Computer Network Security Report Generation with RAG and Small Language Models

Diego das Neves Nascimento,[1] **Tiago Buarque Assunção de Carvalho,**
[1]**Sérgio Francisco Tavares de Oliveira Mendonça**[1]

[1] Universidade Federal do Agreste de Pernambuco, Pernambuco, Brasil.

{diego.nascimento,tiago.buarque,sergio.mendonca}@ufape.edu.br

Abstract. *Computer network monitoring is a crucial activity for institutional security; however, many organizations face a shortage of specialists to operationalize this infrastructure. To mitigate this issue, this paper presents a proof of concept of a Retrieval-Augmented Generation (RAG) pipeline executed 100% locally on CPU for incident detection and mitigation. The system utilizes Small Language Models (SLMs) for structured report generation. Experimental results demonstrate the technical viability of the approach under severe hardware constraints. The pipeline achieved 100% MITRE technique identification within the evaluated attack categories and 70% precision in extracting the corresponding defensive safeguards.*

1. Introduction

Computer network security and monitoring require specialized personnel. However, institutions do not always have such human resources available. In this work, we propose the use of SLMs, running locally, to assist generalist information technology technicians in network monitoring tasks. We consider an SLM to be a model with up to 7 billion parameters. Recent systematic reviews demonstrate that while compact models favor local privacy, they still face major trustworthiness challenges. Consequently, SLMs frequently generate incorrect or inconsistent statements, especially in tasks requiring deep knowledge or in open-ended dialogues. To mitigate this accuracy degradation, the adoption of RAG and grounding in external knowledge bases emerge as essential architectural solutions [Corradini et al. 2025]. The injection of pertinent facts from an external *corpus* into the model's context, as well as integration with structured databases for real-time fact-checking, allows statements to be verified before final generation, helping SLMs overcome their parametric knowledge limitations.

In the context of this work's target application, the security of government infrastructures imposes severe operational and regulatory restrictions: privacy guidelines such as the *Lei Geral de Proteção de Dados* (LGPD, a law for data protection) and the *Programa de Privacidade e Segurança da Informação* [Brasil 2018] (PPSI, a federal program for privacy and information security) [Brasil 2025] prohibit the exposure of data and network traffic to third-party cloud providers, requiring incident analysis to be conducted through strictly local processing. Within *Instituições Federais de Educação Superior* (IFES, higher education federal institutions), this regulatory scenario collides with a critical operational barrier: a severe shortage of specialized IT personnel, with deficits exceeding 50% of the ideal workforce [Brasil 2026]. Given the high volume of alerts generated by detection

systems and the absence of specialized analysts to handle them, automating network incident triage becomes an absolute necessity. However, for this automation to be useful for generalist IT teams, it must be rigorously accurate.

Another requirement is feasibility on legacy hardware (frequently restricted to CPU-only processing). This requires that generated reports produce outputs grounded in recognized knowledge bases, avoiding hallucinations and ensuring the traceability needed to accelerate incident response. Recent literature in critical government domains demonstrates that external context injection via RAG is fundamental to suppress hallucinations and ensuring technical validity in compact models [Freitas et al. 2024, Novais et al. 2024], while local execution enables compliance with the LGPD and legacy infrastructures [Zitei et al. 2024, Rocha and Pessoa 2024]. In the context of cybersecurity, prior alert classification via Machine Learning (ML) has demonstrated high robustness even on restricted hardware [da Silva Júnior et al. 2024, dos Santos Almeida et al. 2025], enabling the automation of Network Intrusion Detection System (NIDS) logs.

However, integrating these solutions into a single *pipeline* for grounded report generation remains a gap. To fill it, this article proposes and validates a Proof of Concept (PoC) of a 100% CPU-local RAG *pipeline*. The *pipeline* integrates network log classification, hybrid retrieval (semantic and lexical), and structured extraction grounded in two complementary knowledge bases — MITRE ATT&CK (a knowledge base with hierarchical standardized identifiers for offensive tactics) and CIS Controls v8 (a defensive knowledge base with 153 safeguard categories) — consolidating the result into an incident report. The remainder of this article is organized as follows: Section 2 presents related work. Section 3 details the architecture of the proposed pipeline. Section 4 covers the experimental methodology and results. Finally, Section 5 concludes the work.

2. Related Work

This section positions our proposal against state-of-the-art architectural choices. Following a search across IEEE Xplore, ACM Digital Library, and Scopus (2020–2025), we filtered the retrieved works through the lens of design constraints and technical grounding. By classifying existing solutions into two distinct groups based on their architectural choices in knowledge grounding and execution modes, we expose a critical gap that our proposal directly addresses.

The first group comprises approaches that employ RAG architectures grounded in knowledge bases but rely on cloud infrastructure or specialized hardware, thereby violating the constraint of local execution on restricted hardware. Perrina et al. utilize knowledge bases for report generation but depend on cloud infrastructure [Perrina et al. 2023]. Similarly, Paduraru et al. generate structured *playbooks* but require 24 GB GPUs [Paduraru et al. 2025]. Additionally, Daniel et al. leverage formal knowledge bases but lack feasibility for local execution [Daniel et al. 2025]. Collectively, these approaches prioritize semantic structure yet fail to meet the constraint of local processing on limited hardware (CPU-only).

The second group encompasses ungrounded local inference architectures [Barros et al. 2025, Melhem et al. 2025], which deploy LLMs on CPUs without an external *schema* for verification and remain prone to hallucinations. Similarly, the local inference approach by Jones et al. [Jones et al. 2025] requires manual intervention and

lacks automated grounding. Furthermore, Jüttner et al. [Jüttner et al. 2024] and Houssel et al. [Houssel et al. 2024] map log alerts to countermeasures without a structured reference. Ultimately, while these frameworks enable edge or local execution, they cannot guarantee data traceability or rigorous knowledge grounding.

Table 1 consolidates this architectural comparison, illustrating that no existing solution concurrently integrates knowledge base grounding with strictly local CPU execution and automated log analysis. To the best of our knowledge, our proposal is the first to combine all these dimensions, delivering both automated log analysis and structured incident reporting. The column headers are defined as follows: [**Logs**] Log Analysis; [**Knowl.**] Grounding in Structured Knowledge Base; [**Report**] Structured Report Output; [**CPU**] CPU-only Execution; [**Local**] Strictly Local Processing.

Tabela 1. Architectural comparison of related works. ✓ = satisfies; × = does not satisfy.

Work	Logs	Knowl.	Report	CPU	Local
[Perrina et al. 2023]	✓	✓	✓	×	×
[Jüttner et al. 2024]	✓	×	✓	×	×
[Houssel et al. 2024]	×	×	✓	×	×
[Jones et al. 2025]	×	×	✓	×	✓
[Barros et al. 2025]	✓	×	×	✓	✓
[Paduraru et al. 2025]	✓	✓	✓	×	×
[Melhem et al. 2025]	✓	×	×	✓	×
[Daniel et al. 2025]	✓	✓	×	×	×
Proposal (RAG-S)	✓	✓	✓	✓	✓

3. Methodology

The complete system pipeline is outlined in Figure 1. The execution flow starts with network traffic capture (1), classified via Random Forest (2) into structured JSON alerts (3). These alerts feed a local RAG core, which integrates the MITRE ATT&CK (4) and CIS Controls v8 (5) knowledge bases, a hybrid retrieval mechanism (6), and an SLM extraction module (7). Finally, the extracted grounded IDs (8) are sent to the report generation module (9) to produce the final report (10). Subsequent subsections detail each component: Classification Module (3.1), knowledge bases (3.2), hybrid retrieval (3.3), prompt design (3.4), proposed pipeline variants (3.5), language models (3.6), and evaluation metrics (3.7).

3.1. Classification Module and Alert Construction

The initial phase of the architecture monitors network traffic (item 1 in Figure 1). To ensure experimental replicability, we employ the publicly available CICIDS2017 (*Canadian Institute for Cybersecurity Intrusion Detection System*) benchmark dataset [Sharafaldin et al. 2018].

Subsequently, the dataset logs are processed by a Random Forest (RF) classifier (item 2 in Figure 1). This model is trained on statistical network features, utilizing hyperparameters and a stratified split that strictly replicate the methodology established by Khan et al. [Khan et al. 2024]. The RF model predicts the corresponding attack macro-category, specifically: `Bot`, `DDoS`, `PortScan`, `Brute Force`, or `Web Attack`. For

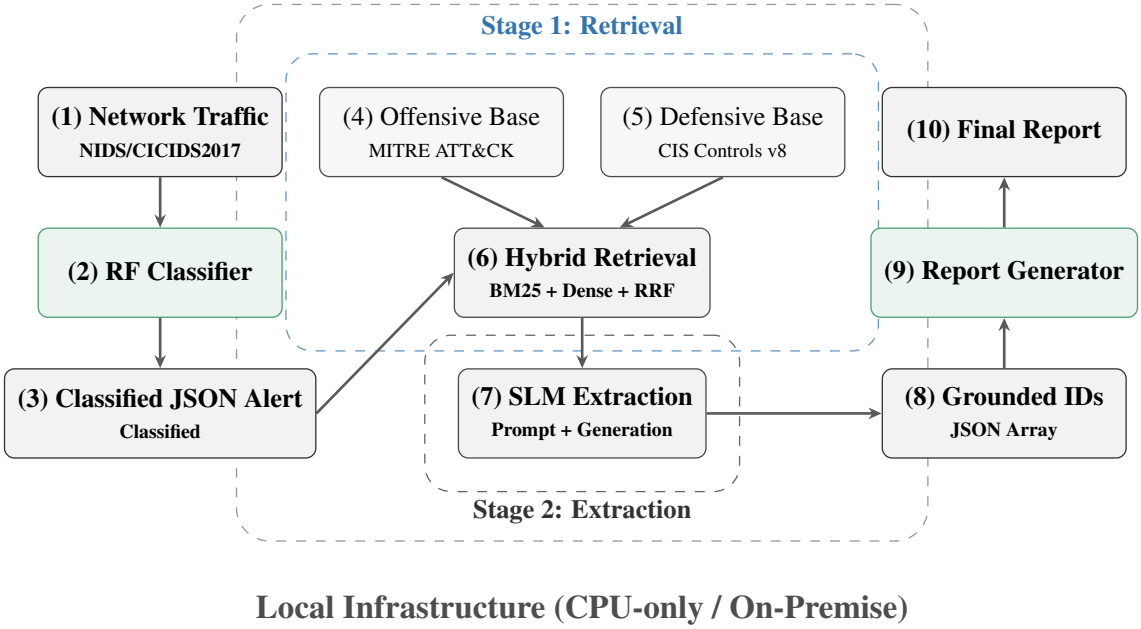


Figura 1. Architectural overview of the dual-grounded local RAG pipeline.

evaluation purposes, we extract a balanced test sample consisting of 30 alerts per class, totaling 150 evaluation instances distributed evenly across all attack vectors.

To interface with the downstream RAG engine, the classifier outputs are formalized as structured JSON alerts (item 3 in Figure 1). Each JSON object encapsulates the RF-predicted attack type alongside foundational network log attributes, including source and destination IP addresses, source and destination ports, and transport protocols.

3.2. Knowledge Base Preprocessing

MITRE ATT&CK is a knowledge base with hierarchical standardized identifiers for offensive tactics. The pipeline employs MITRE ATT&CK, which comprises over 500 techniques grouped into 11 attack tactics. For this work, 10 specific techniques were selected—allocating 2 techniques per attack macro-category described in Section 3.1. This extraction yields 10 distinct knowledge chunks, each consisting of a standardized identifier (ID), tactic, technique name, and attack description (item 4 in Figure 1).

Concurrently, the CIS Controls v8 framework serves as our defensive knowledge base, comprising 153 granular safeguards organized into 18 security domains. We preprocess this corpus by generating one standalone textual chunk per individual safeguard, resulting in a total of 153 defensive chunks (item 5 in Figure 1). While the MITRE ATT&CK chunks are utilized downstream to contextualize and classify the attack profile, the CIS Controls chunks provide localized, actionable remediation strategies tailored to each identified threat vectors.

3.3. Hybrid BM25 + Dense Retrieval via RRF

The text chunks described in Section 3.2 were encoded and indexed in a vector database to support both semantic (dense) and lexical (BM25) retrieval. For dense vector generation, we utilized the `BAAI/bge-large-en-v1.5` embedding model. To optimize retrieval alignment, an asymmetric search query prefix was applied during encoding. This process mapped each chunk into a 1024-dimensional vector space. Crucially, each vector was

enriched with metadata tags identifying its source repository—either MITRE ATT&CK (offensive) or CIS Controls (defensive). Finally, the resulting embeddings (semantic) and lexical tokens were indexed concurrently within ChromaDB.

Utilizing the classified JSON alert from Section 3.1, the pipeline triggers dual independent queries (item 6 in Figure 1). The **offensive retrieval** branch queries the MITRE ATT&CK corpus using dense semantic search to extract the single most relevant technique (Top-1). Concurrently, the **defensive retrieval** branch queries the CIS Controls v8 database via a hybrid BM25 and dense search framework. These lexical and semantic scores are consolidated using Reciprocal Rank Fusion (RRF) to retrieve a candidate pool of 10 defensive safeguards, In the defensive retrieval branch, a preliminary pool of 20 candidate chunks is extracted, comprising 10 chunks retrieved via BM25 and 10 chunks from dense search. These candidate sets are consolidated into a single unified rank list using RRF, formalized as:

$$\text{RRF_Score}(d \in D) = \sum_{r \in R} \frac{1}{k + \text{rank}_r(d)} \tag{1}$$

where $R = \{\text{BM25}, \text{Dense}\}$ denotes the set of constituent retrieval rankings, $\text{rank}_r(d)$ represents the position of document d within ranking r, and $k = 60$ is the standard smoothing constant established by Cormack et al. [Cormack et al. 2009]. To mitigate structural redundancy and ensure broad thematic coverage within the final prompt context, the resulting top-20 unified ranking is subjected to a diversity filter. This constraint restricts inclusion to a maximum of 2 documents per individual CIS domain, ultimately yielding a diversified subset of 10 defensive chunks (Top-10).

3.4. Prompt Design and Report Generation

The downstream stage of the pipeline orchestrates a two-tier language model execution flow. The initial SLM invocation (item 7 in Figure 1) initializes the agent with the system role: `"You are a Security Operations (SecOps) analyst."` The user prompt explicitly encapsulates the input features using structured XML tags (`<json_alert>`, `<mitre_context>`, and `<cis_context>`). To ensure deterministic schema adherence and downstream error traceability, the model's output is structurally constrained to a valid JSON array consisting of exactly 3 strings (item 8 in Figure 1). This strict formatting isolates distinct context sources. Parsing this intermediate array triggers a secondary language model call (item 9 in Figure 1) that synthesizes the technical findings into a cohesive narrative. This process generates the final incident report (item 10 in Figure 1), which details the identified vulnerabilities, system impact metrics, and actionable mitigations (as exemplified in Figure 2).

3.5. Proposed Pipeline Variants

To systematically evaluate the central hypothesis of this work, we conduct an ablation study comparing five structural pipeline configurations. These variants differ exclusively in their retrieval dependencies and query formulation strategies:

- **RAG-S (Sequential):** In this configuration, defensive retrieval (Section 3.3) is directly conditioned on offensive retrieval outputs. The complete MITRE chunk extracted in

Stage 1 (item 4 in Figure 1) enriches the subsequent CIS query formulation (item 5), thereby guiding defensive selection with contextual offensive threat intelligence. We hypothesize that this directional enrichment optimizes safeguard relevance, yielding measurable performance gains in nDCG@10_CIS and Precision. Prompt XML tags: `<json_alert>`, `<mitre_context>`, and `<cis_context_enriched>`. It is the only approach that uses `<cis_context_enriched>`.

- **RAG-P (Parallel):** Offensive and defensive retrieval (Section 3.3) operations are decoupled and execute independently, with both branches utilizing the original JSON alert (items 3) as their primary search query. While items 4 and 5 are both executed, the selection of defensive context remains uninfluenced by offensive classification results. Prompt XML tags: `<json_alert>`, `<mitre_context>`, and `<cis_context>`. This approach differs from RAG-S only by the tag `<cis_context>`.
- **RAG-C (CIS Only):** This variant bypasses the MITRE pipeline entirely (item 4), providing the model exclusively with retrieved defensive context. Prompt XML tags: `<json_alert>` and `<cis_context>`. This approach differs from RAG-P by ommitting `<mitre_context>` tag.
- **RAG-M (MITRE Only):** This variant isolates the offensive pipeline, providing the extraction engine solely with retrieved adversary technique context. Accordingly, the CIS defensive retrieval phase (item 5) is deactivated. Prompt XML tags: `<json_alert>` and `<mitre_context>`. This approach differs from RAG-P by omitting the `<cis_context>` tag.
- **ZS (Zero-Shot):** This baseline completely omits external knowledge retrieval. The language model relies exclusively on its parametric internal knowledge to infer and extract relevant MITRE and CIS identifiers from the raw JSON alert data. Retrieval layers (items 4, 5, and 6) are bypassed, routing the data flow directly from item 3 to the extraction layer (item 7). Prompt XML tag: `<json_alert>`. The tag `<json_alert>` is the same on the four approches.

3.6. Language Models and Inference Environment

Three lightweight language models were evaluated using the local LM Studio REST API:

- `bartowski/Llama-3.2-3B-Instruct-GGUF`, (3 Billion parameters)
- `google/gemma-4-e2b-it`, (2.3B effective parameters, 5.1B with embeddings)
- `Smoffyy/Qwen3.5-4B-Instruct-Revised-GGUF`, (4 Billion parameters)

These models serve as representative baselines for the state of the art in SLMs within the 2B–4B parameter range. This selection provides architectural diversity—contrasting models with specialized attention optimizations against dense baseline architectures—thereby enabling a comparative assessment of how varying parametric capacities handle rigid JSON schema compliance under strict hardware limitations. To align with the targeted design constraints, a strict CPU-only execution environment was enforced.[1] Generation hyperparameters were locked across all evaluation runs to minimize variance: temperature was set to 0.15, *max_tokens* was capped at 512, and the random

[1]Experimental environment: Windows 11, Python 3.10.12, Intel Core i5-1135G7 CPU, and 16 GB RAM. Software stack: LM Studio REST API, `rank_bm25`, `chromadb`, and `sentence-transformers` running `BAAI/bge-large-en-v1.5`. Source code repository available at: https://github.com/diego-ufape/Pipeline-RAG-Local.

seed was fixed to 42 to ensure absolute reproducibility.

3.7. Evaluation Metrics

Evaluation metrics are categorized into two core pipeline stages (Table 2): retrieval and extraction. Retrieval metrics evaluate the quality of the operational contexts fetched during the search phase (item 6 in Figure 1), whereas extraction metrics assess the structured output produced by the downstream SLM (item 8). The experimental ground truth (GT) is derived from the official cross-framework CIS Controls v8 $\leftrightarrow$ MITRE ATT&CK mapping documentation [CIS 2022].

Retrieval Stage (item 6, Figure 1):

- Hit@1_MITRE: A binary indicator scored as 1 if the exact ground-truth MITRE attack technique is successfully retrieved, and 0 otherwise.
- Hit@10_CIS: A binary indicator scored as 1 if at least one valid CIS safeguard is present within the top-10 retrieved candidates, and 0 otherwise.
- Hit@K: Represented broadly as Hit@1_MITRE or Hit@10_CIS, depending on the target infrastructure domain.
- Recall@10: Set-based recall evaluated over the CIS domain, formalized as:

$$\text{Recall@10} = \frac{|\mathcal{T}_{10} \cap \text{GT}|}{|\text{GT}|} \tag{2}$$

  where $\mathcal{T}_{10}$ denotes the set of the top-10 retrieved safeguards and GT (ground truth) is the full set of correct baseline countermeasures [CIS 2022]. Because |GT| may naturally exceed 10, Recall@10 is bounded and may not always reach 100%.
- nDCG@10_CIS: Normalized Discounted Cumulative Gain (nDCG) computed over the top-10 CIS ranking, where higher magnitudes indicate that valid safeguards are prioritized at higher ordinal ranks.

Extraction Stage (item 7, Figure 1):

- EM (Exact Match): A binary indicator scored as 1 if the SLM extracts the precise alphanumeric MITRE technique identifier matching the target baseline, and 0 otherwise.
- Precision: Set-based precision evaluated over the CIS domain, computed as

$$Precision = \frac{|\text{Pred} \cap \text{GT}|}{|\text{Pred}|}. \tag{3}$$

  Where Pred is the set of two CIS ids and GT as brefore. This metric yields discrete values of 1.0 if both predicted safeguards are correct, 0.5 if a single safeguard is correct, and 0 if both are incorrect.
- PipeScore: The cumulative cross-stage pipeline success score, computed as the product EM $\times$ Precision, yielding discrete outcomes within the bounded set $\{0, 0.5, 1\}$.

4. Experiments and Results

4.1. Retrieval

According to the empirical results in Table 3, Hit@K for MITRE consistently achieves 100%. For CIS, Hit@K reaches 100% only under RRF fusion (Section 3.3). Retrieval

Tabela 2. Evaluation metrics by pipeline stage.

Stage	Metric	Definition	Type
Retrieval	Hit@k	Relevant MITRE ID in Top-1 or relevant CIS in Top-10	Binary $\in \{0, 1\}$
	nDCG@10_CIS	nDCG over CIS Top-10 ranking	Continuous $\in [0, 1]$
	Recall@10	Set-based recall over CIS	Continuous $\in [0, 1]$
Extraction	EM	MITRE ID identical to GT	Binary $\in \{0, 1\}$
	Precision	Set-based precision over CIS	Continuous $\in [0, 1]$
	PipeScore	EM $\times$ Precision (strict success)	Continuous $\in \{0,\ 0.5,\ 1\}$

Tabela 3. Retrieval stage evaluation.

Source	Strategy	Variant	Hit@K	nDCG@K	Recall@10
MITRE	Dense (Top-1)	RAG-M	**100.0**%	N/A	N/A
	Dense (Top-1)	RAG-P / RAG-S	**100.0**%	N/A	N/A
CIS	BM25 (Lexical)	—	80.0%	24.7%	24.0%
	Dense (BGE)	—	80.0%	36.3%	28.0%
	RRF (BM25+Dense)	RAG-C/RAG-P	**100.0**%	33.7%	32.7%
	RRF + Seq. Enrich.	RAG-S	**100.0**%	**46.8**%	**49.3**%

quality (nDCG@10) peaks at 46.8 % and Recall@10 at 49.3 % under the RAG-S variant, justifying its adoption as the retrieval mechanism. It should be emphasized that the Recall@10 metric is strictly limited by the varying dimensions of the target ground-truth listings (Section 3.7).

4.2. Extraction

Table 4 evaluates the extraction phase across the language models and pipeline variants. The Zero-Shot (ZS) baseline exposes clear operational limitations: absent explicit grounding schemas, the SLMs exhibit severe deficits when inferring rigid compliance identifiers. This bottleneck collapses Precision (Gemma: 0.0%; Llama: 7.6%) and constrains Qwen's joint PipeScore to 20.0%, validating that isolated parametric knowledge is insufficient for high-assurance log-auditing. Notably, however, ZS yields unexpectedly high MITRE EM scores for Qwen (84.0%) and Gemma (72.0%), indicating that pre-trained weights suffice to resolve offensive threat categories from native alert signatures. Yet, this capability fails entirely in the defensive domain, proving compliance mapping cannot rely solely on latent weights.

Isolated context ablation (RAG-M and RAG-C) reveals highly asymmetric performance degradation, resulting in a collapse of the joint PipeScore across most architectures. Providing exclusively offensive context (**RAG-M**) maximizes threat identification, yielding a maximum EM of 100.0% for Gemma and Qwen; however, the lack of defensive grounding induces spurious generation of ungrounded compliance tokens, driving the Precision of Gemma and Llama to 0.0%. Conversely, suppressing offensive context (**RAG-C**) eliminates the capability of Gemma and Llama to discriminate the core threat vector (0.0% EM). Under this configuration, Qwen exhibits singular structural resilience, maintaining an 88.0% EM without explicit offensive metadata. This indicates that its parametric space preserves strong latent cross-semantic alignments between defensive controls and threat tactics, sustaining an anomalous PipeScore of 50.0%.

Tabela 4. Extraction stage evaluation.

Model	Metric	ZS	RAG-M	RAG-C	RAG-P	RAG-S
Gemma (5B)	Exact Match (MITRE)	72.0	**100**	0.0	**100**	**100**
	Precision (CIS)	0.0	0.0	51.0	61.0	60.0
	PipeScore (EM×Prec)	0.0	0.0	0.0	61.0	60.0
Qwen (4B)	Exact Match (MITRE)	84.0	**100**	88.0	**100**	**100**
	Precision (CIS)	20.0	20.0	56.0	70.0	70.0
	PipeScore (EM×Prec)	20.0	20.0	50.0	70.0	70.0
Llama (3B)	Exact Match (MITRE)	4.3	90.0	0.0	80.0	80.0
	Precision (CIS)	7.6	0.0	41.0	40.0	50.0
	PipeScore (EM×Prec)	4.3	0.0	0.0	30.0	40.0

Consistent performance peaks were achieved exclusively through the dual-grounding variants, **RAG-P** (Parallel) and **RAG-S** (Sequential). Structuring the alert and offensive/defensive contexts into distinct XML tags mitigated semantic dispersion and eliminated identifier fabrication in our evaluation in all compact models. Qwen established itself as the most robust model for the proposed pipeline, reaching a stable *PipeScore* of **70.0%** under both dual-grounding variants. Gemma also exhibited strong adaptive behavior, recovering from collapse in ZS to 61.0% (RAG-P) and 60.0% (RAG-S). Llama displayed the greatest intrinsic limitations in comprehension and structured JSON mapping, recording the lowest results across all variants.

Cumulatively, these data reveal a performance ceiling inherent to local SLMs. Although the RAG-S workflow yields significant statistical enhancements at the retrieval layer (shifting nDCG@10_CIS from 33.7% to 46.8% and Recall@10 from 32.7% to 49.3%, as shown in Table 3), these upstream gains do not generate proportional extraction improvements for the higher-tier models (where Qwen plateaus at 70.0% and Gemma varies marginally). Only Llama, the weakest model, demonstrably benefits from the richer context.

4.3. Hallucinations

An analysis of pipeline failures reveals a nuance regarding semantic hallucinations (the generation of fabricated identifiers for MITRE or CIS). While strict regex parsing successfully enforces output formatting, it remains inherently blind to fabricated compliance entities (e.g., an invented safeguarding ID like "9.99"). By cross-referencing extracted tokens against the authoritative knowledge bases, we verify that the ungrounded Zero-Shot (ZS) baseline exhibits a substantial 47.3% macro-average rate of identifier fabrication. Conversely, the dual-grounding paradigms (RAG-P and RAG-S) suppress this fabrication rate entirely to 0.0%. Constrained strictly by the injected context window, the SLMs cease to synthesize non-existent countermeasures; consequently, residual pipeline errors stem exclusively from the selection of suboptimal, yet authentic, safeguards.

4.4. Qualitative Evidence: ZS vs. RAG-S

Figure 2 contrasts Zero Shot (ZS) with RAG-S (Qwen 3.5-4B) outputs for the same *Brute Force* alert, illustrating the impact of dual grounding on report quality. Without grounding (ZS), the SLM explicitly acknowledges its inability to identify the vulnerability

Comparison: Brute Force Alert (Qwen 3.5-4B)

Input (JSON Alert): `{"alert_type": "Brute Force", "src_ip": "10.0.0.1", "dst_ip": "192.168.1.100", "protocol": "TCP", "severity": "High"}`

ZS (No Grounding)	**RAG-S**
Extraction: No valid ID extracted	**Extraction:** `["T1110", "5.2", "6.4"]`
Vuln.: "Could not identify a specific vulnerability [...] due to absence of MITRE context in the system"	**Vuln.:** "MITRE ID T1110 — Brute Force"
Impact: "[...] is uncertain without further details on scan scope and vulnerable services"	**Impact:** "Risk of unauthorized access to legitimate accounts [...] compromising data confidentiality and integrity"
Mitig.: "No specific preventive or corrective measures applied [...] due to absence of CIS context in the system"	**Mitig.:** "Account lockout after failed attempts (CIS-5.2); Multi-factor authentication for remote access (CIS-6.4)"

Figura 2. Qualitative evidence for a *Brute Force* alert.

or recommend countermeasures, producing a vague and non-actionable report. With RAG-S, the grounded extraction of the MITRE ID T1110 and CIS safeguards 5.2 and 6.4 supports a specific, traceable technical narrative. The contrast highlights the central causal chain of this work: knowledge base grounding during the extraction stage is a prerequisite for textual coherence in report generation. In ZS, the absence of grounding not only degrades structured extraction but propagates downstream—the model cannot even estimate the incident's impact. In RAG-S, the retrieved identifiers anchor both the impact assessment and the mitigation recommendations, enabling a fully traceable report.

5. Conclusion and Future Work

This paper presented and validated a proof-of-concept (PoC) for a RAG pipeline executed 100% locally under CPU-only constraints. The proposed framework successfully integrates NIDS-driven traffic classification, hybrid source retrieval via RRF, and structured JSON extraction grounded across dual knowledge bases, specifically MITRE ATT&CK and CIS Controls v8.

Our empirical results demonstrate that: (i) single-domain or isolated grounding (RAG-M and RAG-C) may be more detrimental to pipeline performance than its complete absence (Zero-Shot), inducing severe identifier fabrication and schema failures in the lower-capacity model (Llama); and (ii) sequential query enrichment (RAG-S) optimizes retrieval quality, yielding a $+13.1$ percentage point increase in nDCG@10_CIS compared to the parallel baseline (RAG-P); (iii) dual-grounded estimation (RAG-P and RAG-S) eliminates identifier fabrication in our evaluation, consistently providing authentic identifiers for both the classified threat and its corresponding mitigation measure. In conclusion, these insights prove that severe hardware limitations can be overcome by architectural principles, facilitating automated, dependable extraction while maintaining absolute data sovereignty.

Despite the consistency of the empirical data, direct quantitative comparisons with the state-of-the-art were precluded by a gap in the literature regarding systems operating under identical design constraints (strictly local, CPU-only inference with dual structured

grounding), as mapped in Table 1. Evaluating the proposed framework against cloud-based alternatives would compromise core design prerequisites concerning privacy and regulatory compliance (e.g., LGPD). As a PoC, this study focuses on demonstrating feasibility and isolating the causal impact of dual-domain grounding via a rigorous ablation framework, reserving formal comparative benchmarking with architectural equivalents for future work.

Several limitations bound this study, each charting a direct path for future extensions. First, the limited sample scale and temporal horizon of the evaluation data ($N = 150$ records from the legacy CICIDS2017 dataset) restrict definitive generalizations regarding contemporary cyber threats; to address this, future iterations will focus on expanding both the volume and structural diversity of the network telemetry dataset. Second, the parametric bottleneck of the evaluated SLMs (3B–4B parameters); as an immediate priority, we aim to mitigate this inference constraint by evaluating moderately larger model architectures (e.g., 7B parameters) subjected to strict quantization to preserve the local CPU-only execution mandate. Finally, our current methodology remains strictly quantitative, omitting qualitative human-in-the-loop validation that would permit an immediate appraisal of operational utility. Consequently, we plan to execute a human-centric case study, submitting the synthesized reports to senior Security Operations Center (SOC) analysts for formal qualitative validation, usability scoring, and live workflow interpretability assessments.

Referências

Barros, C., Lima, J., Arruda, A., Sousa, R., and Bandeira, A. (2025). Simplificação da análise forense de logs utilizando grandes modelos de linguagem com a técnica rag. In *Anais do XXV Simpósio Brasileiro de Cibersegurança*, pages 839–854. SBC.

Brasil (2018). Lei nº 13.709, de 14 de agosto de 2018. lei geral de proteção de dados pessoais (lgpd).

Brasil (2025). Portaria sgd/mgi nº 9.511, de 28 de outubro de 2025. institui o ppsi 2.0. Ministério da Gestão e da Inovação em Serviços Públicos.

Brasil (2026). Acórdão 258/2026 — tcu — plenário. acompanhamento da governança e gestão de ti nas ifes. Tribunal de Contas da União.

CIS (2022). Cis controls v8 master mapping to mitre enterprise att&ck v8.2. Technical report, Center for Internet Security.

Cormack, G. V., Clarke, C. L. A., and Büttcher, S. (2009). Reciprocal rank fusion outperforms condorcet and individual rank learning methods. *32nd international ACM SIGIR conference on Research and development in information retrieval*.

Corradini, F., Leonesi, M., and Piangerelli, M. (2025). State of the art and future directions of small language models: A systematic review. *Big Data and Cognitive Computing*, 9:189.

da Silva Júnior, A. C. C., Quincozes, S. E., Cattelan, R. G., and Miani, R. S. (2024). Robustness of machine learning-based intrusion detection systems against adversarial attacks in cyber-physical systems. In *Anais do XXI Encontro Nacional de Inteligência Artificial e Computacional (ENIAC 2024)*, pages 227–238.

Daniel, N., Kaiser, F. K., Giladi, S., Sharabi, S., Moyal, R., Shpolyansky, S., Murillo, A., Elyashar, A., and Puzis, R. (2025). Labeling network intrusion detection system (nids)

rules with mitre att&ck techniques: Machine learning vs. large language models. *Big Data and Cognitive Computing*, 9:23.

dos Santos Almeida, L., Spredemann, F., Morales, A. S., Rodrigues-Filho, R., and Panisson, A. R. (2025). Towards a network intrusion classification system using machine learning. In *Anais do XXII Encontro Nacional de Inteligência Artificial e Computacional (ENIAC 2025)*, pages 1914–1925.

Freitas, C. et al. (2024). Enhancing epidemiological insights with rag. In *Anais do Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*.

Houssel, P. R. B., Singh, P., Layeghy, S., and Portmann, M. (2024). Towards explainable network intrusion detection using large language models. In *2024 IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT)*, pages 67–72. IEEE.

Jones, G., Kasimatis, D., Pitropakis, N., Macfarlane, R., and Buchanan, W. (2025). Analysing the role of llms in cybersecurity incident management: Analysing the role of llms in cybersecurity incident managementg. jones et al. *International Journal of Information Security*, 24.

Jüttner, V., Rennekamp, T., Kliem, D., Linke, M., and Reuter, C. (2024). Chatids: Advancing explainable cybersecurity using generative ai. In *2024 IEEE International Conference on Communications (ICC)*. IEEE.

Khan, F. A., Shah, A. A., Alshammry, N., Saif, S., Khan, W., Malik, M. O., and Ullah, Z. (2024). Balanced multi-class network intrusion detection using machine learning. *IEEE Access*, 12:178222–178236.

Melhem, S., Golec, M., Alwarafy, A., and Khamayseh, Y. (2025). Lens: Lightweight and explainable llm-based apt detection at the edge for 6g security. *IEEE Access*, 13:172402–172415.

Novais, M. et al. (2024). Enhancing legal question answering with small models. In *Anais do Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*.

Paduraru, C. et al. (2025). Automated generation of cybersecurity response playbooks via large language models. *Procedia Computer Science*, 270:2987–2996. KES International 2025.

Perrina, F., Marchiori, F., Conti, M., and Verde, N. V. (2023). Agir: Automating cyber threat intelligence reporting with natural language generation. In *2023 IEEE International Conference on Big Data (BigData)*, pages 3053–3062.

Rocha, R. and Pessoa, M. (2024). Advanced rag for local llms. In *Anais do Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*.

Sharafaldin, I., Lashkari, A. H., and Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In *Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP)*, pages 108–116.

Zitei, M. et al. (2024). Open-world text classification with small language models. In *Anais do Encontro Nacional de Inteligência Artificial e Computacional (ENIAC)*.