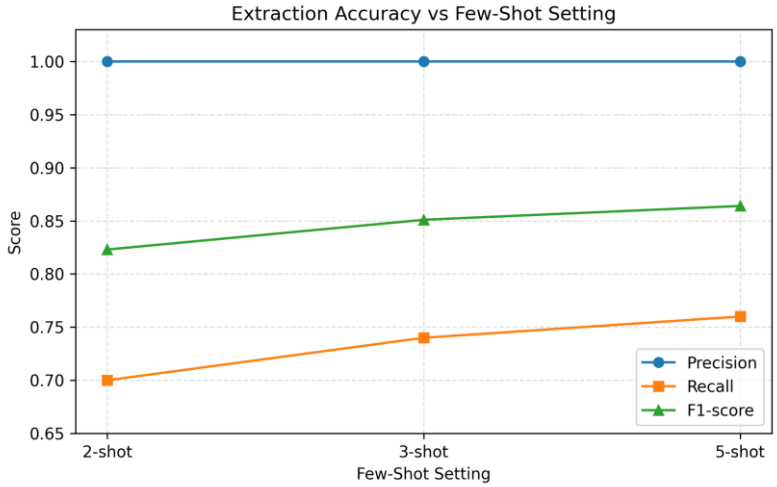

Fig. 2: Extraction accuracy over different few-shot configurations

4.3 Experiment 2: Context-Aware Investigation Prioritization

Objective: To evaluate the system's ability to generate an investigative plan that aligns with expert guidance (MITRE ATT&CK)[23] while intelligently adapting to the specific evidence within an alert.

Methodology: We analyzed the prioritization plans generated for each of the four alerts. The output was qualitatively compared against the official detection guidance on MITRE's technique pages to assess both alignment and the rationale for any deviations (Table 2).

Results: The system demonstrated a crucial capability for context-aware refinement (Fig. 3). In all cases, it aligned with MITRE's high-level recommendations (e.g., always prioritizing "Process Behavior") but intelligently adjusted weights based on the available telemetry. For example, for technique T1105, it correctly deprioritized "Network Behavior"—a dimension MITRE rates highly—because the source alerts contained no network logs. This prevents the system from pursuing investigative paths unsupported by evidence, a key step in avoiding the verbose, speculative ("philosopher") behavior common in unsteered LLMs.

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Table 2: LLM generated Prioritization Plans

Alert ID	MITRE Techniques	Critical Dimensions	Contextual Deviation Rationale
1753761379	T1105, T1059.007	Process Behavior, IOC in Threat Intel	Network Behavior deprioritized due to absent telemetry.
1753761350	T1055, T1055.001	Process Behavior	IOC weight reduced; no threat feed linkage present in the alert.
1753752610	T1055	Process Behavior, Host Vulnerability	Host Vulnerability elevated for persistence/escalation context.
1753608159	T1105, T1059.007	Process Behavior, IOC in Threat Intel	Network Behavior deprioritized due to absent telemetry.

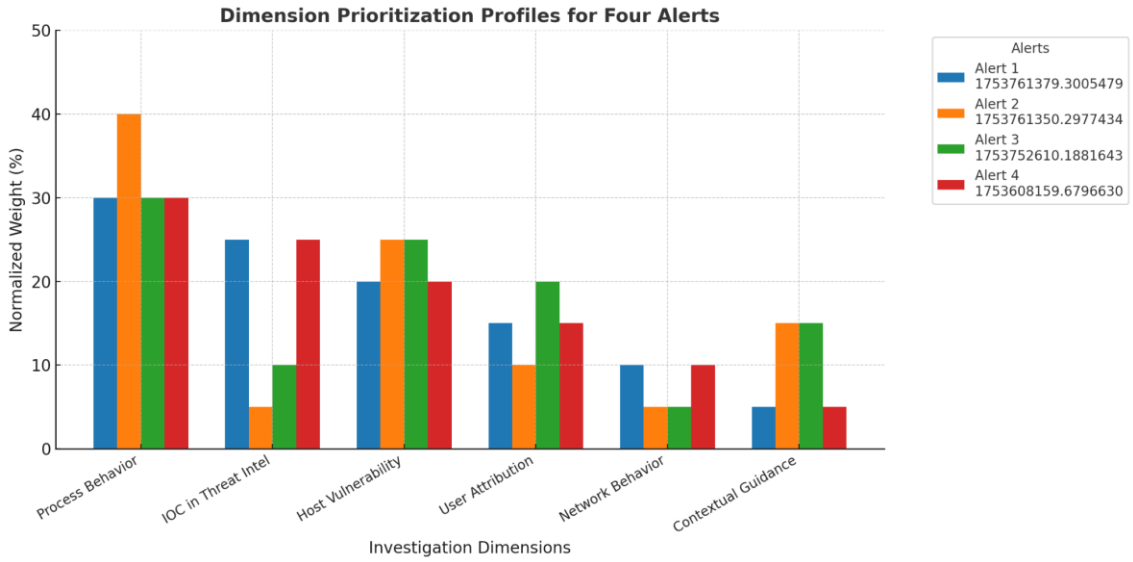

Fig. 3: Dimension Prioritization Profiles for 4 alerts

Implications:

This prioritization mechanism is a critical guardrail against a common failure mode of LLMs: what we term "philosopher behavior," where the model generates verbose and speculative narratives. By forcing the reasoning to anchor to available evidence, the system ensures that the subsequent hypothesis and question generation phases are targeted and productive. Furthermore, the design choice to never assign a zero weight to any dimension acts as a fail-safe, ensuring that no investigative path is ever completely closed off, balancing efficiency with investigative completeness.

4.4 Experiment 3: Quality of Investigative Reasoning

Objective: To assess the logical coherence, contextual awareness, and reasoning fidelity of the prioritization, contextualization, and hypothesis generation phases.

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Methodology: As these qualitative aspects cannot be measured by simple metrics, we used a multi-judge LLM evaluation framework. The outputs for each of the four alerts were independently scored by ChatGPT-4.0, Grok-4 (Expert), and Claude Sonnet 4.0 based on a structured rubric assessing IOC grounding, logical consistency, and MITRE alignment (Fig. 5).Each judge was guided by a structured rubric that instructed them to score the output from 1-5 across four key categories:

1. Summary Quality: Logical, concise, and grounded in IOCs.
2. Hypothesis Coherence: Plausible and aligned with evidence.
3. Question Relevance: Actionable and targeting evidence gaps.
4. Prioritization Alignment: Consistent with MITRE guidance and alert context.

Results: The system produced high-quality reasoning artifacts, with merged consensus scores ranging from 13.33 to 16.33 out of 20 (Table 3). The analysis highlighted ALUSKORT's ability to make intelligent, context-driven adjustments. For instance, it correctly deprioritized the "Network Behavior" dimension for alerts lacking network telemetry, even when MITRE guidance suggests it is important for the given technique. This demonstrates a crucial capability: aligning with expert knowledge while adapting to the available evidence.

Table 3 – Quantitative Scores Across Alerts and Models

Alert ID	ChatGPT-4.0	Grok 4 (Expert)	Claude Sonnet 4.0	Merged Score
1753761379	17	17	15	16.33
1753761350	19	11	12	14.00
1753752610	17	13	10	13.33
1753608159	17	14	16	15.67

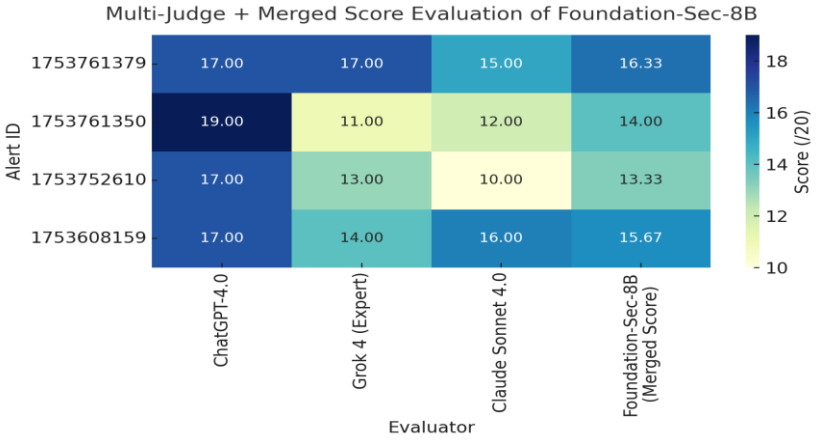

Fig. 4 : Performance scores assigned by the 3 independent LLM Judges

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

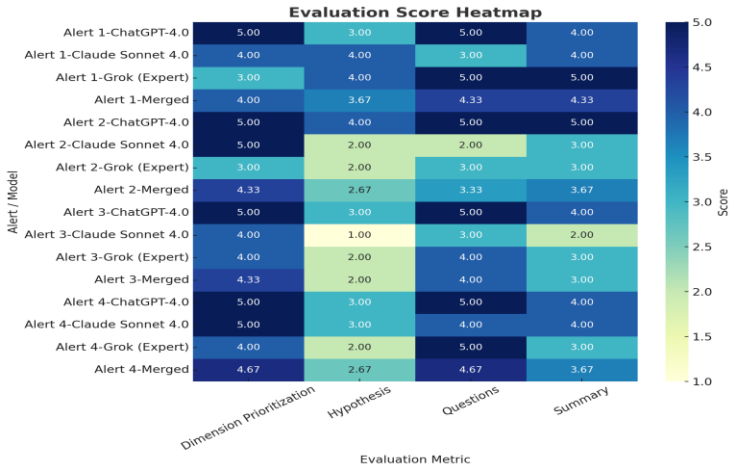

Fig. 5: Comparative Performance Analysis of AI Models (LLM as Judge) on SOC Alert Evaluation Task

A critical finding, however, is the significant variance in inter-judge agreement. For alert 1753761350, for example, ChatGPT-4.0 gave a near-perfect score of 19, while Grok and Claude rated it much lower at 11 and 12, respectively. To understand this variance, we performed a qualitative analysis of each judge's reasoning patterns.

Table 4: Key Reasoning Patterns across all alerts and judges

Model Judge	Observed Strengths	Observed Weaknesses / Analytical Bias
ChatGPT-4.0	Excellent IOC grounding and MITRE mapping.	Occasionally made unsupported claims about user intent.
Grok 4 (Expert)	Concise, logical, strong on persistence checks.	Prone to misinterpreting technical details (e.g., access rights).
Claude Sonnet 4.0	Well-structured, good coverage of all IOCs.	Used subjective language and attributed intent without proof.

This analysis reveals that each LLM judge applies its own distinct analytical lens, leading to different scores (Table 4). Grok, for instance, appears to penalize more heavily for technical inaccuracies, while ChatGPT-4.0 prioritizes alignment with the provided IOCs and MITRE framework.

Implications:

This experiment confirms that ALUSKORT generates high-quality reasoning outputs. More importantly, it highlights the value of using a multi-judge panel for evaluation, as it reveals the inherent subjectivity in assessing complex security analysis. The "malicious-first" stance of ALUSKORT, which encourages plausible hypotheses, may be scored as a strength by one judge (good for not missing threats) but as a weakness by another (making "unsupported claims"). This

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

variance is not a flaw in the method but a key finding in itself, reflecting the diverse perspectives found among human analysts in a real SOC.

4.5 Experiment 4: Prompt Architecture and Model Attention

Objective: To determine how the ordering of contextual blocks in a prompt affects a base LLM's focus during question generation.

Methodology: We developed a novel metric, the Critical Focus Score (CFS), which measures the percentage of the model's attention directed at the HYPOTHESIS and IOCs blocks at the first generated token. We compared four different prompt structures by varying the order of the context blocks.(The detailed methodology and mathematical formulation for CFS are provided in Appendix A).

Results: The results revealed a strong Recency Effect. Placing the most critical and variable information (HYPOTHESIS and IOCs) at the end of the prompt more than doubled the model's focus on them, achieving a CFS of 14.86% (Fig. 6). Conversely, placing them at the beginning caused attention to diffuse, dropping the CFS to just 5.09%. This finding provides strong empirical evidence that prompt architecture is a critical lever for steering the reasoning of base LLMs.

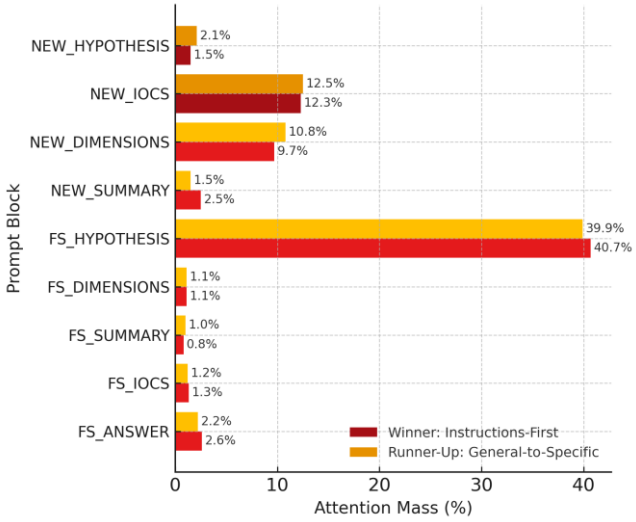

Fig. 6 : Attention Distribution by Prompt Structure (First Generated Token)

4.6 Experiment 5: Safety of Tool Mapping and Parameter Grounding

Objective: The reliability of any autonomous SOC workflow depends on its ability to interact with tools safely and accurately. This experiment was designed to empirically assess the safety and reliability of ALUSKORT's tool-calling mechanism, focusing on two primary goals:

1. Mapping investigative questions to correct, logical tools.
2. Ensuring that all parameters provided to those tools are factually grounded in verified alert data, preventing a common failure mode of LLM agents.

Methodology:

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

We assessed the 27 tool mappings generated across the four alerts against our pilot toolset, which consisted of six implemented tools focused on host state inspection (e.g., get_host_identity) and historical log investigation (e.g., trace_process_behavior). Each mapping was evaluated on two key criteria: Tool Availability (whether the chosen tool was part of our implemented set) and, most critically, Argument Grounding (whether every parameter was verifiably present in the validated IOCs from Phase 1).

Results: The grounding guardrail was 100% effective, with a Grounded Parameter Rate of 100% (Table 5). This is a critical safety finding, as it confirms the system completely prevents the hallucination of tool arguments. The Tool Availability Rate was 59.3%, with all misses due to tools that were designed but not yet implemented in the pilot phase, not incorrect mapping by the LLM.

Table 5: Tool mapping and Parameter Grounding

Alert ID	Total Mappings	Available Tools	Tool Availability (%)	Argument Grounding Rate (%)
1753761379	2	2	100	100
1753761350	10	6	60	100
1753752610	12	7	58.3	100
1753608159	3	1	33.3	100
Overall	27	16	59.3	100

Implications:

This evaluation confirms that ALUSKORT's guardrail-driven approach to tool interaction is fundamentally safe. As the toolset is expanded, the tool availability rate will naturally approach 100%, while the system is designed to maintain the perfect 100% grounding standard demonstrated in this pilot study.

4.7 Performance Profile: Latency and Token Efficiency

To provide a transparent and detailed view of the framework's operational footprint, we conducted a thorough performance analysis of the entire reasoning pipeline. The goal was to quantify the latency and token consumption of each phase, offering a clear benchmark for the system's resource requirements when running on commodity hardware.

The following table (Table 6) consolidates the average performance metrics for each phase of the ALUSKORT pipeline, based on the 5-shot configuration for IOC Extraction.

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Table 6: Performance Profile of ALUSKORT

Pipeline Phase	Avg. Inference Time (s)	Avg. Input Tokens	Avg. Output Tokens	Avg. Total Tokens	% of Total Latency
1. IOC Extraction (5-shot)	335.10	4,030	1,381	5,411	78.1%
2. Investigation Prioritization	46.94	2,624	173	2,797	10.9%
3. Hypothesis Generation	13.60	759	70	829	3.2%
4. Question Generation	33.14	1,538	146	1,684	7.7%
Total Pipeline	428.78	8,951	1,770	10,721	100.0%

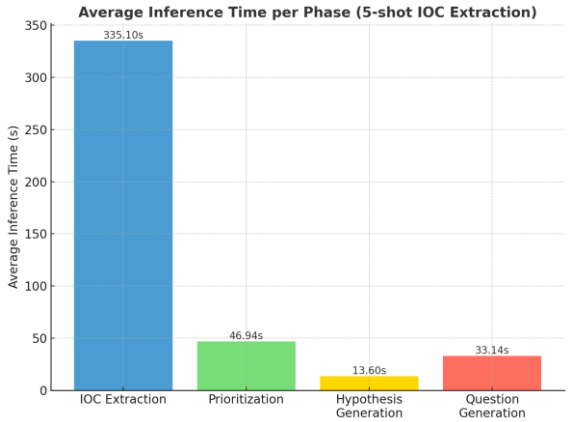

Fig. 7: Inference Time Distribution across ALUSKORT phases

Analysis:

This detailed breakdown reveals two primary findings:

1. Operational Footprint: The entire reasoning pipeline completes in an average of 428.78 seconds (~7 minutes) and consumes an average of 10,721 total tokens. This provides a clear benchmark for the system's resource consumption.

2. Latency Distribution: The analysis pinpoints a significant performance bottleneck in the IOC Extraction phase, which accounts for 78.1% of the total pipeline latency.

Implications:

The key implication of this data is the identification of the token-heavy IOC Extraction phase as the primary performance bottleneck (Fig. 7). ALUSKORT's deterministic g_{clean} function (Phase 0) is a direct architectural mitigation for this, reducing input token counts before they reach the LLM. However, this highlights a clear area for future work: re-architecting this phase with more advanced techniques, such as fine-tuning or RAG, to further improve scalability and speed.

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Evaluation Scope and Limitations:

It is important to note the defined scope of this pilot study. To isolate and accurately measure the framework's intrinsic cognitive performance, our evaluation focused exclusively on the tool-agnostic LLM reasoning phases: IOC Extraction, Dimension Prioritization, Hypothesis Generation, and Question Generation.

The final phases involving tool execution and evidence synthesis were deliberately excluded from the latency benchmark. This is because their performance is inherently influenced by the responsiveness and availability of external, deployment-specific tools (e.g., SIEM, EDR APIs). Including them would conflate the model's reasoning time with external I/O latency, preventing a fair and reproducible measurement of ALUSKORT's cognitive performance across different environments.

Future system-level studies will benchmark the full end-to-end latency under controlled conditions with a fixed toolset, enabling a reproducible measurement of its true operational performance.

5 Discussion

The primary contribution of this work is the empirical demonstration that a complete, multi-phase security investigation can be effectively automated within a resource-constrained environment. Our results show that a smaller, open-source base model—in this case, an 8-bit quantized version of the 8B parameter Foundation-Sec-8B—can be successfully steered by an agentic framework to perform sophisticated reasoning tasks on commodity hardware, such as a single GPU in a Kaggle environment.

The success of this approach hinges on the synergy between the model's domain-specific pretraining and the framework's structured design. While the base model provides the foundational cybersecurity knowledge, the ALUSKORT pipeline provides the necessary scaffolding to guide its reasoning. The phase-based progression, from deterministic cleaning in Phase 0 to evidence-backed synthesis in the final stages, breaks down the complex problem of incident investigation into a sequence of manageable, well-defined tasks. This structured workflow is what enables a smaller model to achieve a high degree of logical coherence and contextual awareness without the need for massive computational resources.

Furthermore, this study serves as a practical guide for making such systems operationally safe. The 100% grounding rate for tool parameters is a critical finding, proving that deterministic guardrails are a highly effective method for preventing LLM hallucination in safety-critical actions. This is particularly important when using base models, which are not explicitly instruction-tuned for safety. ALUSKORT's "correction-over-rejection" policy and post-hoc validation checks provide a robust and auditable layer of control, ensuring that every automated step is anchored to verifiable data from the source alert.

In essence, ALUSKORT provides a reproducible blueprint for harnessing the power of domain-specific open-source LLMs. It proves that by combining a thoughtful agentic architecture with strict, rule-based guardrails, it is possible to build an effective and trustworthy autonomous investigation system that is both powerful and accessible, directly addressing the critical need for scalable, efficient solutions in modern Security Operations Centers.

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

6 Limitations

While this pilot study successfully demonstrates the feasibility and core capabilities of the ALUSKORT framework, it is important to acknowledge its limitations to provide context for the results.

1. **Limited Dataset:** The evaluation was conducted on a small, focused dataset of four Wazuh alerts. While these alerts were chosen to be representative of common threat techniques, this small sample size limits the statistical significance and generalizability of our findings. A larger-scale study is required to validate the framework's robustness across a wider variety of alert types and sources.

2. **Constrained Evaluation Scope:** Our performance analysis deliberately focused on the tool-agnostic LLM reasoning phases to isolate the cognitive performance of the framework. As a result, the full end-to-end latency, including external tool execution and final evidence synthesis, was not measured.

3. **Pilot-Phase Toolset:** The evaluation of tool mapping was performed against a limited library of six implemented tools. Consequently, the "Tool Availability Rate" metric reflects the scope of our pilot implementation rather than a limitation of the LLM's logical tool selection capabilities.

4. **Proposed Architectural Features:** Several advanced features of the ALUSKORT architecture, such as the ECS mapping layer for portability and the deterministic MITRE technique resolver for alerts lacking metadata, were designed but not implemented in this pilot study.

7 Future Work

The findings and limitations of this study lay a clear foundation for future research and development. Our planned next steps are focused on three key areas:

1. Scaling and Validation: The immediate priority is to conduct a large-scale evaluation of ALUSKORT on a diverse dataset of hundreds or thousands of alerts from various sources (e.g., EDRs, cloud security platforms). This will include a full system-level benchmark to measure true end-to-end operational latency and a direct comparison of the system's output against the findings of human SOC analysts to quantify its real-world value.

2. Architectural Enhancements:

a. **Optimizing the IOC Extraction Bottleneck:** Guided by our performance analysis, we will re-architect the token-heavy IOC extraction phase. This will involve exploring advanced techniques such as fine-tuning a smaller, specialized extraction model or implementing a Retrieval-Augmented Generation (RAG) approach to dynamically select the most relevant few-shot examples.

b. **Expanding the Tool Library:** We will significantly expand the library of integrated tools to cover threat intelligence lookups, vulnerability scanning, user behavior analysis, and automated remediation actions.

c. **Implementing Proposed Features:** We will implement the proposed ECS mapping layer and the MITRE technique inference module to enhance the framework's portability and robustness.

3. Advanced Agentic Capabilities: We plan to introduce a long-term memory component, likely through integration with a case management system or a vector database. This would allow ALUSKORT to reason across multiple related alerts, identify broader attack campaigns, and learn from the outcomes of past investigations, moving it closer to a truly autonomous security reasoning platform.

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

8 Conclusion

This paper introduced ALUSKORT, a hierarchical multi-agent framework that provides an empirical demonstration that a complete, multi-phase security investigation can be effectively and safely automated within a resource-constrained environment. Our work proves that by combining a thoughtful agentic architecture with strict, rule-based guardrails, it is possible to steer a smaller, domain-specific open-source model to perform sophisticated reasoning tasks on commodity hardware.

Our comprehensive evaluation confirmed that the framework produces high-quality, context-aware investigative artifacts and, most critically, that its design guarantees the factual grounding of all safety-critical tool interactions. The detailed performance analysis not only establishes the feasibility of this approach but also provides actionable insights for future optimization.

Ultimately, ALUSKORT serves as a reproducible blueprint for harnessing the power of accessible AI. It proves that it is possible to build effective, trustworthy, and scalable autonomous investigation systems, offering a clear path forward in addressing the critical challenges of alert fatigue and analyst burnout in modern Security Operations Centers.

Appendix A: Methodology for Critical Focus Score (CFS) Calculation

References

[1] Divakaran, Dinil Mon, and Sai Teja Peddinti. "LLMs for cyber security: New opportunities." arXiv preprint arXiv:2404.11338 (2024).

[2] Kassianik, Paul, Baturay Saglam, Alexander Chen, Blaine Nelson, Anu Vellore, Massimo Aufiero, Fraser Burch et al. "Llama-3.1-foundationai-securityllm-base-8b technical report." arXiv preprint arXiv:2504.21039 (2025).

[3] Amin Karbasi & Foundation AI – Cisco. (2025, April 28). *Foundation-Sec-8B* [Model]. Hugging Face. https://huggingface.co/fdtn-ai/Foundation-Sec-8B

[4] M. Vielberth, F. Böhm, I. Fichtinger and G. Pernul, "Security Operations Center: A Systematic Study and Open Challenges," in IEEE Access, vol. 8, pp. 227756-227779, 2020, doi: 10.1109/ACCESS.2020.3045514.keywords:{Security; Databases; Libraries; Computer security;Computer architecture;Technological innovation;Systematics;Security management;security operations center;security operations;SOC},

[5] Roshanaei, Maryam, Mahir R. Khan, and Natalie N. Sylvester. "Enhancing cybersecurity through AI and ML: Strategies, challenges, and future directions." Journal of Information Security 15, no. 3 (2024): 320-339.

[6] Binbeshr, Farid, Muhammad Imam, Mustafa Ghaleb, Mosab Hamdan, Mussadiq Abdul Rahim, and Mohammad Hammoudeh. "The Rise of Cognitive SOCs: A Systematic Literature Review on AI Approaches." IEEE Open Journal of the Computer Society (2025).

[7] Tariq, Shahroz, Mohan Baruwal Chhetri, Surya Nepal, and Cecile Paris. "Alert fatigue in security operations centres: Research challenges and opportunities." ACM Computing Surveys 57, no. 9 (2025): 1-38

.[8] Alahmadi, Bushra A., Louise Axon, and Ivan Martinovic. "99% false positives: A qualitative study of {SOC} analysts' perspectives on security alarms." In 31st USENIX Security Symposium (USENIX Security 22), pp. 2783-2800. 2022.

[9] Yang, Limin, Zhi Chen, Chenkai Wang, Zhenning Zhang, Sushruth Booma, Phuong Cao,

Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Constantin Adam et al. "True attacks, attack attempts, or benign triggers? an empirical measurement of network alerts in a security operations center." In 33rd USENIX Security Symposium (USENIX Security 24), pp. 1525-1542. 2024.

[10] Song, Chengyu, Linru Ma, Jianming Zheng, Jinzhi Liao, Hongyu Kuang, and Lin Yang. "Audit-llm: Multi-agent collaboration for log-based insider threat detection." arXiv preprint arXiv:2408.08902 (2024).

[11] Xu, HanXiang, ShenAo Wang, Ningke Li, Kailong Wang, Yanjie Zhao, Kai Chen, Ting Yu, Yang Liu, and HaoYu Wang. "Large language models for cyber security: A systematic literature review." arXiv preprint arXiv:2405.04760 (2024).

[12] Sun, Danyu, Jinghuai Zhang, Jiacen Xu, Yu Zheng, Yuan Tian, and Zhou Li. "From Alerts to Intelligence: A Novel LLM-Aided Framework for Host-based Intrusion Detection." arXiv preprint arXiv:2507.10873 (2025).

[13] Zhang, Jie, Haoyu Bu, Hui Wen, Yongji Liu, Haiqiang Fei, Rongrong Xi, Lun Li, Yun Yang, Hongsong Zhu, and Dan Meng. "When llms meet cybersecurity: A systematic literature review." Cybersecurity 8, no. 1 (2025): 55.

[14] Tonmoy, S. M., S. M. Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. "A comprehensive survey of hallucination mitigation techniques in large language models." arXiv preprint arXiv:2401.01313 6 (2024).

[15] Wei, Jason, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V. Le, and Denny Zhou. "Chain-of-thought prompting elicits reasoning in large language models." Advances in neural information processing systems 35 (2022): 24824-24837.

[16] bKshetri, Nir. "Transforming cybersecurity with agentic AI to combat emerging cyber threats." Telecommunications Policy (2025): 102976.

[17] Ismail, Rahmat Kurnia, Zilmas Arjuna Brata, Ghitha Afina Nelistiani, Shinwook Heo, Hyeongon Kim, and Howon Kim. "Toward Robust Security Orchestration and Automated Response in Security Operations Centers with a Hyper-Automation Approach Using Agentic Artificial Intelligence." Information 16, no. 5 (2025): 365.

[18] Freitas, Scott, Jovan Kalajdjieski, Amir Gharib, and Robert McCann. "AI-driven guided response for security operation centers with Microsoft Copilot for Security." In Companion Proceedings of the ACM on Web Conference 2025, pp. 191-200. 2025.

[19] Wang, Junlin, Jue Wang, Ben Athiwaratkun, Ce Zhang, and James Zou. "Mixture-of-agents enhances large language model capabilities." arXiv preprint arXiv:2406.04692 (2024).

[20] Kong, He, Die Hu, Jingguo Ge, Liangxiong Li, Tong Li, and Bingzhen Wu. "Vulnbot: Autonomous penetration testing for a multi-agent collaborative framework." arXiv preprint arXiv:2501.13411 (2025).

[21] Zhu, Yuxuan, Antony Kellermann, Akul Gupta, Philip Li, Richard Fang, Rohan Bindu, and Daniel Kang. "Teams of llm agents can exploit zero-day vulnerabilities." arXiv preprint arXiv:2406.01637 (2024).

[22] Elastic. (n.d.). *ECS reference*. Elastic. Retrieved May,20, 2025, from https://www.elastic.co/docs/reference/ecs

[23] "MITRE ATT&CK®," MITRE, [Online]. Available: https://attack.mitre.org/. [Accessed: July16, 2025].

[24] Gu, J., Jiang, X., Shi, Z., Tan, H., Zhai, X., Xu, C., Li, W., Shen, Y., Ma, S., Liu, H., et al. A

Received: August 06, 2025

Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

survey on llm-as-a-judge. arXiv preprint arXiv:2411.15594 (2024).

[25] Guo, T., Chen, X., Wang, Y., Chang, R., Pei, S., Chawla, N. V., Wiest, O., and Zhang, X. Large language model based multi-agents: A survey of progress and challenges. arXiv preprint arXiv:2402.01680 (2024).

[26] Tseng, PeiYu, ZihDwo Yeh, Xushu Dai, and Peng Liu. "Using llms to automate threat intelligence analysis workflows in security operation centers." arXiv preprint arXiv:2407.13093 (2024).

[27] Kremer, R., P. N. Wudali, S. Momiyama, T. Araki, J. Furukawa, Y. Elovici, and A. Shabtai. "IC-SECURE: Intelligent System for Assisting Security Experts in Generating Playbooks for Automated Incident Response. arXiv 2023." arXiv preprint cs.CR/2311.03825.

[28] Kaheh, Mehrdad, Danial Khosh Kholgh, and Panos Kostakos. "Cyber sentinel: Exploring conversational agents in streamlining security tasks with gpt-4." arXiv preprint arXiv:2309.16422 (2023).

Appendix A: Methodology for Critical Focus Score (CFS) Calculation

This appendix details the procedure for calculating the Critical Focus Score (CFS), a metric designed to measure a base model's attention on the most critical entities of a new, unseen alert during question generation.

A.1 Prompt Structures

We designed a controlled experiment to compare four distinct prompt structures:

A: General-to-Specific: HYPOTHESIS → DIMENSIONS → SUMMARY → IOCS

B: Evidence-First: IOCS → HYPOTHESIS → DIMENSIONS → SUMMARY

C: Instructions-First: DIMENSIONS → SUMMARY → HYPOTHESIS → IOCS

D: Balanced: HYPOTHESIS → IOCS → DIMENSIONS → SUMMARY

Calculation Procedure:

Step 1: Attention Extraction — The prompt is processed with output_attentions=True, capturing token-level attention weights from the last transformer layer for the first generated token.

Step 2: Averaging Across Heads — Attention values for each prompt token are averaged across all attention heads to obtain a single weight per token:

$$\overline{A}_{t,j} = \frac{1}{n_h} \sum_{h=1}^{n_h} A_{l,h,t,j}$$

Where:

- $\overline{A}_{t,j}$ = averaged attention weight for token j at generation step t
- n_h = number of attention heads
- $A_{l,h,t,j}$ = attention weight from layer l, head h, generation step t, to token j

Step 3: Token-to-Block Mapping — Using character offsets, each token is mapped to its originating prompt block (e.g., NEW_HYPOTHESIS, NEW_IOCS, DIMENSIONS, SUMMARY).

Step 4: Block-Level Attention Mass — The attention mass for block B is:

Received: August 06, 2025

International Journal of Applied Mathematics
Volume 38 No. 5s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$$M_B = \sum_{j \in B} \overline{A}_{t,j}$$

Where $j \in B$ represents all tokens belonging to block B.

Step 5: Total Attention Mass —

$$M_{\text{total}} = \sum_{B} M_B$$

Step 6: Normalized Attention Percentage for Block B —

$$P_B = \frac{M_B}{M_{\text{total}}} \times 100$$

Where P_B represents the percentage of total attention allocated to block B.

Step 7: Critical Focus Score — The final CFS is the sum of attention percentages for the two critical blocks:

$$\text{CFS} = P_{\text{NEW\_HYPOTHESIS}} + P_{\text{NEW\_IOCS}}$$

Design Assumption

This metric assumes that, for investigative question generation in SOC workflows, **the analyst's hypothesis and IOCs are the most valuable context**.

Received: August 06, 2025