TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Sanan Mammadli 242907IVCM

REDUCING ALERT FATIGUE IN SECURITY OPERATIONS CENTRES: LLM-BASED INTELLIGENT ALERT TRIAGE

Master's thesis

Supervisor: Ahmed Nagi Abdelaziz Mohamed Nasr

MSc

Tallinn 2026

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Sanan Mammadli 242907IVCM

HÄIREVÄSIMUSE VÄHENDAMINE TURBEKESKUSTES: SUUREL KEELEMUDELIL PÕHINEV INTELLIGENTNE HÄIRETE TRIAAŽ

Magistritöö

Juhendaja: Ahmed Nagi Abdelaziz
Mohamed Nasr

MSc

Tallinn 2026

Author’s declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Sanan Mammadli

12.05.2026

Abstract

Security Operations Centres (SOCs) face a growing challenge in managing the volume of alerts generated by detection systems. A high portion of false positives within this volume contributes to alert fatigue, where analysts become desensitised to alerts and real threats risk being missed. This thesis presents the design, implementation and evaluation of a playbook-guided, multi-step LLM triage assistant that automates the Tier 1 SOC analyst workflow by classifying alerts as High Priority or Low Priority and generating structured investigation notes.

The proposed framework employs a two-round workflow where the first round selects the most relevant playbook for a given alert. The second round performs triage through structured reasoning and various tool calls, including alert and event correlations, IP reputation checks, hash analysis and user lookups. Two GPT-5 models, mini and nano, are evaluated under three operational modes: general, soft playbook-guided and strict playbook-guided. The evaluation uses a synthetic dataset of 100 alerts across 8 categories. A Tool Compliance Rate metric is introduced to measure adherence to expected tool invocation steps. GPT-5-mini under strict guidance achieves the strongest results with 95% accuracy, 94.12% precision, 96% recall and 6% false positive rate. An analyst survey with 14 SOC team members at Neverhack Estonia shows broadly positive perceptions, with 4 out of 5 evaluation dimensions averaging above 4.50 out of 5. Trust is the lowest-scoring dimension. Analyst time estimates range from 12 to 21 minutes per alert compared to the model's sub-minute processing time. This indicates a substantial throughput advantage.

The findings show that LLMs can indeed improve the accuracy and efficiency of alert triage in SOCs by reducing the burden of false positives and analyst workload. The main contribution of this thesis is a low-cost, reproducible playbook-guided LLM triage framework evaluated through both model performance metrics and a SOC analyst survey.

This thesis is written in English and is 70 pages long, including 7 chapters, 10 figures and 29 tables.

List of abbreviations and terms

A2C	Automated, Augmented, Collaborative
AI	Artificial Intelligence
API	Application Programming Interface
AUC	Area Under Curve
CPU	Central Processing Unit
CSA	Cloud Security Alliance
DNS	Domain Name System
EDR	Endpoint Detection and Response
FN	False Negative
FP	False Positive
FPR	False Positive Rate
HCI	Human Computer Interaction
HP	High Priority
IDS	Intrusion Detection System
IIoT	Industrial Internet of Things
IoT	Internet of Things
IP	Internet Protocol
JSON	JavaScript Object Notation
LLM	Large Language Model
LP	Low Priority
LSTM	Long Short-Term Memory
MFA	Multi-Factor Authentication
ML	Machine Learning
MSSP	Managed Security Service Provider
NLP	Natural Language Processing
PII	Personally Identifiable Information
PR	Precision-Recall
RAG	Retrieval-Augmented Generation
RL	Reinforcement Learning
RNN	Recurrent Neural Networks

SD	Standard Deviation
SERC	Security Event Response Copilot
SIEM	Security Information and Event Management
SOC	Security Operations Centre
TCR	Tool Compliance Rate
TN	True Negative
TP	True Positive
TTM	Time to Mitigate
URL	Uniform Resource Locator
XDR	Extended Detection and Response

Table of contents

List of figures

List of tables

1 Introduction

This chapter provides an overview of the study, covering its motivation, research problem, research questions, scope and goals.

1.1 Motivation

Security Operations Centres (SOCs) are responsible for continuous monitoring and response to security threats across organisational networks. As modern infrastructure expands, systems such as Security Information and Event Management (SIEM), Endpoint Detection and Response (EDR), and Extended Detection and Response (XDR) generate an increasing number of alerts. A report from Orca [1] shows that security teams receive hundreds of various alerts every day, with over half of them identified as false positives (FPs). Such a load results in wasted analyst time, delayed investigations and overall inefficiency [2]. This challenge, known as alert fatigue, occurs when analysts become desensitised to large volumes of alerts [2].

Traditional SOCs involve the investigation of each alert manually [3]. This is especially important in the initial stage, as the main outcome is to decide whether the alert should be prioritised. Recent developments in Large Language Models (LLMs) present the potential to address the previously mentioned problems. LLMs are capable of comprehending context, processing large amounts of information and producing structured reasoning. These features make them suitable for supporting complex decision-making tasks such as alert triage in SOCs. As a result, researching the application of LLMs within this domain is important for improving operational effectiveness and mitigating analyst fatigue.

1.2 Research Problem and Questions

The main research problem is the limited availability of intelligent, automated mechanisms for alert triage in SOC environments. While detection systems have become advanced, the triage process relies mostly on manual analyst intervention [4]. This can result in inefficiencies and inconsistent decision outcomes.

Although LLMs have shown potential for automating reasoning and decision support in cybersecurity [5], their specific application in alert triage remains underexplored. A triage workflow is not a single classification task but a sequence of decisions. These include selecting the right investigation approach, invoking the appropriate enrichment tools, interpreting the results and producing a documentable conclusion. How effectively low-cost LLMs can follow such structured processes in a zero-shot setting, without fine-tuning or domain-specific training, remains an area that requires further exploration. In particular, playbook-based triage frameworks that guide LLMs through investigation steps, allowing them to follow analyst reasoning, prioritise alerts and generate investigation summaries, have not been sufficiently explored.

Therefore, this thesis aims to address the following research questions:

MRQ. How can the use of LLMs contribute to the automation of Tier 1 SOC alert triage?

RQ1. What is the performance and cost of LLMs in security alert triage?

RQ2. How does LLM-driven triage address the workload and decision-making challenges faced by SOC analysts?

1.3 Scope and Goal

The scope of this thesis is limited to simulating the Tier 1 SOC analyst workflow using LLMs. It focuses on the classification of synthetically generated alerts but does not include advanced incident response actions. It also excludes model fine-tuning due to the scarcity of available datasets and the high computational cost of training LLMs. Similarly, the model utilises a zero-shot prompting approach rather than few-shot prompting for the same cost reasons. Instead, it relies on general pre-trained LLMs guided through structured prompts and playbooks to evaluate their effectiveness in automating SOC

triage. These constraints represent the main limitations of this study, as the use of synthetic alerts, zero-shot prompting and limited evaluation data may affect the generalisability of the findings.

The main goal is to design, implement and evaluate a playbook-guided, multi-step LLM triage assistant that automates key aspects of alert analysis and reduces the burden of FPs. Additionally, the model emphasises classifying alerts as either “high priority” or “low priority”, and generating structured investigation notes as the outcome. Evaluation includes quantitative metrics such as accuracy, precision, recall, F1 score, false positive rate (FPR), and processing time, alongside survey-based metrics that capture analyst perceptions and estimated triage time. It also includes qualitative feedback obtained through the same survey to evaluate workload and decision-making effectiveness.

The following are the key assumptions for this study:

- Pre-trained LLMs possess sufficient general cybersecurity knowledge to perform accurate alert triage without fine-tuning
- The structured prompts and playbooks are adequate for the representation of Tier 1 SOC analyst workflows.

2 Background

This section focuses on the structure and functions of SOCs and provides a general overview of LLMs.

2.1 Security Operations Centre

A SOC functions as the central hub for an organisation's cybersecurity defence. It combines data from various telemetry sources, such as SIEM platforms, Intrusion Detection Systems (IDS), firewall and endpoint sensors [6]. In essence, SOC is a broad concept referring to a system designed to detect and respond to cybersecurity incidents [7].

SOCs are traditionally structured into 3 or 4 tiers, depending on the organisation [6]. These tiers are divided based on distinct responsibilities that progressively increase in complexity:

- Tier 1 (Alert Analyst) acts as the first line of defence. Their primary duty is to continuously monitor security tools for suspicious activity. Additionally, they verify alerts, document findings and escalate confirmed incidents to higher tiers. [8]

- Tier 2 (Incident Response Analyst) manages escalated cases by performing further analysis and remediation. They coordinate with stakeholders and document response activities. [8]

- Tier 3 (Threat Hunter) takes a proactive approach by identifying hidden or advanced threats through threat-intelligence analysis and in-depth log review. Moreover, threat hunters design mitigation strategies, guide lower tiers and enhance SOC detection. [8]

The hierarchical structure of the SOC analyst roles discussed above is illustrated in Figure 1.

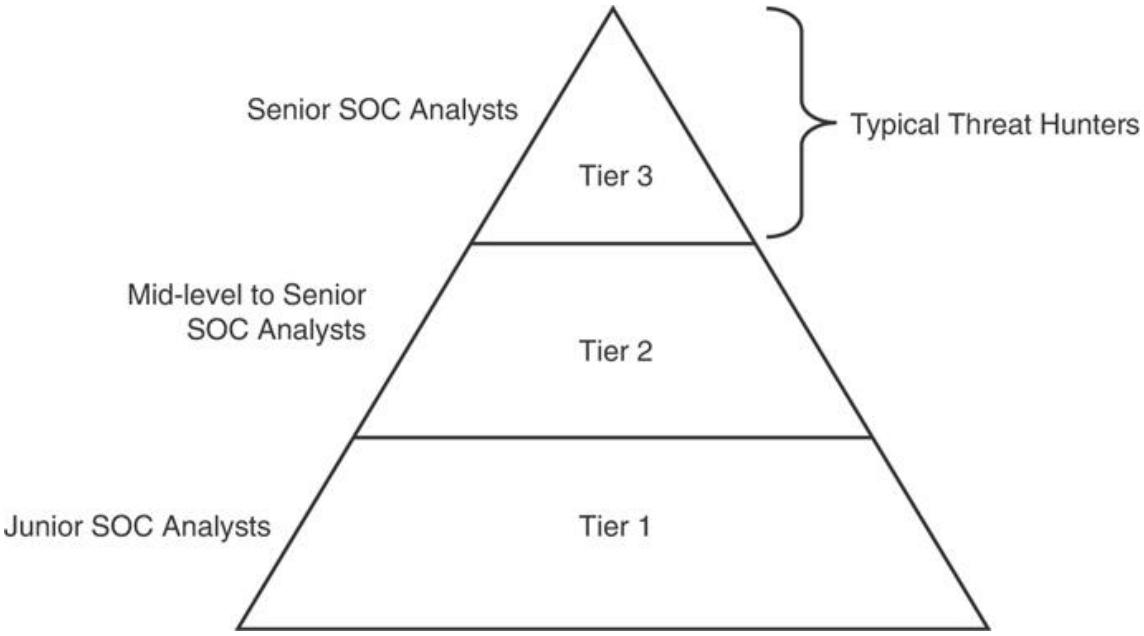

Figure 1. Tiered SOC Analyst Structure [9]

In practice, a triage workflow for a single alert begins when the SIEM generates and queues it based on correlated security events, with Tier 1 analysts serving as the initial escalation point. The analyst then applies analytical and correlation methods to assess and rank the alert by likely significance. Drawing on technical knowledge and operational experience, the Tier 1 analyst assesses whether the activity poses a real threat. Alerts that are identified as genuine threats are passed to higher tiers for further action, while the rest are dismissed. [10,11]

Expanding on the Tier 1 analyst workflow, Kersten et al. [12] classify triage into 4 main categories, as shown in Table 1.

Table 1. Categories of Triage Process [12]

Category	Definition
Relevance indicators	Information used to determine alert relevance to SOC based on its signature and customer scope.
Additional alerts	Related alerts that occurred before and around the time of the parent alert.
Contextual information	Data illustrating the behaviour and other observables of the affected internal host.
Attack evidence	Proof related to the possible attack.

The process shown in Table 1 requires manual review of all alerts assigned to the SOC analyst. Aside from this, analysts are often under pressure to close tickets, since some SOCs evaluate analyst performance based on ticket closure metrics [13]. The nature of this process results in analyst fatigue and burnout [13].

A recent SANS survey [14] indicates that 64% of cybersecurity professionals consider FPs as a major issue. Additionally, this survey mentions that the generation of multiple FPs by detection tools can result in alert fatigue. Similarly, a survey conducted by Tines [15] shows that 63% of SOC respondents experience some degree of burnout, while more than 80% report experiencing increased workload in the past year. Such surveys show that actual threats can be missed, which presents severe risks to organisations. To avoid such threats, Mareedu [16] proposes implementing an Autonomous SOC that also focuses on triage automation.

2.2 Large Language Models

As a subset of artificial intelligence (AI), LLMs are statistical language models trained on extensive datasets utilising machine learning (ML) algorithms to generate and predict human-like text as well as perform other natural language processing (NLP) tasks [17,18]. These capabilities have been driven by advancements in model architectures, marked by the transition from sequential processing to highly parallel and context-sensitive designs [19].

Before the introduction of the Transformer model by Vaswani et al. [20], which laid the foundation for modern LLMs, NLP was primarily dominated by Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks [19]. While effective for short sequences, these architectures struggled with “long-range dependencies”, which is the ability to link information at the start of a long text with information at the end [19]. This was due to the sequential nature of such models.

The new generation of AI was introduced with the launch of the Transformer model proposed by Vaswani et al. [20]. With the replacement of the recurrence method with the self-attention mechanism, it became possible for the model to process all the data at once [20]. This method allows the model to assign attention weights to each word in a sentence relative to all the other words, regardless of their position. For example, in the sentence

“The vase fell off the table because it was unstable”, the self-attention mechanism allows the model to recognise that the pronoun “it” refers to “vase” and not “table”. OpenAI employed the Transformer architecture to build the GPT [21] series of models. GPT-1 validated the potential of semi-supervised learning by pre-training a model on a vast unlabelled dataset and fine-tuning it on a specific task [22]. This was followed by GPT-2 and GPT-3, which validated the potential of scaling model parameters to 175 billion [23]. Currently, multiple other vendors are using the same architecture to develop LLMs with capabilities similar to OpenAI’s GPT [21] series. These include:

- **Claude** [24], developed by Anthropic
- **LLaMA** [25], developed by Meta Platforms, Inc.
- **Mistral** [26], developed by Mistral AI
- **Gemini** (formerly Bard) [27], developed by Google

Current LLMs are useful because they can perform various tasks using different prompts without needing to be retrained for each task. The most basic method is zero-shot prompting, in which the model performs a task using only what it already knows, without being given any examples. To get better results on specialised tasks, few-shot prompting is used. In this technique, the model is only given a few examples in the prompt to guide it in recognising the pattern. [23]

Despite these advancements, LLMs face significant structural and ethical limitations that undermine their reliability. One of the well-known challenges is hallucination, which is when the model generates outputs that are incorrect, nonsensical or unsupported by its training data [28]. Another limitation is data privacy, as models can unintentionally memorise and reveal sensitive information from their training data or from user inputs during interactions [29]. While LLMs have various capabilities, these limitations underscore the need to use them with caution in sensitive or critical situations.

3 Literature Review

This section reviews the current literature on reducing the burden of FP alerts and the application of LLMs and related approaches within SOCs and broader cybersecurity. It focuses on analysing the identified studies in terms of addressed research problems, methods, key findings and limitations. Given the novelty of this research domain, there is currently a limited body of academic work that specifically focuses on the application of LLMs for triage in SOCs. Therefore, some studies in this chapter address threat detection or response tasks instead of direct triage. They are included because effective triage often requires further investigation, including manual threat detection, event correlation and early-response insights. Lastly, this section examines the existing research gaps in the current literature and emphasises the novelty of this thesis.

3.1 Review Protocol

For more focused and objective results, this paper adopted the following literature review protocol:

- **Search Strategy:**
  Databases such as ACM Digital Library, Scopus and Google Scholar were searched using search terms containing the following keywords:
  - "LLM" or "Large Language Model"
  - "SOC" or "Security Operations Center"
  - "Alert Triage"
  - "Incident Triage"
  - "Threat Detection"

  Based on these keywords, the following search terms were formulated:
  - ST1: ("LLM" OR "Large Language Model") AND "Security Operations Center"
  - ST2: ("LLM" OR "Large Language Model") AND ("Alert Triage" OR "Incident Triage")
  - ST3: ("LLM" OR "Large Language Model") AND "Cybersecurity" AND "Threat Detection"

It is important to note that these search terms were adjusted to match each database's structure. Initially, they were searched in all the metadata of the studies. Subsequently, they were limited to only article titles, abstracts and keywords. Later, the forward snowball sampling technique was also utilised by browsing through the list of references for the papers found in the first round. For any other existing tools or products that focus on automated triage or adjacent security automation, a Google search was conducted to identify tools that may not appear in published studies. These included search terms such as "Automated Triage Using LLMs in SOC" and similar queries.

- **Inclusion and Exclusion Criteria:**

  Articles were included if they had discussed alert fatigue, or the use of LLMs and similar models for alert triage, threat detection or response. On the other hand, grey literature was excluded unless it directly described the functionality of an existing tool or product**.** Additionally, papers published before 2017 were also excluded.

- **Selection Process:**

Table 2. Summary of Selection Process

Source	Initial	Inclusion	Manual Inspection
ACM	193	4	1
Scopus	101	8	3
Google Scholar	2740	25	14
Others	-	-	2

Others include grey literature like vendor reports and briefings.

Initially, a total of 3034 articles were found from the above-mentioned databases. This high number was primarily due to broad search results retrieved by Google Scholar. Later, this was narrowed down to 37 articles through title and abstract screening. After a more detailed review and the application of the forward snowballing technique, 20 relevant papers were selected.

3.2 Analysis of Selected Studies

LLM Applications and Analyst Interactions in SOCs

Singh et al. [30] present a field study of GPT-4 use in a live SOC environment, spanning ten months and 3090 analyst queries. This survey included 45 analysts. The results of this survey showed that analysts primarily use LLMs as on-demand cognitive aids for sensemaking. Examples include command interpretation (31%) and refinement of communication for written materials (21%). This study concludes that LLMs assist analyst expertise rather than fully replacing it. One of the limitations of this study is the lack of performance metrics such as triage speed, LLM accuracy and FPR. Findings from Singh et al. [30] emphasise the design goals of this thesis by showing commonly used LLM functions that can be leveraged for automated triage.

Frameworks for Alert Fatigue and Collaboration in SOCs

The study by Chhetri et al. [31] tackles the problem of alert fatigue in SOCs where analysts are overwhelmed by the excessive volume of security alerts. This research breaks down alert fatigue into multiple factors, including high volumes of security events, FPs, understaffing, poor alert prioritisation, and a lack of automation. In order to address these challenges, the authors introduced a conceptual model for human-AI teaming called the A2C Framework (Automated, Augmented, Collaborative), which enables dynamic decision-making across three modes of operation. While the A2C framework looks promising for the mitigation of alert fatigue in SOCs, it utilises a conceptual and qualitative research approach with no specific empirical validation.

Molleti et al. [32] explore the integration of LLM agents into cybersecurity infrastructure to develop automated threat detection and response. Thus, this study addresses the problem of the increasing complexity and velocity of cyber threats, which have rendered traditional defence mechanisms insufficient. Adopting a conceptual approach similar to that of Chhetri et al. [31], this study [32] examines various case studies and frameworks to illustrate potential improvements in threat identification, alert prioritisation, and response planning. Moreover, the study discusses how manual triage is time-consuming and it proposes to utilise multi-agent systems for distributed threat detection and response. It is worth noting that these systems can also be useful in the triage process. However, the study does highlight significant gaps. These include bias in training data and issues with

regulatory compliance, as the latter is a significant concern in sensitive sectors such as finance and healthcare. Overall, the studies by Chhetri et al. [31] and Molleti et al. [32] provide valuable insights into how LLM-based collaboration frameworks can influence analyst workload and decision-making processes in SOC.

Threat Detection and Classification Approaches

Unlike the conceptual framework mentioned in Chhetri et al. [31] for SOC, the study by Trad and Chehab [33] is narrow in scope, as it focuses on phishing Uniform Resource Locator (URL) detection. This is done with the help of LLMs by comparing two approaches: prompt engineering and fine-tuning. This work adopts a quantitative experimental approach to compare model performance (accuracy, precision, recall, F1 score). The results indicate that the fine-tuning approach performs better; however, it requires more resources and maintenance. In addition, prompt-engineered models lack consistency in realistic scenarios.

Ferrag et al. [34] present a new way to detect cyber threats in Internet of Things (IoT)/Industrial Internet of Things (IIoT) networks by implementing LLMs and adapting them to process network traffic data. Similar to the use of LLM-family models for classification in the study conducted by Trad and Chehab [33], this work utilises a transformer called SecurityBERT, but applied to IoT/IIoT traffic. SecurityBERT is evaluated using the Edge-IIoTset cybersecurity dataset. The evaluation results showed an overall accuracy of 98.2% with an inference time of less than 0.15 seconds on average central processing unit (CPU), proving its real-world applicability. On the other hand, the paper mentions various limitations and gaps. For example, some attack types like ransomware presented a lower recall value (0.40), which indicates the need for improvement in the model's classification. Both Trad and Chehab [33] and Ferrag et al. [34] successfully demonstrate LLM adaptability for detection tasks, which can be utilised for event correlation and further investigation during the triage.

Unlike SecurityBERT [34], which focuses on single-event classification, MAD-LLM, developed by Du et al. [35], detects multi-stage attacks using prompt engineering for the aggregation and correlation of security alerts. By doing so, the model intends to reconstruct a multi-stage attack chain. The results indicate an F1 score of 87.49%, a detection rate of 100% and an alert compression rate of 99.8%. This shows significant

noise reduction in alert volume and a strong correlation between alerts. One of the shortcomings of this work is the absence of operational evaluation, which results in a lack of evidence for practical deployment.

Endpoint and Triage-Focused Models

In line with earlier research [31,32], the study performed by Sharif et al. [36] addresses challenges in endpoint protection, one of which is alert fatigue. They propose a self-supervised language model called "DrSec", which is pre-trained using large-scale endpoint telemetry (∼91M processes and ∼2.55B events) to create process representations. These representations are later fine-tuned for multiple SOC tasks, such as triage, process identification, and EDR rule learning. For its alert triage application, this paper directly compares DrSec's ability to distinguish true alarms from false alarms against several baseline systems. This was performed using two datasets: DSOC (from an EDR vendor) and DAtlasV2 (from two Windows 7 endpoints under various attack conditions). The results show that this model achieves 10.8% - 16.94% higher Precision-Recall (PR) Area Under Curve (AUC) than the best baselines, showing better classification and less alert fatigue.

In continuation of efforts to enhance automated triage, Wang et al. [37] focus on automating triage to extract incident content and recommend responsible teams using a model called "COMET". Their methodology involved data collected over a one-year period, offline comparisons with prior triage systems, and an online evaluation against the production baseline. As a data source, the study used proprietary incident records, primarily from Microsoft's cloud services. Similar to the successful results demonstrated by DrSec [36], COMET also showed significant improvements over the existing traditional model, improving triage accuracy by 30% and reducing the time to mitigate (TTM) by 35%. Together, Sharif et al. [36] and Wang et al. [37] provide valuable insight into triage automation; however, both studies fall short of producing comprehensive investigation notes.

Unlike DrSec [36] and COMET [37], three agent-based triage tools extend triage beyond classification accuracy and team routing towards a full investigation workflow. These are Dropzone AI [38], CORTEX [39] and Radiant Security [40]. All three tools target Tier 1 style alert handling with the help of LLM agents; however, Dropzone AI [38] requires a

considerable investment, starting at $36,000 annually. Radiant Security [41] also offers only paid plans, starting at $49/month, with custom pricing available for enterprises. A study [42] conducted by Cloud Security Alliance (CSA) shows a benchmark of analysts with and without the use of Dropzone. It stated that they were faster (45% and 61%) across 2 escalated scenarios with higher accuracy, though the results are sponsor-focused. While there were no specific studies found related to the efficiency of Radiant Security [40], the vendor's solution brief [43] claims up to 95% reduction in analyst workload. CORTEX [39], on the other hand, utilises multi-agent design with typed tool queries. It improves the actionable F1 score by 12% and reduces the FPR by 10.7%, with a median full-ticket resolution time of 152.4 seconds. Nevertheless, this work does not compare triage time against human analysts. Regardless of these limitations, Dropzone AI [38], CORTEX [39] and Radiant Security [40] are useful to this thesis, as they are closely related to LLM-assisted Tier 1 triage and investigation.

Chavali et al. [44] propose reinforcement learning (RL) methods (TD3-AP, SAC-AP) to prioritise IDS alerts, which report better results in reducing defender's loss compared to previous methods. Wang et al. [45] present AlertPro, a context-aware model that re-ranks alerts using contextual cues and analyst feedback. The results of this paper demonstrate that AlertPro is capable of discovering previously missed attacks in a specific public dataset. Both works [44,45] mainly address alert management rather than full investigations and both have limitations. Models in the study conducted by Chavali et al. [44] are computationally demanding, while AlertPro [45] relies on partial analyst feedback and dataset-bound validation. These studies remain essential to this thesis, as alert prioritisation is one of the key approaches in LLM-based triage. The LLM-driven alert correlation used in this study can further support comprehensive investigation and prioritisation during triage.

Contrary to the previous papers [36,37,39,42,44,45], Danquah [46] introduces a conceptual framework to automate triage in SOCs. This study employs a qualitative approach to identify strengths and weaknesses in SOC implementations and later uses this as a baseline for a new framework. It presents an eight-stage automation model that integrates AI specifically into baseline and escalation processes to improve operational efficiency. Nevertheless, the framework is theoretical and lacks empirical validation.

Regardless of these limitations, this work is beneficial as groundwork for automating the triage process.

Conversational Assistants for SOC

Systems like Artemis [47] and the Intelligent SOC Chatbot [48] offer a chat interface over security data, which helps analysts during alert triage. Artemis [47] allowed analysts to ask natural-language questions to avoid query-syntax overhead, while the SOC Chatbot [48] used ChatOps [49] model to integrate tools like VirusTotal [50] and automate ticketing. SOC Chatbot [48] also reduced a SOC analyst's context-switching across various consoles. Being pre-LLM systems, neither Artemis nor the SOC Chatbot was capable of the deep contextual reasoning that modern LLMs provide. Both studies argued that their models result in a higher efficiency; however, only the study conducted by Filar et al. [47] underwent user testing. The underlying design of these models is comparable to this thesis, with the main difference being their lack of LLM integration.

Analyst Assistance and Copilot System

Kaheh et al. [51] developed Cyber Sentinel, a security chatbot powered by GPT-4 [52], designed to help analysts understand threats and take action. It connects to security tools to retrieve threat data and updates rules based on the analyst's instructions. One of the features of this chatbot is to triage incidents by classifying incident severity and guiding analysts through the initial investigation. Analogous to Cyber Sentinel [51], the system proposed by Jüttner et al. [53] called ChatIDS focuses on a chat-based system that uses GPT-3.5-Turbo [52] to explain IDS alerts. The output of this model is an alert explanation in plain language and a suggestion of potential countermeasures for non-expert users. It is worth noting that both systems [51,53] are early concepts that have not been fully tested and they come with ethical issues and privacy concerns.

Ismail et al. [54] extend beyond Cyber Sentinel [51] with Security Event Response Copilot (SERC), which provides analysts with actionable recommendations and guidance instead of focusing on classification or detection, like [33,34]. The dataset was generated by simulating adversarial techniques via the Atomic Red Team tool. The final results showed that Pixtral [55] delivered stable results with consistently high metrics, while OpenAI [52] showed greater adaptability but inconsistent performance across some attack scenarios. However, this study lacks a direct integration of advanced AI capabilities in

the current Wazuh [56] implementations and shows inconsistent performance results across different models used in the framework. Nevertheless, Cyber Sentinel [51], ChatIDS [53] and SERC [54] provide notable insights into how LLM copilots can enhance SOC triage efficiency through recommendation-based and guided workflows.

On the other hand, Mareedu [16] reviews autonomous SOCs and AI agents that also support alert triage, in addition to incident response. It categorises agent types into modular agents, LLM-powered assistants and RL systems. Additionally, this paper highlights dialogue agents that can answer analyst questions, summarise alerts, and trigger playbooks to reduce Tier 1 and Tier 2 triage time. Reported benefits show faster response and reduced workload. However, this work also states that even LLMs, which are able to process unstructured data by default, can struggle to execute reliably without structured integration pipelines. In contrast to the previous studies [51,53,54], this paper provides a conceptual review, which is one of its constraints. The review concludes that AI agents are still emerging in SOCs, which highlights the need for practical frameworks that utilise LLMs within triage and escalation.

3.3 Research Gap

While prior literature has covered the use case of LLMs for automating incident detection, classification and response, few have replicated the Tier 1 SOC analyst workflow at low cost. This workflow involves alert assessment, documentation and escalation. Additionally, most papers rely on fine-tuned models, which limit reproducibility and require significant computational resources.

To address this gap, this thesis aims to focus on a playbook-guided LLM that interprets alerts, follows investigative steps and searches for additional data when necessary. This system is implemented using the latest pre-trained models to produce investigation notes with classification and is evaluated based on multiple SOC analysts' experience.

3.4 Novelty

The novelty of this work specifically lies in its low-cost and reproducible design and the integration of playbook-driven reasoning to automate key Tier 1 SOC analyst tasks. Contrary to prior studies that rely on direct black-box LLM classification, this approach employs explicit reasoning steps aligned with SOC playbooks to improve analysis accuracy.

Compared to similar studies [36,37], which depend on fine-tuning models or embeddings, this paper adopts a zero-shot prompting strategy using publicly available pre-trained models. This makes the model replicable and cost-efficient, especially for smaller SOCs. Another aspect of novelty is the multi-step orchestration. The model introduces separate role-specific LLM modules for playbook selection and alert analysis. Furthermore, this system outputs more than alert labels, including investigation notes that contain supporting alert evidence. Finally, the thesis introduces a dual evaluation framework that combines model performance metrics with assessments from a SOC analyst survey. Such a framework makes the study distinct by bridging gaps between academic LLM experimentation and operational SOC application. This offers empirical insights and a reproducible implementation baseline for future research.

Overall, the main contribution of this thesis is the development of a framework that focuses on the Tier 1 analyst workflow using a playbook-guided multi-step LLM model.

4 Methodology

This chapter presents the research methodology used in this study. It covers the data collection process, overall research design and approach, the fundamental prompts and workflow of the proposed LLM system, and the validation procedures used to assess its performance. Full source code, playbooks, alerts, supplementary scripts, evaluation logs and final investigation notes are available in the repository referenced in Appendix 2.

4.1 Research Design and Approach

This study adopts a mixed methods research design by combining quantitative and qualitative methods to evaluate the proposed LLM-based alert triage framework from both performance and operational perspectives [57]. Creswell [57] defines quantitative research as a theory-driven approach in which the researcher identifies variables, measures them through structured instruments and analyses the resulting numerical data with statistical methods. He further describes qualitative research as an exploratory approach aimed at understanding how individuals perceive and make sense of a given problem, relying on data gathered in natural settings and analysed through an inductive process that moves from specific observations toward broader themes. Mixed methods research combines both forms of data collection and analysis within a single study; its central rationale is that bringing together quantitative and qualitative data yields a fuller understanding of a research problem than relying on either form of data in isolation [57].

In the context of this thesis, the quantitative component consists of two parts. The first is an evaluation of the LLM-based triage framework across 6 different setups on a synthetic dataset of 100 alerts spanning 8 categories, using standard classification metrics described in Section 4.3.4. The second part is the structured Likert-scale ratings and multiple-choice responses collected from SOC analysts through a survey, which provide measurable indicators of analyst attitudes and perceptions. The Likert scale, described by Lazar et al. [58] as an example of ordinal data commonly used for collecting subjective evaluations in Human Computer Interaction (HCI) studies, is an established technique for measuring attitudes in which respondents indicate their level of agreement with a given statement on

an ordered scale. In this study, each statement was rated on a scale where 1 = Strongly Disagree and 5 = Strongly Agree. Tool compliance, processing time and estimated per-alert LLM Application Programming Interface (API) cost are also recorded to capture operational characteristics of each configuration.

The qualitative component consists of the open-ended responses collected within the same survey. These include the per-alert omission descriptions in which participants explained what information they considered missing from the LLM-generated investigation notes, and the two final open-ended questions in which participants described the most operationally useful aspects of the system and the improvements they considered necessary before production deployment.

Classification metrics alone cannot capture how SOC analysts perceive, interpret or would use the model's outputs in practice, while qualitative feedback alone cannot establish whether the model performs reliably across alert categories. The combination of the two directly supports the research questions, particularly the sub-question concerning operational workload and decision-making. It is worth noting that surveys are described by Lazar et al. [58] as an appropriate HCI research method for measuring attitudes, feedback about user experiences and characteristics of users, which aligns with the goals of the SOC analyst survey conducted in this study.

4.2 Data Collection and Synthesis

The dataset for this thesis consists of both true and false positive alerts. All alerts were synthetically generated based on real SOC alert patterns and structures observed during the author's work experience at a Tallinn-based managed security service provider (MSSP), Neverhack Estonia [59]. MSSP refers to an organisation that delivers outsourced security monitoring and management services to multiple clients. Before data processing, formal communication and approval were obtained from Neverhack Estonia to analyse alert structures in a privacy-preserving manner. Due to the near absence of publicly available SOC datasets for research purposes, particularly because of confidentiality and security constraints, a synthetic dataset generation approach was adopted. To ensure data privacy while maintaining the alert accuracy, the process involved the following approach:

- **Pattern Extraction**: Common alert types were identified based on the author's experience at Neverhack Estonia. Representative instances were then collected for each selected category.
- **Personally Identifiable Information (PII) Removal**: All PII, including employee names, internal hostnames and proprietary file paths, was removed.
- **Synthetic Reconstruction**: Using the extracted patterns, similar alerts were generated.

4.2.1 Alert Priority Definition

Within the context of this thesis, alerts were labelled into two categories:

- **High Priority (HP)**: Alerts that represent confirmed threats (true positives) or contain sufficient indicators to require escalation or further investigation.
- **Low Priority (LP)**: Alerts that are either FPs or activities that may be optionally reported at the analyst's discretion and do not require escalation to higher-level analyst teams.

4.2.2 Alert Types and Dataset Distribution

The dataset utilised for this thesis contains eight distinct alert categories, based on common threat scenarios encountered during the author's time at Neverhack Estonia. Each category corresponds to a recognisable SOC detection pattern and is briefly described below to provide context for the evaluation that follows.

- **Data Exfiltration Anomaly**: triggered when an abnormally large or unusual outbound data transfer is detected from an internal asset, indicating potential theft or unauthorised transfer of sensitive data.
- **Suspicious PowerShell Script**: alerts on PowerShell execution patterns that match known malicious behaviour, such as encoded commands, downloads from external sources or the use of suspicious cmdlets such as Invoke-Expression.
- **Domain Name System (DNS) Tunnelling Anomaly**: detects abnormal DNS query patterns that may indicate the use of DNS as a covert channel for command-and-control communication or data exfiltration, typically characterised by unusually long subdomains, high query volumes or queries to uncommon domains.

- **EDR**: alerts generated by Endpoint Detection and Response platforms when malicious or suspicious activity is observed on a managed endpoint. In this thesis, EDR alerts cover **three subtypes**: anti-tampering alerts triggered when attempts to disable or interfere with security services are detected, email identity alerts triggered when indicators of account takeover or email compromise are identified, and malware alerts triggered based on file reputation scoring, threat state classification or suspicious process lineage.
- **Suspicious Azure Network Activity**: detects anomalous or unauthorised activity in Azure environments, including unusual API calls or modifications to network security configurations.
- **Suspicious Login**: alerts when authentication events deviate from normal patterns, such as logins from unusual locations or impossible travel scenarios.
- **Threat Intelligence (Company Logo Detection):** triggered when threat intelligence sources detect the use of the company's logo on suspicious or fraudulent websites, indicating potential phishing or brand impersonation campaigns targeting the organisation.
- **Threat Intelligence (Company Mention in Leaked Documents):** alerts when threat intelligence sources identify mentions of the company in publicly leaked or stolen documents, such as data breach dumps or paste sites, indicating potential data exposure or breach.

6 out of 8 categories consist of 10 samples (5 HP and 5 LP). However, due to the potential for additional scenarios within the Suspicious PowerShell Script and Data Exfiltration Anomaly categories, these were expanded to 20 samples each (10 HP and 10 LP). For further details, see Table 3.

Table 3. Dataset Composition by Alert Type and Label

Alert Category	Total	HP	LP
Data Exfiltration Anomaly	20	10	10
Suspicious PowerShell Script	20	10	10
DNS Tunnelling Anomaly	10	5	5
EDR	10	5	5
Suspicious Azure Network Activity	10	5	5
Suspicious Login	10	5	5
Threat Intelligence (Company Logo Detection)	10	5	5
Threat Intelligence (Company Mention in Leaked Docs)	10	5	5
Total Samples	100	50	50

4.2.3 Synthetic Dataset Format

Each alert was stored in a structured JavaScript Object Notation (JSON) format containing several attributes such as timestamp, alert name, source IP, destination IP, endpoint hostname and process details. Such a format allows consistent parsing and contextual analysis by the model's components. The final dataset provides a balanced set of LP and HP samples required for the evaluation of the model's performance. An example of the mentioned structured JSON format can be seen in Figure 2, where a sample alert from the EDR category is portrayed.


```
{
  "alert_id": "SOC-29890",
  "alert_name": "SentinelOne: XDR Miscellaneous Malware",
  "threat_display_name": "Malware",
  "status": "new",
  "start_time": "2026-01-30T11:01:53.000Z",
  "description": "A threat 'Malware' related to a JavaScript asset within a Chrome Extension was detected. SentinelOne Cloud
  has blacklisted this hash as a known malicious component.",
  "alert_web_url": "https://sentinelone.net/threats/2403686678695455241/overview",
  "evidence": {
    "endpoint": "WKSTN-SALES-080",
    "source_ip": "10.40.15.22",
    "user": "a.jakub@client.ee",
    "user_domain": "CORP-AD",
    "os_name": "Windows 11 Business",
    "file_name": "ea4feaebb8fce87a082623ec5d08d8e7.js",
    "file_path": "C:\\Users\\a.jakub\\AppData\\Local\\Google\\Chrome\\User
    Data\\Default\\Extensions\\inhcgfpbfdjbjogdfjbclgolkmhnooop\\1.6.5_1\\aitopia\\assets\\ea4feaebb8fce87a082623ec5d08d8e7.js",
    "file_sha256": "4a365b80f0de287dcfbf7860075e0956e450048933ddebb0506837afbced404e",
    "process_parent": "chrome.exe",
    "detection_engine": "SentinelOne Cloud",
    "cloud_verdict": "black",
    "confidence_level": "malicious",
    "remediation_status": "mitigated",
    "mitigation_actions": ["kill", "quarantine"]
  }
}
```


Figure 2. Example alert from the EDR category

4.2.4 Contextual Enrichment Structure

In order to simulate a real-world Tier 1 SOC Analyst investigation, not all alerts in the dataset contain the same amount of additional supporting information. While some alerts are provided in isolation, others are accompanied by specific enrichment files to simulate broader visibility:

- **Global Enrichment Files**: These are general reference files available to provide context across the entire environment.
    - **users.json**: A synthetic directory of employees, which includes their emails, full names, departments and roles.
    - **domain_age.json**: A reputation database used to simulate WHOIS [60] lookups.
- **Alert-Specific Telemetry:** These files are created uniquely for specific alert instances when relevant history exists.
    - **recent_events.json:** Previous logs that were observed on the same host or account prior to the alert trigger.
    - **recent_alerts.json**: Previous alerts from the same host or involving similar artefacts that were observed before the actual alert.
- **Project Vitalis (Threat Intelligence)**: A fictional company called Vitalis, created for synthetic threat intelligence alerts.

It should be noted that the depth and time span of these enrichment files were set specifically for this simulation to test LLM performance. Therefore, exact technical details, such as how far back in time data is retrieved (look-back windows), were not considered.

4.2.5 Playbook Structure

10 playbooks were created to guide the LLM through the investigation process. This involved helping it determine what to do and which tools to use. It is worth noting that the number of playbooks does not match the alert categories, as the EDR category encompasses various subscenarios such as malware, identity and anti-tampering. For the full list of playbook names and their associated alert descriptions, see Table 4.

Table 4. List of Playbooks and Associated Alert Descriptions

Playbook Name	Associated Alert Description
EDR Anti-Tampering	Attempts to tamper with security services
EDR SaaS & Email Identity	Account Takeover and Email Compromise
EDR Malware	File Reputation, Threat State and Process Lineage
Excessive Download	Data Exfiltration
Visual Brand Abuse & Logo	Visual Impersonation of the Company
Document Exposure & Threat Intel	Leaked Company Documents
Azure Network Security Group Modification	Network Group Modification in Cloud
Suspicious DNS Query	DNS Tunnelling
Suspicious Login	Location-based login anomalies
Suspicious PowerShell	Script decoding and tool validation

Furthermore, each playbook was written in two linguistic styles to evaluate the LLM's performance in each scenario:

- **Strict**: Aggressive and direct language. Strict playbooks include phrases such as “MAKE SURE”, “INVOKE” and “ONLY” in uppercase letters to emphasise urgency and to indicate that additional tools must be explicitly executed and not forgotten.
- **Soft**: Uses the same instructions as strict playbooks but without capitalising phrases fully or including explicit reinforcing terms like “make sure”.

An example of such linguistic comparison can be seen below:

- Strict Action: "MAKE SURE TO INVOKE 'check_ip_reputation' to retrieve the Abuse Confidence Score, ISP Name, Usage Type, and VPN status for the Source IP."
- Soft Action: "Invoke 'check_ip_reputation' to retrieve the Abuse Confidence Score, ISP Name, Usage Type, and VPN status for the Source IP."

The goal is to see if the usage of strict language reduces the possibility of the model skipping critical investigation steps, especially the execution of required tools.

4.3 Large Language Models

The integration of LLMs into SOC shows a shift from deterministic, rule-based automation to agentic reasoning. This subchapter presents the chosen models, describes the design of the base prompts, outlines the triage workflow and finally defines the evaluation metrics.

4.3.1 Model Choice

As of the time of writing this thesis, the LLM landscape is dominated by several competing providers. OpenAI's most recent model is GPT-5.2, which was introduced on 11 December 2025 as the most advanced model series developed so far for professional applications [61]. This model achieves leading results on a wide range of benchmarks, especially GDPval [62], where it surpasses industry professionals in well-defined knowledge work tasks across 44 domains [61]. Alongside the GPT-series, other leading models include Anthropic's Claude 4 family, Google's Gemini 3 Pro, xAI's Grok 4.1 and open-weight models such as DeepSeek V3.1 and Qwen 3 [63].

However, GPT-5.2 is available only in four new variants through API: gpt-5.2, gpt-5.2-pro, gpt-5.2-chat-latest and gpt-5.2-codex, with no mini or nano versions released so far [64]. The company priced API usage at $1.75 per million input tokens, which is a 40 percent increase over the previous GPT-5.1 and GPT-5 models [64]. Such price points make GPT-5.2 unsuitable for high-volume SOC alert triage, where hundreds to thousands of alerts may need to be processed daily.

To meet the study's budget and performance requirements, the research leverages OpenAI's GPT-5 lineup. This family was initially released with three API-accessible sizes: gpt-5, gpt-5-mini and gpt-5-nano [65]. Each of these models was designed for a different balance of performance, cost and latency. GPT-5-mini is described as a cost-efficient model suited for well-defined tasks, while GPT-5-nano is the smallest and fastest variant, optimised for classification and summarisation [66]. It is also worth noting that all 3 models share a 400000-token context window and accept text and image inputs [65].

For this thesis, GPT-5-mini and GPT-5-nano were chosen. The upper model GPT-5 was excluded due to its higher cost of $1.25 per million input tokens and $10 per million output tokens [64]. Compared to the cheaper tiers, the base GPT-5 model is 5 times more

costly than the mini and 25 times more expensive than the nano model. By choosing the lower-cost tiers, the study remains aligned with real-world deployment constraints. This also allows a direct comparison of the two lower-tier models. It is worth mentioning that among major API-accessible providers at the time of model selection, the OpenAI GPT-5 family offered one of the strongest cost-to-performance ratios at the lower tiers. This configuration ensures a cost-efficient evaluation framework without compromising practical applicability.

In terms of performance, benchmarking shows that the gap between GPT-5-mini and GPT-5 is relatively small across most categories. On GPQA Diamond [67], which is a benchmark that measures PhD-level scientific reasoning, GPT-5-mini scores 82.3% compared to GPT-5's 85.7%. Such a small difference can also be observed in the other benchmarks between the flagship and the mini model. GPT-5-nano shows a more noticeable drop, especially on HLE [68], which is a benchmark that focuses on extremely difficult expert-level questions. It attains 8.7% compared to the flagship model's 24.8%. On the Tau²-bench [69] airline benchmark, which evaluates realistic multi-step function-calling scenarios, the mini model achieves 60% compared to 62.6% of GPT-5. Meanwhile, the nano tier scores 41% in the same benchmark, which is below GPT-4.1's score of 56%. Despite this, GPT-5-nano still outperforms GPT-4.1 on most of the measurements while costing 40 times less per input token. [65]

For the full comparison of the benchmark results, see Table 5.

Table 5. Benchmark Performance and Pricing of GPT Models [64,65]

Model	AIME '25	GPQA Diamond	HLE	MMMU	MMMU-Pro	Tau²-bench airline	Tau²-bench retail	Input $/1M	Output $/1M
GPT-5 (high)	94.6%	85.7%	24.8%	84.2%	78.4%	62.6%	81.1%	$1.25	$10.00
GPT-5-mini (high)	91.1%	82.3%	16.7%	81.6%	74.1%	60.0%	78.3%	$0.25	$2.00
GPT-5-nano (high)	85.2%	71.2%	8.7%	75.6%	62.6%	41.0%	62.3%	$0.05	$0.40
GPT-4.1	46.4%	66.3%	5.4%	74.8%	60.3%	56.0%	74.0%	$2.00	$8.00

It is important to note that the benchmark results shown in Table 5 are related to the high reasoning effort of the GPT-5 family. However, this study utilises the medium reasoning effort setting for both mini and nano models due to lower response latency. This is especially relevant in SOC environments where timely triage is important. For GPT-5-mini at high reasoning effort, the time to first token is approximately 107.5 seconds, while at medium, it drops to 33.23 seconds [70]. Such a difference can also be seen in the nano

model (114.74 to 46.7 seconds) [71]. Since OpenAI has not published official benchmark results for the medium reasoning effort on the mini and nano tiers, the values in Table 5 should be considered as upper-bound reference points. Third-party benchmarks from Artificial Analysis demonstrate that the difference between high and medium is insignificant for both models [70,71]. The mini tier scores 41 at high effort and 39 at medium effort on the Intelligence Index, while the nano tier decreases only from 27 to 26. The Intelligence Index scoring used in Artificial Analysis is based on ten different evaluations covering reasoning, mathematics, agentic tool use and coding, including several benchmarks mentioned in Table 5 [72].

4.3.2 Fundamental Prompts

The triage workflow consists of two essential prompts that guide the LLM through the investigation process. The initial prompt is used to select the playbook and the second prompt contains system-level instructions to initiate the triage process. Both of these prompts are written in the form of static base templates and they remain consistent across all the alerts.

The study employs two prompt structures, namely a playbook-driven approach and a non-playbook approach. Each of these structures slightly changes the way that the prompts are created. Only the alert data and the content of the playbook are injected dynamically during runtime. Moreover, all the prompts utilised in this study are created using the zero-shot prompting method.

Figure 3 presents the prompt used in round 1 to match the incoming alert to the most relevant playbook.

```
Given this alert: {alert_name}
And these available playbooks: {playbook_summaries}
Which filename is the most appropriate to use for this investigation?
Return ONLY the filename (e.g., 'suspicious_powershell.json').
```

Figure 3. Prompt used for automated playbook selection.

Figure 4 demonstrates the system instruction that specifies the role of the model and the expected output form.

```
You are a Tier-1 SOC Analyst. Output JSON only matching this structure:
{template_guide}
Note: {template_note}
```

Figure 4. Base system prompt used across all operational modes.

In addition to the system instruction above, a user message is sent with the playbook and the alert contents. In the general mode, the system instruction is extended with additional triage guidance and the tool usage descriptions, which replace the role of the playbook. For the extended prompt used specifically in general mode, see Figure 5.

GENERAL TRIAGE GUIDANCE:
Weigh all available evidence together. No single factor should determine the outcome. Consider how the indicators interact and whether the overall picture is consistent with legitimate activity or suspicious behavior.

Classify as **High Priority (HP)** if the alert represents a legitimate threat, requires further investigation, or warrants escalation — this includes notifying senior analysts or the client.

Classify as **Low Priority (LP)** if the activity is a false positive or represents authorised behaviour that requires no further action.

TOOL USAGE & PURPOSE: Use the following tools if necessary and if the alert contains the relevant data points:

- **'check_ip_reputation':** Use this to check if a source or destination IP is known for malicious activity.
- **'check_hash_reputation':** Use this to verify if a file hash is associated with known malware.
- **'get_domain_age':** Use this to check if a domain was recently created, which is a common indicator of phishing or C2 infrastructure.
- **'lookup_users':** Use this to verify if an email address belongs to an internal employee and check their organizational role.
- **'get_recent_events':** Use this to check for recent logs which happened before the alert timestamp from same entity.
- **'get_recent_similar_alerts':** Use this to see if the same or similar activity has been flagged before from the same entity or in the broader environment, and how it was previously resolved.

Return **ONLY** the JSON object.

Figure 5. System prompt for general mode triage.

Regardless of the operational mode, the system instructions refer to the same template guide and template note. The template guide defines the expected fields in the output, such as alert metadata, IPs, users and a description summarising the results of the investigation. It also has a classification section in which the model has to categorise the alert as LP or HP. Figure 6 outlines the full template guide structure.

```
{
   "investigation_notes":                                                {
      "Alert ID": "Unique identifier assigned to the alert within the
      SOC system.",
      "Start Time": "Timestamp indicating when the alert was
      generated.",
      "Alert Name": "Name or type of the alert as reported by the
      SIEM.",
      "Source IP": "IP address of the originator or user device
      associated with the alert.",
      "Destination IP": "Target IP address contacted or affected
      during the activity.",
      "Endpoint": "Hostname or device name where the alert
      originated.",
      "User": "Username linked to the activity or login session.",
      "Playbook Used": "Name of the investigation playbook
      automatically selected for analysis.",
      "Description": "Brief summary of what triggered the alert and
      the key observations identified during the investigation."
   },
   "classification": "LP or HP. Only write LP or HP for this part and
   no need for full form."
}
```

Figure 6. Template guide defining the expected output structure.

The template note provides detailed content and formatting guidelines that instruct the model on how to populate the fields defined in the template guide. For the full prompt, see Figure 7.

```
Use the template above as a guide. If a specific field within the template
guide does not exist or cannot be found in the alert evidence, do not
include it in the final output.

You may include up to 4-5 additional relevant keys inside
investigation_notes, but only if they are truly needed, make sense, and
provide actionable value for a client or a SOC analyst solving the ticket
in a system like JIRA. Keep all fields single-level (no nested objects
inside extra fields). All keys inside investigation_notes must be written
in clear, regular English with spaces (e.g., 'IP Reputation Score', not
snake_case or underscored keys).

The Description should summarise what happened and the key observations
clearly in full sentences, written as if a Tier 1 SOC analyst is reporting
it, with a natural human-written tone. It should be between 2 and 4
brief sentences that are not too long, covering all main information.
Remember that the Description field is a MUST for the investigation
notes and that it should not use first person (e.g., 'I classified this
alert as HP').

In addition to summarizing, the Description should provide reasoning for
why the alert is considered Low Priority (LP) or High Priority (HP),
without explicitly mentioning playbooks or SOPs. There is no need to
include remediation actions in the Description.
```

```
Make sure to contextualize the alert properly based on all evidence
provided, including any recent events or similar past alerts.
If certain fields are missing from the alert, you do not need to include
them in the investigation notes or use placeholders like 'none'.
Likewise, there is no need to mention these missing fields in the
description. Some alerts may not have specific fields as they differ.
There is no need to say that 'there is no this or that' in the description
if they are missing from the alert JSON.

Output the final JSON in a readable, beautified format with proper
indentation, not as a single-line minified string.
```

Figure 7. Template note defining content guidelines for investigation notes.

4.3.3 Model Workflow

The proposed model operates through a multi-step approach that replicates the Tier 1 SOC analyst triage process. The workflow, illustrated in Figure 8, begins with the selection of the operational mode (strict, soft, or general).

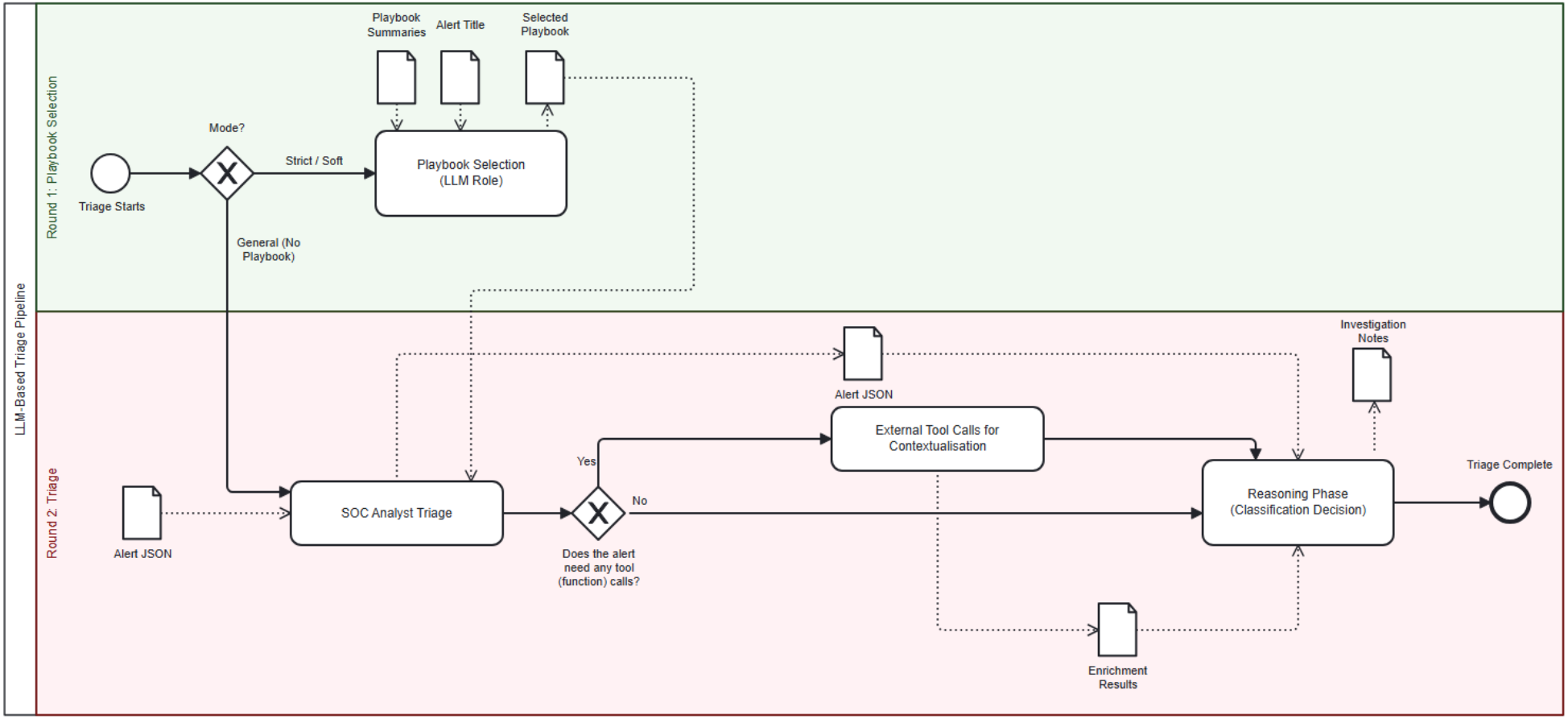

Figure 8. LLM-Based Triage Model Workflow

The workflow illustrated in Figure 8 is as follows:

- **Mode Selection**: The triage process starts based on the determination of the operational mode. In strict or soft mode, the flow proceeds to playbook selection. However, in the general mode, playbook selection is skipped and the alert JSON is passed directly to the SOC analyst triage step in the later round.
- **Playbook Selection (LLM Role)**: Receives the alert title and the summary of all available playbooks from the corresponding directory as input and returns the most relevant playbook for the alert investigation.
- **SOC Analyst (LLM Role)**: Receives the selected playbook in addition to the alert contents and performs the steps outlined in the playbook. Additionally, the model evaluates whether external function calls are needed for additional context.
- **External Tool Calls:** If the model determines that tool calls are required, it invokes the functions to retrieve enrichment data. This is especially relevant in general mode, as the model must independently decide which tools to use based on the alert content and tool descriptions. In the playbook-based modes, the model is expected to follow the tool execution steps outlined in the playbook itself.

  The available tools include:

  - **check_ip_reputation**: Queries AbuseIPDB [73] and VPNAPI.io [74] for abuse confidence, ISP, usage type and VPN/proxy status.
  - **check_hash_reputation**: Queries VirusTotal [50] for detection results, community reputation score and file name.
  - **get_domain_age**: Looks up the domain creation date from the local domain age database.
  - **lookup_users**: Searches for employee role and department from the local employee list database.
  - **get_recent_events**: Returns prior logs from the same entity that occurred before the alert detection.
  - **get_recent_similar_alerts**: Returns previously flagged similar alerts from the same entity or broader environment.

  It should be noted that tool invocations were not enforced at the code level in the playbook-guided modes. This is because even when a playbook explicitly instructs the model to invoke a specific tool, the decision ultimately remains with the model. Certain steps may be skipped if the relevant data fields are absent from

the alert instance or if the model determines the tool is not applicable given the available evidence. For example, invoking check_ip_reputation on a private Internet Protocol (IP) address would yield no meaningful result and the model is expected to recognise this and skip the call accordingly. The results from these tools are later sent to the reasoning process. If no function calls are needed, then the flow advances directly to the reasoning phase.

- **Reasoning Phase:** After gathering all relevant information, the model applies structured reasoning to determine whether the alert is HP or LP.
- **Investigation Notes (Final Output)**: The model returns a structured investigation note that includes key findings and classification results, along with a brief description of what happened in the alert.

The final output is structured as shown in Figure 9.


```
{
  "investigation_notes": {
    "Alert ID": "SOC-12200",
    "Start Time": "2026-01-31T14:45:10.000Z",
    "Alert Name": "Data Exfiltration Anomaly",
    "Source IP": "194.126.101.99",
    "Destination IP": "40.108.201.15",
    "Endpoint": "WKSTN-OFFSITE-04",
    "User": "Liam.O-Sullivan@globalcorp.ee",
    "Playbook Used": "Excessive Download Triage",
    "Workload": "SharePoint",
    "File Count": 850,
    "Baseline 24h": 5,
    "Geo Location": "Tallinn, Estonia",
    "User Role": "Regional Logistics Coordinator (Ireland)",
    "Description": "An unusually large volume of SharePoint downloads (850 files
    vs a 24-hour baseline of 5) was observed from the account while accessing an
    executive/board confidential site at approximately 16:45 local time in
    Tallinn. The account is associated with a regional logistics role that does
    not align with access to executive, financial, and M&A documents, and the
    observed user agent (WinHttp.WinHttpRequest.5) suggests scripted or automated
    retrieval rather than interactive browsing. No recent related events or
    similar alerts were found, but the combination of high volume, sensitive
    content, role mismatch, and programmatic access warrants escalation for
    further investigation."
  },
  "classification": "HP"
}
```


Figure 9. Investigation Notes Example from Data Exfiltration Category.

4.3.4 Model Evaluation and Metrics

This subsection outlines the quantitative method used to evaluate the LLM-based triage workflow. The metrics are designed for the assessment of classification accuracy, compliance, processing time and per-alert LLM API cost. Further quantitative and qualitative evaluation through a survey is addressed separately in Section 4.4.

For the purpose of metric calculation, HP is treated as the positive class, as it is related to alerts that require action or escalation. Table 6 shows how the model's output classification maps to the confusion matrix values against the ground truth label.

Table 6. Triage Classification Logic for Confusion Matrix Derivation.

	Classified as LP	**Classified as HP**
Ground Truth: LP	True Negative (TN)	False Positive (FP)
Ground Truth: HP	False Negative (FN)	True Positive (TP)

For the evaluation of each model and operational mode's performance, a set of standard binary classification metrics was applied across all 100 evaluated alerts. Each alert was assigned a ground truth label prior to the evaluation. Moreover, the model's output classification was compared against this label to compute the following standard metrics widely used in ML and classification tasks [75,76]:

- **Accuracy**: The proportion of correctly classified alerts out of the total evaluated. It provides an overall measure of model performance across both classes.

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

- **Precision**: Measures whether the alerts classified as HP are truly high priority. A low precision value means the model is generating an excessive number of FPs. In terms of SOC, it translates to unnecessary escalations and increased analyst workload.

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$

- **Recall**: The rate at which the model correctly identifies actual HP alerts. This measurement is particularly important in alert triage, as a missed HP alert represents a failure to identify a real threat.

$$Recall = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

- **FPR**: Proportion of actual LP alerts that were incorrectly classified as HP. High FPR is a concern in terms of alert fatigue for SOC analysts.

$$FPR = \frac{\text{FP}}{\text{FP} + \text{TN}}$$

- **F1 Score:** Combined measure of precision and recall. This measurement provides a balanced assessment of model performance by considering FPs and FNs. Additionally, the F1 score is useful when the class distribution is uneven.

$$F1 = 2 \times \frac{Precision \times Recall}{\text{Precision} + \text{Recall}}$$

In addition to the classification metrics mentioned above, a supplementary metric was applied specifically to the playbook-guided approaches. As these modes instruct the model to invoke a specific set of tools, it is important to assess whether the model followed the instructions across all evaluated alerts. The Tool Compliance Rate (TCR) measures the share of alerts in which the model invoked at least the expected set of tools as defined by the selected playbook. A low value for this metric indicates that the model is bypassing investigative steps. Moreover, it undermines the reliability of playbook-guided triage regardless of whether the final classification was correct. It is important to note that while each alert category has a defined set of expected tools based on its playbook, individual alert instances within a category may not require all of them. In these cases, specific exclusions were made to ensure that TCR only captures genuine non-compliance.

$$TCR = \frac{Alerts\ with\ full\ expected\ tool\ invocation}{\text{Total alerts in playbook mode}}$$

Processing time and per-alert LLM API cost were also recorded for each configuration. Processing time measures the total elapsed time from alert submission to the final investigation note, including all tool calls and reasoning steps. Per-alert LLM API cost

was calculated from the observed token counts multiplied by the published OpenAI pricing for the respective model.

4.4 SOC Analyst Evaluation

In addition to the quantitative evaluation, a survey was conducted among SOC staff to assess the operational quality of the model's output and to examine the system's impact on analyst workload and decision-making.

4.4.1 Participants

The survey was conducted among SOC staff employed at Neverhack Estonia. Participants included SOC analysts across all experience levels within the organisation. These include junior, mid-level and senior analysts. A total of 14 SOC team members participated in the study. All participants provided informed consent and all the responses were anonymous.

4.4.2 Alert Selection

Each participant was presented with the same five alerts that were selected from the dataset. This was done to enable meaningful aggregation of ratings per alert. Each of these alerts was from a different category with a balanced set of HP and LP classifications.

The five selected alerts for the survey are as follows:

- Data Exfiltration Anomaly (HP)
- Suspicious PowerShell Script (LP)
- Suspicious Login (HP)
- EDR (LP)
- Threat Intelligence (Company Logo Detection) (HP)

It is important to note that the availability of contextual enrichment across alerts varies, as some include additional enrichment data, while others contain only raw alert data. This variation stems from the category and the instance of the alert.

The investigation notes provided to the participants were the outputs generated by the overall best-performing model and operational mode. The evaluation identified this model to be GPT-5-mini in strict playbook mode. Before the inclusion of investigation

notes in the survey, they were converted into a free-text paragraph style using a Python script for better readability.

4.4.3 Survey Design

The survey was conducted via Google Forms and it followed a mixed approach by combining five-point Likert scale items, multiple-choice questions and open-ended questions. The questionnaire was organised into four sections:

- **Participant Background (A)**: This section gathers professional context for the segmentation of responses based on experience level and role. It also includes the average workload and personal experience of alert fatigue during shifts.
- **Current Triage Practice (B)**: Before the outputs of the model are introduced, participants are asked to estimate the proportion of alerts they typically classify as FPs and specify the stage of the triage process that they consider most time-consuming. These questions establish a baseline of decision-making conditions and workload.
- **Evaluation of LLM-Generated Investigation Notes (C)**: Participants are given five alerts, which they review along with any available enrichment data. For each of the LLM outputs, the SOC staff assesses the investigation notes across five Likert-scale dimensions. This is later followed by an open-ended question on whether any critical information was omitted. Additionally, participants are asked to estimate the time for completing each of the presented alerts within a regular SOC workflow.
- **Perceived Workload and Decision-Making Impact (D)**: SOC personnel are asked to rate the system's potential to reduce workload and support triage decisions. Additionally, they are asked about the level of autonomy for the presented model in a production environment. The section concludes with open-ended feedback on what was most operationally useful and what improvements would be needed before considering such a model ready for production.

For the complete list of survey questions, see Appendix 3.

4.4.4 Data Analysis

Survey responses were analysed using a combination of quantitative and qualitative methods. Multiple-choice and Likert-scale responses were aggregated to identify trends among the SOC staff, with frequency distributions and mean scores per item. Moreover, Likert responses were reported as mean and standard deviation (SD). The per-alert time estimates were aggregated and compared against the model's processing time for each corresponding alert. Finally, open-ended responses from Sections C and D were analysed and interpreted with the quantitative findings to provide a fuller understanding of SOC staff perceptions.

4.5 Validation

As noted in the previous chapters, the model was validated using both quantitative and qualitative methods. Quantitative validation compared the model's classification results against pre-labelled ground truth data. Metrics such as accuracy, precision, recall, F1 score, FPR, and TCR were calculated to assess the model's performance and efficiency. Processing time was measured separately by recording the model's full execution duration and comparing it with the estimates provided by the SOC personnel through the survey. Further quantitative validation was done using questionnaire data to identify various trends among SOC staff through Likert-scale and multiple-choice questions. Lastly, qualitative validation involved analysis of the open-ended analyst survey responses from Sections C and D.

5 Results and Discussion

This chapter presents the findings of the study across two components and discusses their implications. The first covers the quantitative evaluation of the LLM-based triage system across six different configurations. The second part focuses on the findings of a survey conducted with SOC professionals at Neverhack Estonia, assessing the operational quality of the proposed system's outputs and its perceived impact on analyst decision-making and workload. Based on the findings, recommendations for organisations considering the adoption of LLM-based triage are also provided.

5.1 LLM Performance Evaluation

The following subsections report the classification performance, tool compliance, processing time and expected token cost of the proposed model. Results are presented for GPT-5-nano and GPT-5-mini, which are evaluated under general, soft playbook-guided and strict playbook-guided modes. Throughout the evaluation, HP is treated as the positive class. TP refers to an alert correctly identified as HP, and FN means an HP alert incorrectly dismissed as LP. Moreover, TN is an alert correctly classified as LP, and FP is an LP alert incorrectly escalated as HP. In the context of SOC, FP outcomes contribute directly to alert fatigue by generating unnecessary noise, while FN outcomes represent the most critical failure as they correspond to real threats being missed.

5.1.1 GPT-5-nano: General Mode

The general mode of GPT-5-nano serves as the baseline for the nano model. This mode utilises no playbook guidance and relies only on the system prompt and tool descriptions. Table 7 shows the confusion matrix derived from the evaluation of 100 alerts.

Table 7. GPT-5-nano General Mode: Confusion Matrix.

	Classified LP	**Classified HP**
Ground Truth LP	TN = 24	FP = 26
Ground Truth HP	FN = 3	TP = 47

Based on the confusion matrix values presented in Table 7, the classification metrics are as follows:

$$Accuracy = \frac{47 + 24}{100} = 71\%$$

$$Precision = \frac{47}{47 + 26} \approx 64.38\%$$

$$Recall = \frac{47}{47 + 3} = 94\%$$

$$FPR = \frac{26}{26 + 24} = 52\%$$

$$F1 = 2 \times \frac{0.6438 \times 0.94}{0.6438 + 0.94} \approx 76.42\%$$

The overall accuracy of 71% and an FPR of 52% indicate that without structured guidance, the nano model over-escalates a large portion of LP alerts. Despite this, the recall of 94% shows that most real threats are indeed captured, with only 3 HP alerts being missed. Table 8 outlines key results for each alert category.

Table 8. GPT-5-nano General Mode: Per-Category Results.

Alert Category	Accuracy	Recall	FPR
Data Exfiltration Anomaly	60%	100%	80%
DNS Tunnelling Anomaly	50%	100%	100%
EDR	80%	100%	40%
Suspicious Azure Network Activity	50%	60%	60%
Suspicious Login	90%	80%	0%
Suspicious PowerShell Script	85%	100%	30%
Threat Intelligence (Company Logo Detection)	70%	100%	60%
Threat Intelligence (Company Mention in Leaked Documents)	80%	100%	40%

Suspicious Azure Network Activity is the most concerning in this configuration, as it achieved 60% recall compared to 80% and 100% in other alert categories. DNS Tunnelling Anomaly and Data Exfiltration Anomaly both yielded a 100% recall, but at the cost of 100% and 80% FPR respectively. This means that almost every LP alert in these two categories is incorrectly escalated. In contrast to these categories, Suspicious Login and Suspicious PowerShell Script recorded lower FPR values. However, these results should be interpreted with caution, as inconsistent tool invocation in general mode results in classifications based mostly on raw alert content. This is mostly relevant to alert instances where contextual enrichment data was available, but the corresponding tools were not invoked. In such cases, the correct LP outcome was reached despite incomplete investigation rather than as a result of it.

5.1.2 GPT-5-nano: Soft Playbook Mode

In soft playbook mode, GPT-5-nano is instructed with investigative steps written in standard directive language. The confusion matrix results can be seen in Table 9.

Table 9. GPT-5-nano Soft Mode: Confusion Matrix.

	Classified LP	Classified HP
Ground Truth LP	TN = 33	FP = 17
Ground Truth HP	FN = 2	TP = 48

The classification metrics derived from Table 9 are as follows:

$$Accuracy = \frac{48 + 33}{100} = 81\%$$

$$Precision = \frac{48}{48 + 17} \approx 73.85\%$$

$$Recall = \frac{48}{48 + 2} = 96\%$$

$$FPR = \frac{17}{17 + 33} = 34\%$$

$$F1 = 2 \times \frac{0.7385 \times 0.96}{0.7385 + 0.96} \approx 83.48\%$$

Compared to the general mode, overall accuracy rose to 81% and FPR dropped from 52% to 34%. The performance across categories is summarised in Table 10.

Table 10. GPT-5-nano Soft Mode: Per-Category Results.

Alert Category	Accuracy	Recall	FPR
Data Exfiltration Anomaly	70%	100%	60%
DNS Tunnelling Anomaly	70%	100%	60%
EDR	80%	100%	40%
Suspicious Azure Network Activity	80%	100%	40%
Suspicious Login	80%	80%	20%
Suspicious PowerShell Script	90%	90%	10%
Threat Intelligence (Company Logo Detection)	100%	100%	0%
Threat Intelligence (Company Mention in Leaked Documents)	80%	100%	40%

Suspicious Azure Network Activity increased from 60% recall in general mode to 100%. This indicates that structured investigative steps support model reasoning through such alerts. Similarly, Threat Intelligence (Company Logo Detection) achieved perfect accuracy and recall. This suggests that the model can successfully distinguish between HP and LP by focusing on the relevant features of the detected webpage image with the

assistance of playbook guidance. In addition, Suspicious PowerShell Script also saw an improvement, with FPR dropping from 30% to 10%. However, recall in this category dropped slightly from 100% to 90%, though accuracy improved from 85% to 90%. Suspicious Login experienced a drop in accuracy from 90% to 80% with an increase in FPR from 0% to 20%. The observed FPR increase suggests that, in certain cases, the model can correctly gather the required evidence but still fail to draw a correct conclusion from it. In spite of all the improvements in classification, the TCR value of 32% shows that the model follows playbook tool invocation requirements in only one-third of the evaluated alerts. For the detailed comparison to the general mode, see Table 11.

Table 11. GPT-5-nano Soft vs General: Metric Comparison.

Metric	General	Soft	Change
Accuracy	71%	81%	+10
Precision	64.38%	73.85%	+9.47
Recall	94%	96%	+2
F1	76.42%	83.48%	+7.06
FPR	52%	34%	−18
TCR	N/A	32%	—

5.1.3 GPT-5-nano: Strict Playbook Mode

Unlike soft playbook mode, strict playbook mode applies the same instructions in capitalised and reinforced language throughout the investigative process by emphasising mandatory tool invocations. The confusion matrix for this configuration is shown in Table 12.

Table 12. GPT-5-nano Strict Mode: Confusion Matrix.

	Classified LP	Classified HP
Ground Truth LP	TN = 26	FP = 24
Ground Truth HP	FN = 1	TP = 49

Using the values in Table 12, the following metrics are obtained:

$$Accuracy = \frac{49 + 26}{100} = 75\%$$

$$Precision = \frac{49}{49 + 24} \approx 67.12\%$$

$$Recall = \frac{49}{49 + 1} = 98\%$$

$$FPR = \frac{24}{24 + 26} = 48\%$$

$$F1 = 2 \times \frac{0.6712 \times 0.98}{0.6712 + 0.98} \approx 79.67\%$$

Recall reached its highest value across all nano configurations at 98%, with only one HP alert being missed. However, FPR surged to 48% and overall accuracy declined to 75%, which is below the soft mode level. The per-category breakdown is provided in Table 13.

Table 13. GPT-5-nano Strict Mode: Per Category Results.

Alert Category	Accuracy	Recall	FPR
Data Exfiltration Anomaly	50%	100%	100%
DNS Tunnelling Anomaly	50%	100%	100%
EDR	80%	100%	40%
Suspicious Azure Network Activity	80%	100%	40%
Suspicious Login	90%	100%	20%
Suspicious PowerShell Script	85%	90%	20%
Threat Intelligence (Company Logo Detection)	100%	100%	0%
Threat Intelligence (Company Mention in Leaked Documents)	80%	100%	40%

The findings in the table above indicate that Data Exfiltration Anomaly and DNS Tunnelling Anomaly both recorded 50% accuracy and 100% FPR, showing a clear regression from the 70% accuracy achieved in soft mode. Further inspection of the results showed that the drop in performance for Data Exfiltration Anomaly was due to tool

compliance. This specifically affected LP alerts in this category, as they mostly rely on contextual enrichment obtained from tool calls to support a correct LP classification. However, the model failed to invoke the tools in strict mode and classified these alerts as HP, leading to a 100% FPR. It should also be noted that the failure to call the required tools in this category may reflect a one-off inconsistency rather than a systematic pattern. A similar explanation is applicable to DNS Tunnelling Anomaly, where the drop in accuracy points to an isolated occurrence rather than a consistent pattern.

In contrast, Suspicious Login improved from 80% recall in soft mode to 100%, though this should not be attributed to the effectiveness of strict directive language. The improvement can be explained by the fact that in soft mode, the model's overall failure to invoke required tools led to an incorrect LP classification in one HP alert. The same tool omission pattern occurred in strict mode, but in a different HP alert, where the model still reached the correct HP outcome. Hence, the recall difference between the two modes reflects the varying impact of uninvoked tools across individual alerts instead of a consistent improvement driven by stricter guidance. Threat Intelligence (Company Logo Detection), meanwhile, sustained complete accuracy and recall across both playbook approaches. Although classification performance declined in certain alert categories, the TCR of 47% represents an increase of 15% from soft mode, showing that instruction compliance improved. The reduction in classification accuracy can be attributed to the omission of tool calls in the alert categories where contextual enrichment was critical for the classification. This further explains the decline in most of the overall metrics despite the improvement in TCR and highlights an important nuance in interpreting TCR alongside classification metrics. A model can improve its overall tool compliance yet still fail on the specific alerts where enrichment data is most critical, which underlines the importance of evaluating TCR and other classification metrics together rather than treating them in isolation. Table 14 further highlights the changes observed in comparison to soft playbook mode.

Table 14. GPT-5-nano Strict vs Soft: Metric Comparison.

Metric	Soft	Strict	Change
Accuracy	81%	75%	-6
Precision	73.85%	67.12%	-6.73
Recall	96%	98%	+2
F1	83.48%	79.67%	-3.81
FPR	34%	48%	+14
TCR	32%	47%	+15

5.1.4 GPT-5-mini: General Mode

GPT-5-mini in general mode operates under the same prompts as the nano general, which allows a direct model comparison in an equivalent configuration. Table 15 presents the confusion matrix for this configuration.

Table 15. GPT-5-mini General Mode: Confusion Matrix.

	Classified LP	Classified HP
Ground Truth LP	TN = 38	FP = 12
Ground Truth HP	FN = 1	TP = 49

The classification metrics resulting from the table above are as follows:

$$Accuracy = \frac{49 + 38}{100} = 87\%$$

$$Precision = \frac{49}{49 + 12} \approx 80.33\%$$

$$Recall = \frac{49}{49 + 1} = 98\%$$

$$FPR = \frac{12}{12 + 38} = 24\%$$

$$F1 = 2 \times \frac{0.8033 \times 0.98}{0.8033 + 0.98} \approx 88.29\%$$

Compared with the nano baseline mode, GPT-5-mini achieved 87% accuracy and reduced FPR from 52% to 24%. A closer look at the individual alert categories is provided in Table 16.

Table 16. GPT-5-mini General Mode: Per-Category Results.

Alert Category	Accuracy	Recall	FPR
Data Exfiltration Anomaly	85%	100%	30%
DNS Tunnelling Anomaly	70%	100%	60%
EDR	100%	100%	0%
Suspicious Azure Network Activity	70%	100%	60%
Suspicious Login	90%	80%	0%
Suspicious PowerShell Script	100%	100%	0%
Threat Intelligence (Company Logo Detection)	100%	100%	0%
Threat Intelligence (Company Mention in Leaked Documents)	70%	100%	60%

A closer look at Table 16 shows that EDR, Suspicious PowerShell Script and Threat Intelligence (Company Logo Detection) each produced 100% accuracy and 0% FPR. Such an outcome was achieved by the nano model in only one alert category and specifically under playbook guidance. Moreover, Data Exfiltration Anomaly improved from 60% to 85% accuracy and Suspicious Azure Network Activity, despite the 70% accuracy, achieved 100% recall compared to 60% of nano general. Suspicious Login remained the only category with 90% recall, which is consistent across both models in general mode. Despite the overall improvement, DNS Tunnelling Anomaly, Suspicious Azure Network Activity and Threat Intelligence (Company Mention in Leaked Documents) all exhibited a high FPR of 60%, resembling the elevated false escalation pattern observed in the GPT-5-nano baseline. These categories appear to demand structured investigative steps regardless of the model being used. Additionally, a more granular analysis revealed that the increase in overall metrics compared to the nano general configuration was due to more consistent tool invocation. GPT-5-nano in general mode was observed to skip the tool-call process entirely in most cases, whereas GPT-5-

mini engaged more frequently and consistently with a range of tools. For a side-by-side comparison of both models at their baseline, see Table 17.

Table 17. GPT-5-mini vs GPT-5-nano: General Mode Comparison.

Metric	Nano General	Mini General	Change
Accuracy	71%	87%	+16
Precision	64.38%	80.33%	+15.95
Recall	94%	98%	+4
F1	76.42%	88.29%	+11.87
FPR	52%	24%	-28

5.1.5 GPT-5-mini: Soft Playbook Mode

Having established the nano soft configuration results, the same soft playbook approach was applied to the mini model to examine whether this method shows a noticeable improvement in comparison to the baseline. Table 18 displays the confusion matrix for GPT-5-mini in the soft playbook setting.

Table 18. GPT-5-mini Soft Mode: Confusion Matrix.

	Classified LP	Classified HP
Ground Truth LP	TN = 45	FP = 5
Ground Truth HP	FN = 3	TP = 47

Based on Table 18, the following values were calculated:

$$Accuracy = \frac{47 + 45}{100} = 92\%$$

$$Precision = \frac{47}{47 + 5} \approx 90.38\%$$

$$Recall = \frac{47}{47 + 3} = 94\%$$

$$FPR = \frac{5}{5 + 45} = 10\%$$

$$F1 = 2 \times \frac{0.9038 \times 0.94}{0.9038 + 0.94} \approx 92.15\%$$

Compared to the general mode, accuracy improved to 92% and FPR dropped sharply from 24% to 10%. Furthermore, precision improved by approximately 10%, indicating a reduction in incorrect escalations. A further breakdown by alert category is provided in Table 19.

Table 19. GPT-5-mini Soft Mode: Per-Category Results.

Alert Category	Accuracy	Recall	FPR
Data Exfiltration Anomaly	90%	90%	10%
DNS Tunnelling Anomaly	80%	100%	40%
EDR	80%	80%	20%
Suspicious Azure Network Activity	100%	100%	0%
Suspicious Login	90%	80%	0%
Suspicious PowerShell Script	100%	100%	0%
Threat Intelligence (Company Logo Detection)	100%	100%	0%
Threat Intelligence (Company Mention in Leaked Documents)	90%	100%	20%

One of the most notable improvements per category, as shown in Table 19, was seen in Suspicious Azure Network Activity, where accuracy improved from 70% in general mode to 100% with 0% FPR. The structured identity verification and geolocation processes of the playbook seemed to be effective with this rule, as the model was able to dismiss LP cases that were misclassified as HP in the baseline. In total, 3 categories achieved 100% accuracy with 0% FPR, namely Suspicious Azure Network Activity, Suspicious PowerShell Script and Threat Intelligence (Company Logo Detection), which is the same number observed in the general mode. Notably, the 20% increase in accuracy for Threat Intelligence (Company Mention in Leaked Documents) demonstrates the value of having a dedicated playbook to convey the brand context relevant to the alert. Although DNS Tunnelling Anomaly improved from 70% to 80% accuracy, it still retained a high FPR of

40%. This shows that playbook guidance reduced the misclassifications in this category, but not entirely.

Not all categories showed improvement, as EDR experienced a drop in accuracy and recall, as well as an increase in FPR. Further analysis indicated that such a decrease in performance was primarily caused by an unusual case of missing tool calls and one-off occurrences when the LLM did not comply with the playbook. It is also worth mentioning that overall recall fell by 4% compared to the general mode, which reflects a trade-off between reducing FPs and a slight increase in missed real threats. In terms of tool compliance, the TCR of 64% is double the amount recorded for GPT-5-nano under the same mode. This confirms that GPT-5-mini responds considerably more to soft playbook instructions. Changes relative to the mini general mode are summarised comprehensively in Table 20.

Table 20. GPT-5-mini Soft vs General: Metric Comparison.

Metric	General	Soft	Change
Accuracy	87%	92%	+5
Precision	80.33%	90.38%	+10.05
Recall	98%	94%	-4
F1	88.29%	92.15%	+3.86
FPR	24%	10%	-14
TCR	N/A	64%	—

5.1.6 GPT-5-mini: Strict Playbook Mode

GPT-5-mini utilising strict playbook guidance represents the highest level of structured direction evaluated in this study. Table 21 presents the confusion matrix for this configuration.

Table 21. GPT-5-mini Strict Mode: Confusion Matrix.

	Classified LP	**Classified HP**
Ground Truth LP	TN = 47	FP = 3
Ground Truth HP	FN = 2	TP = 48

Table 21 yields the following classification metrics:

$$Accuracy = \frac{48 + 47}{100} = 95\%$$

$$Precision = \frac{48}{48 + 3} \approx 94.12\%$$

$$Recall = \frac{48}{48 + 2} = 96\%$$

$$FPR = \frac{3}{3 + 47} = 6\%$$

$$F1 = 2 \times \frac{0.9412 \times 0.96}{0.9412 + 0.96} \approx 95.05\%$$

GPT-5-mini under the strict playbook guidance achieved the highest overall performance across all 6 configurations with 95% accuracy, 94.12% precision, 96% recall and 6% FPR. Table 22 outlines how this performance was distributed across various alert categories.

Table 22. GPT-5-mini Strict Mode: Per-Category Results.

Alert Category	Accuracy	Recall	FPR
Data Exfiltration Anomaly	95%	100%	10%
DNS Tunnelling Anomaly	90%	100%	20%
EDR	90%	80%	0%
Suspicious Azure Network Activity	100%	100%	0%
Suspicious Login	90%	80%	0%
Suspicious PowerShell Script	100%	100%	0%
Threat Intelligence (Company Logo Detection)	100%	100%	0%
Threat Intelligence (Company Mention in Leaked Documents)	90%	100%	20%

As seen in Table 22, 3 categories sustained perfect accuracy and 0% FPR, which were identical to the ones in soft mode. In contrast to the soft playbook results, Data Exfiltration Anomaly advanced further to 95% accuracy with 100% recall. The improvement stemmed from the recovery of one HP alert that was previously missed in soft mode due to a missing tool invocation. DNS Tunnelling Anomaly also experienced an increase from 80% to 90% accuracy; however, this seems to be a result of LLM's black-box nature rather than systematic improvement. Recall values for EDR and Suspicious Login (both 80%) were identical to the soft playbook approach. In general, these results indicate that improvements compared to the soft playbook mode were not uniform across all alert categories.

Contrary to the transition from GPT-5-nano soft to strict playbook, where accuracy fell sharply, the mini model benefited from the stricter guidance with improvements across every overall metric. On the compliance side, the TCR of 94% represents a 30% gain over the soft mode, highlighting that stricter language indeed results in better instruction-following fidelity. Considering the near-perfect classification performance and peak tool compliance, this configuration is established as the strongest overall result in this research. For a detailed list of overall metric changes relative to soft mode, see Table 23.

Table 23. GPT-5-mini Strict vs Soft: Metric Comparison.

Metric	Soft	Strict	Change
Accuracy	92%	95%	+3
Precision	90.38%	94.12%	+3.74
Recall	94%	96%	+2
F1	92.15%	95.05%	+2.90
FPR	10%	6%	-4
TCR	64%	94%	+30

5.1.7 Cross-Configuration Comparison

The following tables in this subchapter compile the results from all six configurations by presenting classification performance, tool compliance, processing time and cost estimates. In addition, these configurations are compared based on the mentioned four dimensions.

Table 24 gives an overview of classification metrics across all 6 model setups.

Table 24. Overall Classification Metrics Across All Configurations.

Model	Mode	Accuracy	Precision	Recall	F1	FPR
GPT-5-nano	General	71%	64.38%	94%	76.42%	52%
GPT-5-nano	Soft	81%	73.85%	96%	83.48%	34%
GPT-5-nano	Strict	75%	67.12%	98%	79.67%	48%
GPT-5-mini	General	87%	80.33%	98%	88.29%	24%
GPT-5-mini	Soft	92%	90.38%	94%	92.15%	10%
GPT-5-mini	Strict	95%	94.12%	96%	95.05%	6%

Among the configurations above, GPT-5-mini outperforms the nano model on accuracy, precision and F1, and achieves significantly lower FPR. Moreover, the effect of increasing playbook guidance on GPT-5-mini is consistently positive, as the accuracy rose from 87% to 95% and FPR decreased from 24% to 6% across the 3 modes. However, GPT-5-nano followed a less predictable path. While soft mode improved all metrics over

general mode, strict mode exhibited degraded accuracy and FPR compared to the soft mode, despite achieving the highest recall among all the nano configurations. This divergence was mainly caused by the nano model omitting tool calls in alerts where such calls were critical, even though its overall tool-calling performance was better than in the soft playbook approach. Table 25 further presents TCR in all 4 playbook-guided modes.

Table 25. Tool Compliance Rate: Playbook-Guided Configurations.

Model	Mode	Compliant Alerts	Total Alerts	TCR
GPT-5-nano	Soft	32	100	32%
GPT-5-nano	Strict	47	100	47%
GPT-5-mini	Soft	64	100	64%
GPT-5-mini	Strict	94	100	94%

As can be seen from the above table, strict playbook guidance produced substantially higher tool compliance than soft language for both models. In addition, GPT-5-mini followed playbook instructions far more reliably than GPT-5-nano across both modes. The gap between the GPT-5-mini strict (94%) and GPT-5-nano strict (47%) is especially noteworthy, even though both use the same prompt structure. This suggests the models differ fundamentally in how they follow instructions for tool use.

As for processing time, Table 26 shows how long each configuration took to process alerts on average.

Table 26. Average Processing Time per Configuration.

Model	Mode	Average Processing Time (sec)
GPT-5-nano	General	19.18
GPT-5-nano	Soft	27.72
GPT-5-nano	Strict	33.75
GPT-5-mini	General	29.72
GPT-5-mini	Soft	38.92
GPT-5-mini	Strict	41.73

The nano model was faster than the mini in all three approaches. Moreover, processing time increased as the playbook language became stricter for both models. This can be explained by the increasing TCR values seen in Table 25, as the higher volume of tool calls led to longer execution times. The shortest average duration was observed in GPT-5-nano general (19.18 seconds), while GPT-5-mini strict recorded the longest (41.73 seconds). Despite these differences, all six setups operated within the processing times suitable for SOC triage workflows. This is analysed in more detail in Section 5.2, where observed times are compared with analyst estimates from the SOC personnel survey.

It should be mentioned that these numbers were obtained in a synthetic environment with instant results from tool calls. Tools that require live queries, specifically get_recent_events, get_recent_similar_alerts and lookup_users, would rely on connections to internal systems in a real SOC setting. These systems can include SIEMs, employee directories and ticketing platforms. The latency involved in retrieving this data would likely increase the total processing time per alert, though the amount would depend on the environment's infrastructure and volume of historical data being queried.

Concluding the comparison across configurations, Table 27 summarises the average token usage and the corresponding estimated cost per alert across all 6 modes. GPT-5-mini is priced at $0.25 per million input tokens and $2 per million output tokens, while GPT-5-nano is billed at $0.05 per million input tokens and $0.40 per million output tokens.

Table 27. Average Token Usage and Estimated Cost per Alert.

Model	Mode	Avg Input Tokens	Avg Output Tokens	Est. Cost per Alert ($)	Est. Cost per 100 Alerts ($)
GPT-5-nano	General	1970	3353	0.00144	0.144
GPT-5-nano	Soft	4040	4038	0.00182	0.182
GPT-5-nano	Strict	4478	4560	0.00205	0.205
GPT-5-mini	General	2737	1856	0.00440	0.440
GPT-5-mini	Soft	4902	2345	0.00592	0.592
GPT-5-mini	Strict	5516	2521	0.00642	0.642

As shown in Table 27, GPT-5-nano is considerably more cost-efficient than GPT-5-mini across all modes due to its lower token pricing. Estimated costs range from $0.00144 per

alert in general mode to $0.00205 in strict mode, compared with $0.00440 to $0.00642 for the mini model. It is worth mentioning that despite the lower pricing, GPT-5-nano produced significantly higher output token counts in all configurations. This is due to higher reasoning token consumption, which likely indicates that the model requires more deliberation to reach a classification decision, along with final investigation notes.

Input token counts increase as playbook language becomes stricter and more precise, reflecting the added prompt content from playbook instructions and enrichment data retrieved from tool calls. It should be noted that the values in Table 27 represent averages over the evaluated dataset and that the actual token consumption in a production SOC environment would vary depending on alert complexity and available contextual data. Alerts without any enrichment context would consume fewer input tokens and result in lower costs. Conversely, alerts with rich contextual data, such as extensive recent event logs or multiple similar historical alerts, would more than double the input tokens and raise the cost per alert. These figures should therefore be interpreted as indicative estimates instead of fixed operational costs.

5.2 SOC Analyst Survey Insights

This section presents and discusses the findings from the evaluation study administered to SOC staff at Neverhack Estonia. The full quantitative results are provided in Appendix 4 and the qualitative responses in Appendix 5. Percentages in this subchapter and appendices are reported without decimal places to avoid overstating precision given the sample size.

5.2.1 Participant Background and Triage Baseline

A total of 14 SOC members participated in the survey. In terms of experience, 29% reported less than 1 year, 50% between 1 and 3 years, 14% between 3 and 5 years and 1 participant more than 5 years. Regarding role distribution, 43% were junior analysts, 36% were mid-level analysts and 21% held senior analyst positions.

When they were asked about alert fatigue, 57% reported experiencing it sometimes, 29% often, 7% rarely and 7% never. Additionally, 64% of the participants reported that more than 75% of alerts in their workload are FPs. These figures confirm that the problem

motivating this study is directly relevant to the respondent pool. The most cited time-consuming stage of triage was log and event correlation. This was selected by 10 out of 14 participants, followed by external tool lookups and documentation. The full demographic and baseline distributions are provided in Appendix 4.

5.2.2 Evaluation of LLM-Generated Investigation Notes

Each participant was asked to evaluate the results of the same set of five alerts. Scores were collected based on a 5-point Likert scale, where 1 = Strongly Disagree and 5 = Strongly Agree. Table 28 presents the mean and SD across all 5 alerts.

The evaluated metrics include accuracy, classification, clarity, trust, reasoning, and omissions. Accuracy refers to whether the investigation note captures the key findings of the alert, while classification assesses alignment with the HP and LP criteria. In addition, clarity evaluates how clearly and professionally the note is written, and trust reflects the extent to which the output can support a triage decision without further investigation. Finally, reasoning measures whether sufficient justification is provided for the classification, and omissions capture whether any critical information expected from a Tier 1 analyst is missing.

Table 28. Mean Likert scores per alert (mean ± standard deviation).

Alert	Accuracy	Classification	Clarity	Trust	Reasoning	Omissions
Data Exfiltration (HP)	4.64 ± 0.50	4.50 ± 0.65	4.71 ± 0.47	3.57 ± 0.94	4.57 ± 0.51	3/14
PowerShell Script (LP)	4.71 ± 0.47	4.79 ± 0.43	4.79 ± 0.43	4.00 ± 0.78	4.71 ± 0.47	1/14
Suspicious Login (HP)	4.64 ± 0.50	4.79 ± 0.43	4.50 ± 0.76	3.93 ± 1.07	4.71 ± 0.47	5/14
EDR (LP)	4.00 ± 0.68	4.14 ± 0.86	4.14 ± 0.95	2.86 ± 0.86	4.14 ± 0.95	7/14
Threat Intelligence: Leaked Documents (HP)	4.71 ± 0.47	4.79 ± 0.43	4.57 ± 0.65	4.07 ± 1.21	4.64 ± 0.50	4/14
Overall	4.54 ± 0.58	4.60 ± 0.62	4.54 ± 0.70	3.69 ± 1.06	4.56 ± 0.63	20/70

Overall, the results were positive. 4 out of 5 Likert dimensions recorded an overall mean above 4.50 with tight SDs (SD = 0.58-0.70), demonstrating strong and consistent agreement across participants. Furthermore, the classification agreement reached a mean of 4.60 (SD = 0.62). This indicates that the model's HP and LP decisions were generally consistent with study-defined HP and LP criteria. Accuracy and reasoning scored 4.54

(SD = 0.58) and 4.56 (SD = 0.63), respectively, and clarity reached 4.54 (SD = 0.70), showing outputs were considered well-written and suitable for the ticketing system.

Among all these metrics, trust was consistently the lowest scoring, with an overall mean of 3.69 (SD = 1.06). The highest overall SD among the 5 metrics (1.06 compared to 0.58-0.70) also reflects genuine variability in participant attitudes towards trust, in contrast to the strong consensus observed in other metrics. This dimension indicates that while participants agreed with the content of the investigation notes, they were cautious about relying on them for triage decisions without further review. The qualitative feedback identified verifiability as the central concern, with one of the participants requesting direct links back to the source event and fields rather than plain-text summaries. This is a valid consideration for a production environment; however, it was not feasible in the synthetic setting of this study due to the absence of SIEM or similar tool integration.

One of the visible patterns in Table 28 is that the EDR alert recorded the lowest mean scores across all 5 metrics, with accuracy at 4.00, classification, clarity and reasoning all at 4.14 and trust at 2.86. The EDR trust score stands out at 2.86 (SD = 0.86), falling below the neutral midpoint, suggesting that participants were more inclined to distrust the output than to simply hesitate. This category also produced the highest omission rate, with 7 of 14 participants noting that critical information was absent from the investigation notes. Several respondents noted the absence of the file origin context. It is worth noting that file type and folder path were present in the alert data; however, the LLM did not specifically point out the nature of the .lock file within the investigation notes. For the same alert, one of the participants highlighted a further need to investigate the detected file. This is also another feature that can be incorporated into the model; nonetheless, limitations such as the synthetic environment and the absence of live file access in the current implementation did not allow for such investigation. The same participant also noted that the source IP was sent to the IP lookup tool, although it was a private IP. This shows that even though the LLM was instructed with a playbook to avoid such cases, there is still a risk of hallucination.

Suspicious Login resulted in a notable omission rate of 5 out of 14. Participants mostly pointed out that some event correlations were missing. These included prior login history, application correlation and response action guidance. While the mentioned details were due to limitations in the synthetic data creation, the missing response action was not an

inherent gap in the model but rather a deliberate design choice, as the scope of this study was limited to mimicking Tier 1 analyst workflow rather than incident response. In the same alert instance, one of the respondents mentioned that some field naming conventions, such as Multi-Factor Authentication (MFA) Status with a value of false, were ambiguous for client-facing use. This also includes fields such as “IP Reputation Score: 100”, as it is not clear what this key-value pair means, even though it was addressed in the description of the alert investigation notes. Such issues can be directly addressed with the help of a playbook, as the LLM can be instructed to follow a clear naming scheme.

There were also field-based issues raised by one of the SOC members in the Threat Intelligence alert. The first one was the inclusion of the domain IP as the Source IP, which is logically inconsistent. A further concern was the defanging of detected URLs to prevent analysts or clients from accidentally clicking on malicious links. Furthermore, participants indicated that they would make use of additional tools, such as sandbox environments, suggesting that such capabilities could be integrated directly into the model itself. There was also a comment regarding the classification of the Threat Intelligence alert, which stated that such alerts are usually LP unless there is a critical breach. It should be noted that the HP classification assigned to this alert reflects the study-specific definition of High Priority, which covers alerts with sufficient indicators to warrant escalation or further investigation, rather than a universal industry-standard classification. In contrast to other alert instances, the Suspicious PowerShell Script produced the fewest omissions and the highest scores across most dimensions.

General open-ended feedback showed that one of the most useful features of the model was automated aggregation of technical indicators, such as source IP, username, and IP abuse confidence scores. One respondent explicitly noted that the system helps reduce the time spent writing investigation notes and increases focus and time for the analysis itself. Others highlighted automated lookups, automated root cause writing and historical incident references as the most valuable outputs. The correlation between technical indicators and business context, and the availability of additional context from past cases were also mentioned. This indicates that the model supports decision-making by surfacing relevant context that would otherwise need to be gathered manually. Several responses

noted the value of having a formatted, copy-paste-ready summary for ticketing systems, though one mentioned that the descriptions would benefit from sounding less robotic.

Regarding improvements before production use, the most notable concern was the verifiability of evidence. One of the participants requested simplified access to source evidence, including direct links to event fields and user profiles, a mechanism for flagging inaccurate fields for the analyst, clearly marked fields that could not be filled automatically and a demonstrated track record of accuracy. By simplified access to source evidence, the respondent referred to enrichment data populated through unknown methods, arguing that if the source of a value cannot be trusted, it raises the question of whether it should be included at all. This concern is more related to general contextual enrichment, where correlation methods are used to derive and enrich attributes (e.g., associating a username based on multiple underlying logs whose origin may not be fully known). This is indeed relevant before deployment; however, the proposed model does not incorporate enrichment generated through non-transparent methods.

Concerns about the risk to junior analysts were also raised, with a detailed example using the EDR alert, noting that an overconfident root cause conclusion could lead an inexperienced Tier 1 analyst to underestimate the incident's criticality. This could be addressed by introducing explicit confidence qualifiers in the investigation notes, which can be useful in such ambiguous cases. Additionally, the LLM can be instructed to emphasise when there is a need to escalate to a senior analyst if the evidence is insufficient to rule out a threat. Another respondent noted the need for improvements in malware investigation but expressed satisfaction with less technical alert types. The need for more model training on real-world data and a thorough review of outputs before full production deployment was also highlighted. In relation to such a review, AI hallucinations were also mentioned as a potential concern that would need to be addressed. Finally, tool calibration was raised as a further consideration, with unnecessary calls such as IP reputation checks on internal IPs. Such cases can be addressed by implementing a script-based validation that checks whether an IP address falls within private ranges before allowing a tool to be executed. Full individual responses are provided in Appendix 5.

5.2.3 Time Comparison

For each provided alert, participants estimated how long they would personally take to complete end-to-end triage, accounting for accessing relevant environments, reviewing logs, performing tool lookups, and writing the final investigation notes. Table 29 compares these estimates against the model's average processing time per alert category under the GPT-5-mini strict configuration.

Table 29. Analyst Estimated Triage Time vs Model Average Processing Time.

Alert	Mean Analyst Estimate (min)	Model Average Processing Time (sec)
Data Exfiltration Anomaly	13.4	39.22
Suspicious PowerShell Script	12.0	42.89
Suspicious Login	14.3	33.35
EDR	21.2	47.37
Threat Intelligence: Leaked Documents	12.7	31.00

In the case of time estimates provided as ranges in the survey, the lower bound was used to maintain a cautious estimate. Analyst estimates ranged from approximately 12 to 21 minutes across 5 alerts, compared to the model's time of 31 to 47 seconds per category. The EDR alert recorded the highest analyst time estimate, a mean of 21.2 minutes, which aligns with the additional research effort participants described in their omission comments. These specifically included manual investigation of file origin and the Gradle cache context. Furthermore, EDR exhibited the longest model processing time among the 5 alert types, at 47.37 seconds, likely reflecting greater reasoning effort and tool usage. The Threat Intelligence alert showed an inverse pattern, with the shortest model runtime of 31 seconds and a relatively low analyst approximation of 12.7 minutes. This is consistent with the straightforward nature of the investigation steps required for this alert type.

Looking at the throughput, the difference between the analyst and the model is substantial across all five alert categories. Even for the alert type analysts estimated to be the quickest to triage (12 minutes on average), the model completed the triage steps in under a minute. Such a difference shows that, potentially, in production, the system can process a volume

of LP/FP alerts that would otherwise occupy a significant portion of a Tier 1 analyst's shift, leaving more time for higher-priority investigations. It is worth noting that the approximations for each alert given by the SOC staff were made in a reflective rather than operational context. As such, the figures represent perceived effort instead of measured task duration and should be interpreted as indicative of potential throughput benefits rather than precise operational measurements.

5.2.4 Workload Perception and Autonomy Preferences

When asked whether LLM-generated triage notes of the presented quality could meaningfully reduce their workload, respondents reported a mean score of 4.00 (SD = 0.88), with the majority selecting agree or strongly agree. However, comfort in relying on these outputs when making triage decisions was considerably lower (M = 3.29, SD = 1.07). Across 14 participants, workload reduction ratings were equal to or higher than comfort ratings when relying on LLM outputs for triage decisions. No respondent rated comfort higher than perceived workload reduction. This pattern shows that while the model is seen as helpful in reducing workload, there is relatively lower confidence in relying on its results for decision-making. A similar trend was observed in Section 5.2.2, where high agreement with the content did not translate into willingness to act without further review. Analysts suggested several improvements for production use, including direct links to source evidence fields, a demonstrated track record of accuracy over time and exposure to more real-world alert data through further model training. Addressing these can increase trust in the model's outputs. Moreover, these findings suggest that analysts are willing to use the outputs as a starting point but retain a clear preference for verifying conclusions before acting on them.

In terms of autonomy preferences, Figure 10 illustrates the distribution across all 4 options.

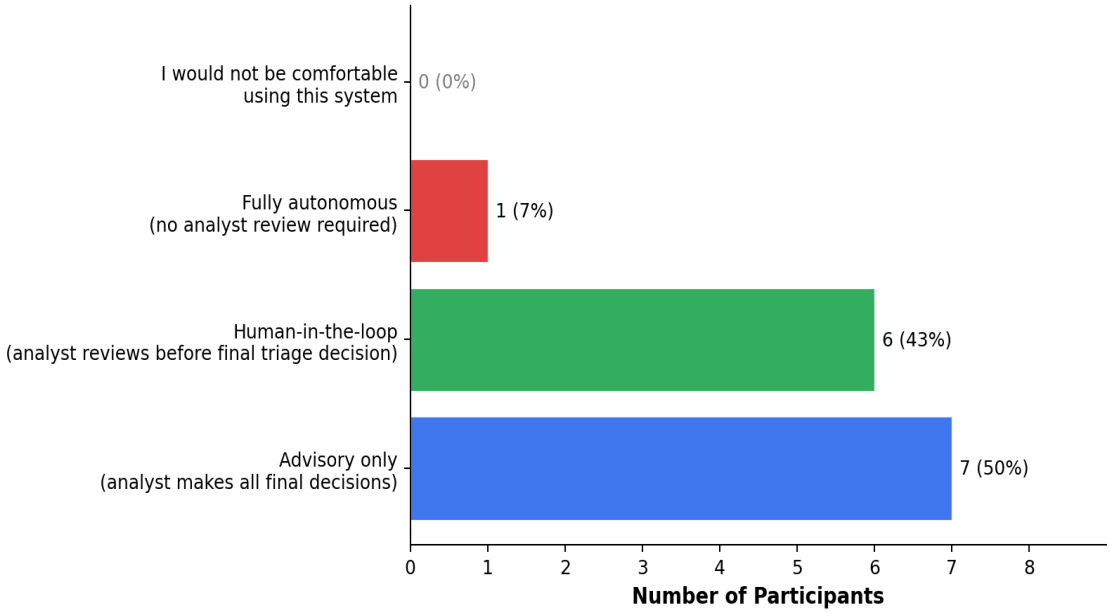

Figure 10. Autonomy Preference Distribution.

As shown in Figure 10, 50% of participants selected the advisory-only option and 43% preferred a human-in-the-loop model in which the analyst reviews before the final triage decision. No participant selected the option indicating they would not be comfortable using the system, which shows that analysts are open to having such a model available as support. The preference for advisory and human-in-the-loop modes, which together account for 93% of the respondents, is consistent with the workload reduction motivation behind this thesis. The model is intended to reduce manual effort at initial triage rather than replacing analyst judgement entirely. The results confirm that this framing is well-received within the evaluated SOC group. The fully autonomous approach was selected by a single participant and falls outside the overall distribution trend. Combined with the trust scores reported in Section 5.2.2 and also at the beginning of the current section, these preferences confirm that a human-in-the-loop or advisory deployment approach is the most appropriate target for this model in its current form. Any further move towards greater autonomy would need to be preceded by meaningful improvements in output verifiability and demonstrated reliability over time.

5.3 Recommendations

Based on the findings discussed throughout this chapter, several recommendations can be drawn for organisations considering LLM-based triage. GPT-5-mini under strict playbook guidance is the recommended configuration for initial deployment, given its combination of accuracy, tool compliance and processing speed. Additionally, a human-in-the-loop or advisory deployment is suggested, consistent with the preference expressed by 93% of survey participants. To address the verifiability concern driving the lower trust scores, it is advised to implement direct references to the source evidence fields within the ticketing platform, such as links to the specific SIEM log events that informed the classification. Furthermore, confidence markers and escalation recommendations for ambiguous alert instances should be incorporated to better support analyst decision-making. Organisations should also route API calls through an enterprise AI platform, such as Microsoft Azure AI Foundry [77] rather than the standard public OpenAI API, as discussed in Section 6.1.

6 Limitations and Future Work

This chapter outlines the constraints that affected the scope of the study and directions for extending and improving the framework in the future.

6.1 Limitations

Several limitations constrain the generalisability and completeness of the findings in this thesis.

Firstly, the data used for evaluation consists of 100 synthetically generated alerts covering 8 categories. While the alerts were designed to reflect real SOC patterns based on the author's experience at Neverhack Estonia, they lack the full contextual depth of live telemetry. Various fields that study participants expected to see in the investigation notes, including MFA status, device management information, and file origin context, were not present in the enrichment data, such as recent events or recent similar alerts. This shows that some model omissions highlighted in the survey reflect dataset limitations rather than model deficiencies.

Besides the dataset, the model was evaluated in a synthetic environment without direct integration with live SOC tools. In a production setting, the triage workflow would need to connect to a SIEM for real-time log retrieval, an EDR platform for endpoint telemetry and ticketing systems such as JIRA [78] for investigation note submission. The absence of these integrations in the current implementation means that the tool calls for recent events, similar alerts and user lookups relied on pre-existing local JSON files instead of live queries against actual data sources. This limits the full visibility of the evaluation and implies that processing times, enrichment quality and classification accuracy may differ in live deployment, where data retrieval involves system latency and variability in data completeness. However, even accounting for additional latency introduced by live queries, the overall delay is unlikely to be significant enough to close the gap between model and analyst triage time.

Closely related to the previous limitation, the current implementation was scoped to static enrichment files to maintain a controlled evaluation environment. Running live queries against a SIEM or EDR was outside the scope of this study, as the main focus was on evaluating the classification and reasoning capabilities of the LLM under structured playbook guidance rather than integration architecture. This represents a boundary between the research prototype and production-ready deployment and does not reflect a technical constraint of the framework itself.

Furthermore, the study did not apply fine-tuning to the evaluated models. While this was a deliberate design choice to demonstrate the feasibility of the zero-shot and low-cost approach, fine-tuning on domain-specific SOC data could improve classification accuracy and reduce FPR.

It is worth noting that the evaluation was conducted using the standard OpenAI API, which would not be suitable for processing client alert data in a real SOC environment due to confidentiality requirements. To fulfil this requirement, Microsoft Azure AI Foundry can be utilised to access the same GPT model family under enterprise privacy terms, where prompts and outputs are not used for training, and are processed within Microsoft's Azure environment in the customer-specified geography [77].

In addition to the LLM API cost, the triage framework relies on external enrichment tools that were used under free-tier API plans during this evaluation. Specifically, VirusTotal was used for hash lookups under its free public API, which allows 500 requests per day at a rate of 4 requests per minute [79]. AbuseIPDB was used for IP reputation checks under its free tier, which allows 1000 requests per day [80]. VPN API (vpnapi.io) was used for VPN and proxy detection under its free plan, which allows 1000 requests per day [81]. These limits were sufficient for the scope of this study. However, in a production SOC environment with a higher volume of alerts, the free-tier limits may be exceeded, and premium API plans with their associated costs would need to be factored into the overall operational cost of the framework. Additionally, the LLM-related cost is directly influenced by token usage, which scales with the size of the alert and the amount of enrichment data included. Larger or more detailed alerts therefore result in higher processing costs, meaning that the per-alert cost observed in this study may increase in real-world deployments depending on alert complexity and enrichment volume.

Beyond the cost considerations above, each alert was evaluated once per configuration and the effect of output variability across repeated queries was not assessed. Due to the probabilistic nature of LLMs, a degree of non-determinism is inherent, meaning that the same alert may produce different classifications or investigation notes across runs.

In terms of participant scope, the survey was conducted with 14 analysts from a single organisation, Neverhack Estonia. The findings therefore reflect the perspective of that organisation and may not generalise directly to SOCs with different tools, client profiles or triage processes. Additionally, the scope of the alerts presented to each participant was limited to one alert per category across five of the eight available categories, as a full-scale evaluation covering all categories would have made the study significantly more time-consuming and burdensome for the respondents.

Finally, this thesis assesses only two models from the GPT-5 family under three operational modes. It does not examine any other model families or retrieval-augmented generation (RAG).

6.2 Future Work

The limitations outlined in the previous subchapter point to several directions for extending and improving the framework.

The most immediate priority is to evaluate the framework with locally hosted open-weight models, such as LLaMA or Mistral, to address the data privacy constraint and enable deployment with real client alert data. This would also allow a direct comparison with commercially hosted models.

A further significant direction is integrating the framework with live SOC tools. This would involve connecting the triage pipeline directly to a SIEM such as Elastic Security [82], an EDR solution, and a ticketing system such as JIRA, so the model can retrieve real event data, submit investigation notes and update ticket status without manual analyst intervention. As part of this integration, the toolset should include live query execution. This can be achieved in two ways. The LLM can supply structured parameters, such as usernames, IP addresses, or hostnames to predefined SIEM or EDR queries. Alternatively, it can issue dynamic queries based on alert context to extract relevant data.

Alongside the integration, fine-tuning the models on domain-specific SOC alert data is another promising direction. The current zero-shot approach demonstrates that competitive performance is achievable without fine-tuning. However, targeted adaptation on labelled triage data from a production environment could improve classification accuracy in the weakest-performing alert categories, particularly EDR and DNS Tunnelling Anomaly. It is worth noting that this approach incurs additional costs, particularly for data preparation, training, and maintenance.

Survey results highlighted evidence presentation and verifiability as the primary trust barrier. Future work should explore features that include direct links or references to the source events, along with confidence markers and escalation recommendations for ambiguous alert instances.

Output consistency across repeated queries should also be evaluated in future work by submitting the same alert multiple times and measuring classification agreement across runs. Findings from such an evaluation would help determine whether output variability poses a practical concern in production deployment.

Finally, the expansion of the evaluation to a larger and more diverse alert dataset, including more alert categories and a higher number of instances across varied client environments, would improve the statistical reliability of the results. It would also provide a better basis for assessing the framework's ability to generalise beyond the alert categories examined in this study.

7 Conclusion

This thesis presents a playbook-guided, multi-step LLM framework for automating Tier 1 SOC alert triage. The main contribution lies in a low-cost and reproducible design that employs explicit reasoning steps aligned with SOC playbooks, a zero-shot prompting strategy using publicly available pre-trained models and a dual evaluation framework combining quantitative metrics with qualitative analyst assessments. Unlike prior work that relies on fine-tuned models or treats LLMs as black-box classifiers, this approach makes the system replicable and cost-efficient, particularly for smaller SOCs, and bridges the gap between academic LLM experimentation and operational SOC practice.

RQ1 addresses the performance and cost of LLMs in security alert triage. The quantitative evaluation across 100 alerts and 6 configurations showed that GPT-5-mini under strict playbook guidance achieved the strongest result with 95% accuracy, 94.12% precision, 96% recall, an F1 score of 95.05% and an FPR of 6%. GPT-5-mini showed consistent improvement as guidance became stricter, while GPT-5-nano responded well to soft guidance but degraded under strict guidance due to tool omissions in enrichment-dependent alert categories. The TCR reached 94% for GPT-5-mini strict, confirming that stricter directive language produces more reliable tool invocation behaviour. On average, all 6 configurations completed triage in under 48 seconds, representing a substantial throughput advantage over the analyst survey estimates of 12 to 21 minutes per alert. In terms of cost, GPT-5-nano is the more affordable option across all modes, ranging from $0.00144 per alert in general mode to $0.00205 in strict mode. GPT-5-mini ranges from $0.00440 in general mode to $0.00642 in strict mode. Despite the higher per-alert cost, GPT-5-mini strict delivers considerably better classification performance, making it the recommended configuration where accuracy is the primary concern. Both models confirm the low-cost premise of this thesis, demonstrating that effective SOC alert triage can be achieved at a fraction of the cost associated with flagship models.

RQ2 targets how LLM-driven triage addresses the workload and decision-making challenges faced by SOC analysts. The analyst survey conducted with 14 SOC professionals at Neverhack Estonia provided direct empirical evidence. Participants

recorded a mean workload reduction score of 4.00 out of 5, with the majority agreeing that the system could meaningfully reduce their workload. Open-ended responses also supported this finding. One participant specifically highlighted that the system reduces time spent writing investigation notes and increases focus on analysis. Decision-making support was also mentioned. Respondents pointed to automated evidence aggregation, historical context and the linking of technical indicators with business context as particularly useful. A total of 93% of participants preferred advisory or human-in-the-loop deployment, confirming the system is best positioned as a decision support tool that reduces manual effort while preserving analyst oversight.

Together, **RQ1** and **RQ2** answer the **MRQ** of how LLMs can contribute to the automation of Tier 1 SOC alert triage. The results show that LLMs can indeed contribute by taking over the classification and investigation note generation tasks that currently occupy a significant portion of analyst time, doing so accurately and within a fraction of the time a human analyst requires. When guided by structured playbooks, the models classify alerts with up to 95% accuracy and process each alert in under a minute, while analysts perceive the outputs as useful for both decision-making and workload reduction. This positions LLMs as a viable automation layer for the initial triage stage, handling routine alert processing and surfacing structured evidence for analyst review rather than replacing analyst judgement entirely.

The key limitations of this study include the use of a synthetic dataset, the absence of live SOC tool integration, the restriction to a single organisation for the analyst survey and the evaluation of only two models under a zero-shot approach. These constraints affect the generalisability of the findings and should be interpreted as indicative of the framework's potential rather than as a direct measure of production performance.

Further work should prioritise a production pilot using live alert data with direct SIEM, EDR and ticketing system integration, including live query execution. Evaluating the framework with locally hosted open-weight models would address data privacy constraints. Fine-tuning on domain-specific SOC data, improving output verifiability through direct evidence links and confidence markers, and expanding to a larger, more diverse alert dataset represent the remaining directions for extending this work. Overall, the results demonstrate that a zero-shot, playbook-guided LLM framework can replicate key aspects of Tier 1 SOC triage at low cost and meaningfully reduce analyst workload,

addressing one of the primary drivers of alert fatigue, though further research and live deployment validation are required before it can be considered fully production-ready.

References

[1] "Orca Security 2022 Cloud Security Alert Fatigue Report." Accessed: Nov. 08, 2025. [Online]. Available: https://orca.security/resources/blog/2022-cloud-cyber-security-alert-fatigue-report/

[2] "Preventing Alert Fatigue in Cybersecurity: How To Recognize & Combat Alert Fatigue | Splunk." Accessed: Nov. 08, 2025. [Online]. Available: https://www.splunk.com/en_us/blog/learn/alert-fatigue.html

[3] "The Evolution of the SOC: From Traditional SOC to AI-Powered SOC 3.0." Accessed: Nov. 08, 2025. [Online]. Available: https://blog.axur.com/en-us/evolution-of-soc-ai-driven-soc-3-0

[4] B. A. Alahmadi, L. Axon, and I. Martinovic, "99% False Positives: A Qualitative Study of SOC Analysts' Perspectives on Security Alarms," *USENIX Security Symposium*, 2022, Accessed: Apr. 12, 2026. [Online]. Available: https://www.usenix.org/conference/usenixsecurity22/presentation/alahmadi

[5] H. Xu *et al.*, "Large Language Models for Cyber Security: A Systematic Literature Review," *ACM Transactions on Software Engineering and Methodology*, vol. 1, May 2024, https://doi.org/10.1145/3769676.

[6] M. Vielberth, F. Böhm, and I. Fichtinger, "Security Operations Center: A Systematic Study and Open Challenges," *IEEE Access*, vol. 8, 2020, https://doi.org/10.1109/ACCESS.2020.3045514.

[7] R. Bidou, "Security Operation Center Concepts & Implementation," 2005. [Online]. Available: https://www.researchgate.net/publication/228587242

[8] A. Basta, N. Basta, W. Anwar, and M. I. Essar, "Open-Source Security Operations Center (SOC) (etc.) (Z-Library)," 2024, Accessed: Oct. 16, 2025. [Online]. Available: https://unidel.edu.ng/focelibrary/books/Open-

Source%20Security%20Operations%20Center%20(SOC)%20(%20etc.)%20(Z-Library).pdf

[9] J. Mlodzianowski, O. Santos, and R. Taylor, *CompTIA Security+ SY0-601 Cert Guide*, 5th edition. Pearson IT Certification, 2021. Accessed: Feb. 26, 2026. [Online]. Available: https://www.pearsonitcertification.com/articles/article.aspx?p=3128871&seqNum=3

[10] L. Kersten *et al.*, "A Security Alert Investigation Tool Supporting Tier 1 Analysts in Contextualizing and Understanding Network Security Events," in *Proceedings - Annual Computer Security Applications Conference, ACSAC*, Association for Computing Machinery, 2024, pp. 890–905. https://doi.org/10.1109/ACSAC63791.2024.00076.

[11] S. Tariq, M. B. Chhetri, S. Nepal, and C. Paris, "Alert Fatigue in Security Operations Centres: Research Challenges and Opportunities," *ACM Comput. Surv.*, vol. 57, no. 9, Apr. 2025, https://doi.org/10.1145/3723158.

[12] L. Kersten, T. Mulders, E. Zambon, C. Snijders, and L. Allodi, "'Give Me Structure': Synthesis and Evaluation of a (Network) Threat Analysis Process Supporting Tier 1 Investigations in a Security Operation Center," 2023, pp. 97–111. Accessed: Oct. 16, 2025. [Online]. Available: www.usenix.org/conference/soups2023/presentation/kersten

[13] S. Chandran Sundaramurthy *et al.*, "A Human Capital Model for Mitigating Security Analyst Burnout," 2015, Accessed: Apr. 12, 2026. [Online]. Available: https://www.usenix.org/system/files/conference/soups2015/soups15-paper-sundaramurthy.pdf

[14] J. Lemon, "SANS 2024 Detection and Response Survey Transforming Cybersecurity Operations: AI, Automation, and Integration in Detection and Response," 2024. Accessed: Oct. 16, 2025. [Online]. Available: https://21984718.fs1.hubspotusercontent-na1.net/hubfs/21984718/Content/Survey_2024-Detection-Response_Prelude%20(1).pdf

[15] E. Hinchy, “Voice of the SOC,” 2023. Accessed: Oct. 16, 2025. [Online]. Available: https://www.tines.com/reports/voice-of-the-soc-2023/

[16] A. Mareedu, “Autonomous Security Operations Centers (SOC): AI Agents for Threat Triage, Response, and Orchestration,” *International Journal of Emerging Research in Engineering and Technology*, vol. 6, no. 2, pp. 63–70, May 2025, https://doi.org/10.63282/3050-922X.IJERET-V6I2P108.

[17] “Large Language Models powered by world-class Google AI.” Accessed: Feb. 26, 2026. [Online]. Available: https://cloud.google.com/ai/llms

[18] “What is a Large Language Model (LLM)? | SAP.” Accessed: Feb. 26, 2026. [Online]. Available: https://www.sap.com/lithuania/resources/what-is-large-language-model

[19] S. Gupta *et al.*, “The Evolution and Future of Generative AI From Theory to Practice,” 2024, Accessed: Feb. 26, 2026. [Online]. Available: https://www.swamivivekanandauniversity.ac.in/backend/resource/assets/images/book/1736929685The%20Evolution%20and%20Future%20of%20Generative%20AI%20From%20Theory%20to%20Practice-1.pdf

[20] A. Vaswani *et al.*, “Attention Is All You Need,” 2017. https://doi.org/10.48550/arXiv.1706.03762.

[21] “ChatGPT.” Accessed: Feb. 26, 2026. [Online]. Available: https://chatgpt.com/

[22] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, “Improving Language Understanding by Generative Pre-Training,” 2018. Accessed: Feb. 26, 2026. [Online]. Available: https://cdn.openai.com/research-covers/language-unsupervised/language_understanding_paper.pdf

[23] T. B. Brown *et al.*, “Language Models are Few-Shot Learners,” *Adv. Neural Inf. Process. Syst.*, May 2020, https://doi.org/10.48550/arXiv.2005.14165.

[24] “The AI for Problem Solvers | Claude by Anthropic.” Accessed: Feb. 26, 2026. [Online]. Available: https://claude.com/product/overview

[25] “Industry Leading, Open-Source AI | Llama.” Accessed: Feb. 26, 2026. [Online]. Available: https://www.llama.com/

[26] “Frontier AI LLMs, assistants, agents, services | Mistral AI.” Accessed: Feb. 26, 2026. [Online]. Available: https://mistral.ai/

[27] “Gemini 3 — Google DeepMind.” Accessed: Feb. 26, 2026. [Online]. Available: https://deepmind.google/models/gemini/

[28] “What Are AI Hallucinations? | IBM.” Accessed: Feb. 26, 2026. [Online]. Available: https://www.ibm.com/think/topics/ai-hallucinations

[29] Y. Shanmugarasa, M. Ding, M. A. P. Chamikara, and T. Rakotoarivelo, “SoK: The Privacy Paradox of Large Language Models: Advancements, Privacy Risks, and Mitigation,” *Proceedings of the ACM Conference on Computer and Communications Security*, vol. 1, pp. 425–441, Jun. 2025, https://doi.org/10.1145/3708821.3733888.

[30] R. Singh *et al.*, “LLMs in the SOC: An Empirical Study of Human-AI Collaboration in Security Operations Centres,” Aug. 2025, https://doi.org/10.48550/arXiv.2508.18947.

[31] M. B. Chhetri, S. Tariq, R. Singh, F. Jalalvand, C. Paris, and S. Nepal, “Towards Human-AI Teaming to Mitigate Alert Fatigue in Security Operations Centres,” *ACM Trans. Internet Technol.*, vol. 24, no. 3, Jul. 2024, https://doi.org/10.1145/3670009.

[32] R. Molleti, V. Goje, P. Luthra, and P. Raghavan, “Automated threat detection and response using LLM agents,” *World Journal of Advanced Research and Reviews*, vol. 24, no. 2, pp. 079–090, Nov. 2024, https://doi.org/10.30574/wjarr.2024.24.2.3329.

[33] F. Trad and A. Chehab, “Prompt Engineering or Fine-Tuning? A Case Study on Phishing Detection with Large Language Models,” *Mach. Learn. Knowl. Extr.*, vol. 6, no. 1, pp. 367–384, Mar. 2024, https://doi.org/10.3390/make6010018.

[34] M. A. Ferrag *et al.*, “Revolutionizing Cyber Threat Detection with Large Language Models: A Privacy-Preserving BERT-Based Lightweight Model for IoT/IIoT Devices,” *IEEE Access*, vol. 12, pp. 23733–23750, 2024, https://doi.org/10.1109/ACCESS.2024.3363469.

[35] D. Du *et al.*, “MAD-LLM: A Novel Approach for Alert-Based Multi-stage Attack Detection via LLM,” in *Proceedings - 2024 IEEE International Symposium on Parallel and Distributed Processing with Applications, ISPA 2024*, Institute of Electrical and Electronics Engineers Inc., 2024, pp. 2046–2053. https://doi.org/10.1109/ISPA63168.2024.00279.

[36] M. Sharif *et al.*, “DrSec: Flexible Distributed Representations for Efficient Endpoint Security,” *Proc. IEEE Symp. Secur. Priv.*, pp. 3609–3624, 2024, Accessed: Apr. 12, 2026. [Online]. Available: https://ieeexplore.ieee.org/document/10646721

[37] Z. Wang *et al.*, “Large Language Models Can Provide Accurate and Interpretable Incident Triage,” in *Proceedings - International Symposium on Software Reliability Engineering, ISSRE*, IEEE Computer Society, 2024, pp. 523–534. https://doi.org/10.1109/ISSRE62328.2024.00056.

[38] “Agentic SOC | AI Agents for Alert Triage & Threat Hunting | Dropzone AI.” Accessed: Oct. 17, 2025. [Online]. Available: https://www.dropzone.ai/

[39] B. Wei *et al.*, “CORTEX: Collaborative LLM Agents for High-Stakes Alert Triage,” 2025, https://doi.org/10.48550/arXiv.2510.00311.

[40] “Radiant AI SOC Platform | Scale and Optimize Security Operations.” Accessed: Oct. 17, 2025. [Online]. Available: https://radiantsecurity.ai/

[41] “Radiant Security AI Review – Cost, Use Cases & Alternatives [2025].” Accessed: Oct. 17, 2025. [Online]. Available: https://aichief.com/ai-detectors/radiant-security-ai/

[42] H. Baron, K. Seifried, S. Lumpe, and S. Smith, “Beyond the Hype: A Benchmark Study of Al Agents in the SOC,” 2025, Accessed: Oct. 17, 2025. [Online].

Available: https://cloudsecurityalliance.org/artifacts/a-benchmark-study-of-ai-agents-in-the-soc

[43] "Radiant Security Al-powered SOC Co-pilot." Accessed: Oct. 17, 2025. [Online]. Available: https://20705827.fs1.hubspotusercontent-na1.net/hubfs/20705827/Docs/Radiant%20Security%20Solution%20Brief.pdf

[44] L. Chavali, A. Krishnan, P. Saxena, B. Mitra, and A. Chivukula, "Off-policy actor-critic deep reinforcement learning methods for alert prioritization in intrusion detection systems," *Comput. Secur.*, vol. 142, p. 103854, Jul. 2024, https://doi.org/10.1016/J.COSE.2024.103854.

[45] X. Wang, X. Yang, X. Liang, X. Zhang, W. Zhang, and X. Gong, "Combating alert fatigue with AlertPro: Context-aware alert prioritization using reinforcement learning for multi-step attack detection," *Comput. Secur.*, vol. 137, p. 103583, Feb. 2024, https://doi.org/10.1016/J.COSE.2023.103583.

[46] P. Danquah, "Security Operations Center: A Framework for Automated Triage, Containment and Escalation," *Journal of Information Security*, vol. 11, no. 04, pp. 225–240, 2020, https://doi.org/10.4236/jis.2020.114015.

[47] B. Filar, R. J. Seymour, and M. Park, "Ask Me Anything: A Conversational Interface to Augment Information Security Workers," 2017, https://doi.org/10.48550/arXiv.1707.05768.

[48] V. H. Perera, A. N. Senarathne, and L. Rupasinghe, "Intelligent SOC Chatbot for Security Operation Center," *2019 International Conference on Advancements in Computing, ICAC 2019*, pp. 340–345, Dec. 2019, https://doi.org/10.1109/ICAC49085.2019.9103388.

[49] "ChatOps for AWS – AWS Chatbot – Amazon Web Services." Accessed: Oct. 17, 2025. [Online]. Available: https://aws.amazon.com/chatbot/

[50] "VirusTotal - Home." Accessed: Oct. 17, 2025. [Online]. Available: https://www.virustotal.com/gui/home/upload

[51] M. Kaheh, D. K. Kholgh, and P. Kostakos, “Cyber Sentinel: Exploring Conversational Agents in Streamlining Security Tasks with GPT-4,” Sep. 2023, https://doi.org/10.48550/arXiv.2309.16422.

[52] “OpenAI.” Accessed: Oct. 17, 2025. [Online]. Available: https://openai.com/

[53] V. Jüttner, M. Grimmer, and E. Buchmann, “ChatIDS: Explainable Cybersecurity Using Generative AI,” Jun. 2023, https://doi.org/10.48550/arXiv.2306.14504.

[54] Ismail *et al.*, “Enhancing Security Operations Center: Wazuh Security Event Response with Retrieval-Augmented-Generation-Driven Copilot,” *Sensors*, vol. 25, no. 3, Feb. 2025, https://doi.org/10.3390/s25030870.

[55] “Pixtral Large | Mistral AI.” Accessed: Oct. 17, 2025. [Online]. Available: https://mistral.ai/news/pixtral-large

[56] “Wazuh - Open Source XDR. Open Source SIEM.” Accessed: Oct. 17, 2025. [Online]. Available: https://wazuh.com/

[57] J. W. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, 4th edition. Sage Publications, 2014.

[58] J. Lazar, J. H. Feng, and H. Hochheiser, *Research Methods in Human-Computer Interaction*, 2nd edition. Morgan Kaufmann Publishers, 2017.

[59] “NEVERHACK.” Accessed: Apr. 12, 2026. [Online]. Available: https://neverhack.ee/

[60] “WHOIS Search, Domain Name, Website, and IP Tools - Who.is.” Accessed: Apr. 11, 2026. [Online]. Available: https://who.is/

[61] “Introducing GPT-5.2 | OpenAI.” Accessed: Feb. 28, 2026. [Online]. Available: https://openai.com/index/introducing-gpt-5-2/

[62] T. Patwardhan *et al.*, “GDPval: Evaluating AI Model Performance on Real-World Economically Valuable Tasks,” Oct. 2025, https://doi.org/10.48550/arXiv.2510.04374.

[63] “Latest AI Models – AGENTVSAI.” Accessed: Feb. 28, 2026. [Online]. Available: https://www.agentvsai.com/latest-ai-models/

[64] “Pricing | OpenAI API.” Accessed: Feb. 28, 2026. [Online]. Available: https://developers.openai.com/api/docs/pricing?latest-pricing=standard

[65] “Introducing GPT-5 for developers | OpenAI.” Accessed: Feb. 28, 2026. [Online]. Available: https://openai.com/index/introducing-gpt-5-for-developers/

[66] S. Ermut and Ş. Alper, “LLM Parameters: GPT-5 High, Medium, Low and Minimal.” Accessed: Feb. 28, 2026. [Online]. Available: https://research.aimultiple.com/llm-parameters/

[67] D. Rein *et al.*, “GPQA: A Graduate-Level Google-Proof Q&A Benchmark,” Nov. 2023, https://doi.org/10.48550/arXiv.2311.12022.

[68] L. Phan *et al.*, “A benchmark of expert-level academic questions to assess AI capabilities,” *Nature 2026 649:8099*, vol. 649, no. 8099, pp. 1139–1146, Jan. 2026, https://doi.org/10.1038/s41586-025-09962-4.

[69] V. Barres, H. Dong, S. Ray Sierra soham, sierraai Xujie Si, and K. Narasimhan Sierra, “Tau^2-Bench: Evaluating Conversational Agents in a Dual-Control Environment,” Jun. 2025, https://doi.org/10.48550/arXiv.2506.07982.

[70] “GPT-5 mini (high) vs GPT-5 mini (medium): Model Comparison.” Accessed: Feb. 28, 2026. [Online]. Available: https://artificialanalysis.ai/models/comparisons/gpt-5-mini-vs-gpt-5-mini-medium

[71] “GPT-5 nano (medium) vs GPT-5 nano (high): Model Comparison.” Accessed: Feb. 28, 2026. [Online]. Available: https://artificialanalysis.ai/models/comparisons/gpt-5-nano-medium-vs-gpt-5-nano

[72] “AI Model & API Providers Analysis | Artificial Analysis.” Accessed: Feb. 28, 2026. [Online]. Available: https://artificialanalysis.ai/

[73] “AbuseIPDB - IP address abuse reports - Making the Internet safer, one IP at a time.” Accessed: Apr. 11, 2026. [Online]. Available: https://www.abuseipdb.com/

[74] “VPN & Proxy Detection API.” Accessed: Apr. 11, 2026. [Online]. Available: https://vpnapi.io/

[75] Ž. Vujović, “Classification Model Evaluation Metrics,” *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 6, pp. 599–606, 2021, https://doi.org/10.14569/IJACSA.2021.0120670.

[76] M. Sokolova and G. Lapalme, “A systematic analysis of performance measures for classification tasks,” *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, Jul. 2009, https://doi.org/10.1016/J.IPM.2009.03.002.

[77] “Data, privacy, and security for Azure Direct Models in Microsoft Foundry - Microsoft Foundry | Microsoft Learn.” Accessed: Apr. 08, 2026. [Online]. Available: https://learn.microsoft.com/en-us/azure/foundry/responsible-ai/openai/data-privacy?tabs=azure-portal

[78] “Jira | Project Management for the AI Era | Atlassian.” Accessed: Apr. 23, 2026. [Online]. Available: https://www.atlassian.com/software/jira

[79] “Public vs Premium API.” Accessed: Apr. 11, 2026. [Online]. Available: https://docs.virustotal.com/reference/public-vs-premium-api

[80] “API Documentation - AbuseIPDB.” Accessed: Apr. 11, 2026. [Online]. Available: https://www.abuseipdb.com/api.html

[81] “IP Address Data API Pricing | VPNAPI.io.” Accessed: Apr. 11, 2026. [Online]. Available: https://vpnapi.io/pricing

[82] “Elastic — The Search AI Company | Elastic.” Accessed: May 02, 2026. [Online]. Available: https://www.elastic.co/

Appendix 1 – Non-exclusive licence for reproduction and publication of a graduation thesis[1]

I, Sanan Mammadli

1 grant Tallinn University of Technology free licence (non-exclusive licence) for my thesis “Reducing Alert Fatigue in Security Operations Centres: LLM-Based Intelligent Alert Triage”, supervised by Ahmed Nagi Abdelaziz Mohamed Nasr

1.1 to be reproduced for the purposes of preservation and electronic publication of the graduation thesis, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright;

1.2 to be published via the web of Tallinn University of Technology, incl. to be entered in the digital collection of the library of Tallinn University of Technology until expiry of the term of copyright.

2 I am aware that the author also retains the rights specified in clause 1 of the non-exclusive licence.

3 I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights, the rights arising from the Personal Data Protection Act or rights arising from other legislation.

[1] The non-exclusive licence is not valid during the validity of access restriction indicated in the student's application for restriction on access to the graduation thesis that has been signed by the school's dean, except in case of the university's right to reproduce the thesis for preservation purposes only. If a graduation thesis is based on the joint creative activity of two or more persons and the co-author(s) has/have not granted, by the set deadline, the student defending his/her graduation thesis consent to reproduce and publish the graduation thesis in compliance with clauses 1.1 and 1.2 of the non-exclusive licence, the non-exclusive license shall not be valid for the period.

Appendix 2 – Source Code and Research Artefacts

The source code, playbooks, alert configurations, supplementary scripts, evaluation logs and final investigation notes produced in this study are available in the following GitHub repository: https://github.com/SananMS/LLM-SOC-Analyst

Appendix 3 – Neverhack Estonia Analyst Survey

Section A: Participant Background

1. **How many years of experience do you have working in a SOC or security operations role?**
    - Less than 1 year
    - 1–3 years
    - 3–5 years
    - More than 5 years
2. **What is your primary role within the SOC team?**
    - Junior Analyst
    - Mid-level Analyst
    - Senior Analyst
3. **On average, approximately how many alerts do you triage per shift? Please consider that this may vary across day, evening and night shifts given the rotation schedule.**
    - Fewer than 10
    - 10–25
    - 25–50
    - More than 50
4. **How frequently do you experience alert fatigue in your work?**
    - Never

- Rarely
- Sometimes
- Often
- Always

Section B: Current Triage Practice

1. **What proportion of alerts you triage would you estimate are false positives?**
    - Less than 25%
    - 25–50%
    - 50–75%
    - More than 75%
2. **Which stages of the triage process do you find most time-consuming?** ***(Select all that apply)***
    - Log and event correlation
    - Accessing relevant environments and tools
    - External tool lookups
    - Documentation and note-writing
    - Escalation decision
    - Other

Section C: Evaluation of LLM-Generated Investigation Notes

Before proceeding, please read the following classification definitions used in this study:

High Priority (HP): Alerts that represent confirmed threats or contain sufficient indicators to require escalation or further investigation.

Low Priority (LP): Alerts that are either false positives or represent authorised activity that does not require escalation.

To support the investigation of each alert, the model has access to the following tools:

- *check_ip_reputation - checks the reputation of a source or destination IP address for known malicious activity*
- *check_hash_reputation - verifies whether a file hash is associated with known malware (VirusTotal)*
- *get_domain_age - retrieves the creation date of a domain to identify recently registered infrastructure*
- *lookup_users - searches the internal employee directory to verify user identity and organisational role*
- *get_recent_events - returns prior logs from the same host or account before the alert was triggered*
- *get_recent_similar_alerts - retrieves previously flagged similar alerts from the same entity or broader environment*

Please note that the data returned by the last four tools is synthetically generated as part of the dataset. Additionally, not all tools may have been invoked for every alert, as their usage depends on the relevance of the tool to the specific alert content based on the corresponding playbook. Furthermore, even when invoked, some tools may return no results, as certain alerts may have no associated contextual data.

For each of the five alerts below, you will first be shown the raw alert data and any associated contextual information. Please review this carefully and form your own initial judgement before reading the investigation note. The following questions are repeated for each of the five alerts.

(Please rate your agreement with the following statements on a scale of 1 to 5, where 1 = Strongly Disagree, 2 = Disagree, 3 = Neutral, 4 = Agree, 5 = Strongly Agree)

1. **The investigation note accurately captures the key findings of this alert.**
   - 1 2 3 4 5
2. **The classification assigned to this alert aligns with the HP and LP criteria defined at the start of this section.**
   - 1 2 3 4 5
3. **The description is written clearly and professionally, suitable for a SOC ticketing system.**
   - 1 2 3 4 5
4. **I would trust this output to support a triage decision without further investigation.**
   - 1 2 3 4 5
5. **The investigation note contains sufficient reasoning to explain the classification.**
   - 1 2 3 4 5
6. **Did this investigation note omit any critical information you would expect a Tier 1 analyst to document?**
   - Yes
   - No

If yes, please describe what was missing: **_____**

7. **Approximately how long would you estimate it would take you to triage this specific alert end-to-end, accounting for accessing relevant environments, reviewing logs and contextual data, performing tool lookups and writing the final investigation notes?**

Section D: Perceived Workload and Decision-Making Impact

(Please rate your agreement with the following statements on a scale of 1 to 5, where 1 = Strongly Disagree, 2 = Disagree, 3 = Neutral, 4 = Agree, 5 = Strongly Agree)

1. **LLM-generated triage notes of this quality could meaningfully reduce my current workload.**
    - 1 2 3 4 5
2. **I would be comfortable relying on outputs like these when making escalation decisions.**
    - 1 2 3 4 5
3. **What level of autonomy would you be comfortable with for a system like this in a production environment?**
    - Fully autonomous - no analyst review required
    - Human-in-the-loop - analyst reviews before triage
    - Advisory only - analyst makes all final decisions
    - I would not be comfortable using this system
4. **Briefly, what aspects of the investigation notes did you find most useful for operational decision-making? (Open Question)**

---

5. **Briefly, what improvements would be necessary before you would consider a system like this suitable for production use? (Open Question)**

Appendix 4 – Survey Quantitative Results

This appendix presents the full quantitative results from the survey (n=14). Categorical question responses are shown as frequency counts and percentages. Likert-scale responses are reported as mean, minimum and maximum per item on a 5-point scale where 1 = Strongly Disagree and 5 = Strongly Agree.

Section A: Participant Background

Years of SOC Experience (How many years of experience do you have working in a SOC or security operations role?)

Experience	Count	%
Less than 1 year	4	29%
1 to 3 years	7	50%
3 to 5 years	2	14%
More than 5	1	7%

Primary Role (What is your primary role within the SOC team?)

Role	Count	%
Junior Analyst	6	43%
Mid-level Analyst	5	36%
Senior Analyst	3	21%

Alerts Triaged per Shift (On average, approximately how many alerts do you triage per shift? Please consider that this may vary across day, evening and night shifts given the rotation schedule.)

Alerts per Shift	Count	%
Fewer than 10	5	36%
10 to 25	8	57%
25 to 50	1	7%

Alert Fatigue Frequency (How frequently do you experience alert fatigue in your work?)

Frequency	Count	%
Never	1	7%
Rarely	1	7%
Sometimes	8	57%
Often	4	29%

Section B: Triage Baseline

Estimated False Positive Proportion (What proportion of alerts you triage would you estimate are false positives?)

FP Proportion	Count	%
25 to 50%	1	7%
50 to 75%	4	29%
More than 75%	9	64%

Most Time-Consuming Triage Stages (Which stages of the triage process do you find most time-consuming? *(Select all that apply))*

Stage	Selected by
Log and event correlation	10
External tool lookups	6
Documentation and note-writing	6
Accessing relevant environments and	5
Triage decision	4
Other	1

Sections C and D: Likert Scale Results

Likert Scale Results: Sections C and D

No.	Question (Shortened)	Mean	Min	Max	SD
Alert 1: Data Exfiltration Anomaly (HP)					
1	**Accuracy:** "The investigation note accurately captures the key findings of this alert."	4.64	4	5	0.5
2	**Classification:** "The classification aligns with the HP and LP criteria."	4.50	3	5	0.65
3	**Clarity:** "The description is written clearly and professionally."	4.71	4	5	0.47
4	**Trust:** "I would trust this output to support a triage decision without further investigation."	3.57	2	5	0.94
5	**Reasoning:** "The investigation note contains sufficient reasoning to explain the classification."	4.57	4	5	0.51
Alert 2: Suspicious PowerShell Script (LP)					
1	**Accuracy:** "The investigation note accurately captures the key findings of this alert."	4.71	4	5	0.47
2	**Classification:** "The classification aligns with the HP and LP criteria."	4.79	4	5	0.43
3	**Clarity:** "The description is written clearly and professionally."	4.79	4	5	0.43

4	**Trust:** "I would trust this output to support a triage decision without further investigation."	4.00	3	5	0.78
5	**Reasoning:** "The investigation note contains sufficient reasoning to explain the classification."	4.71	4	5	0.47
Alert 3: Suspicious Login (HP)					
1	**Accuracy:** "The investigation note accurately captures the key findings of this alert."	4.64	4	5	0.5
2	**Classification:** "The classification aligns with the HP and LP criteria."	4.79	4	5	0.43
3	**Clarity:** "The description is written clearly and professionally."	4.50	3	5	0.76
4	**Trust:** "I would trust this output to support a triage decision without further investigation."	3.93	2	5	1.07
5	**Reasoning:** "The investigation note contains sufficient reasoning to explain the classification."	4.71	4	5	0.47
Alert 4: EDR Malware (LP)					
1	**Accuracy:** "The investigation note accurately captures the key findings of this alert."	4.00	3	5	0.68
2	**Classification:** "The classification aligns with the HP and LP criteria."	4.14	3	5	0.86
3	**Clarity:** "The description is written clearly and professionally."	4.14	3	5	0.95
4	**Trust:** "I would trust this output to support a triage decision without further investigation."	2.86	2	5	0.86
5	**Reasoning:** "The investigation note contains sufficient reasoning to explain the classification."	4.14	3	5	0.95
Alert 5: Threat Intelligence: Leaked Documents (HP)					
1	**Accuracy:** "The investigation note accurately captures the key findings of this alert."	4.71	4	5	0.47
2	**Classification:** "The classification aligns with the HP and LP criteria."	4.79	4	5	0.43
3	**Clarity:** "The description is written clearly and professionally."	4.57	3	5	0.65
4	**Trust:** "I would trust this output to support a triage decision without further investigation."	4.07	1	5	1.21

5	**Reasoning:** "The investigation note contains sufficient reasoning to explain the classification."	4.64	4	5	0.5
Section D: Workload and Autonomy Perception					
1	**Workload reduction:** "LLM-generated triage notes of this quality could meaningfully reduce my current	4.00	2	5	0.88
2	**Comfort relying:** "I would be comfortable relying on outputs like these when making triage decisions."	3.29	1	5	1.07

Autonomy Preference

Preference	Count	%
Advisory only: analyst makes all final decisions	7	50%
Human-in-the-loop: analyst reviews before final triage	6	43%
Fully autonomous: no analyst review required	1	7%
I would not be comfortable using this system	0	0%

Appendix 5 – Survey Qualitative Responses

For the Neverhack Estonia survey, three open-ended question types were included: a per-alert omission question, an operational usefulness question and a production readiness question. The responses are listed below exactly as submitted, with only capitalisation and punctuation corrected where clearly missing.

Most Useful Aspects of the Investigation Notes

Open question:

Briefly, what aspects of the investigation notes did you find most useful for operational decision-making?

Responses:

- Easy to copy-paste formatted evidence and description. I would still rewrite the descriptions to make them sound less robotic.
- Clear correlation between technical indicators and business context. The structured summaries. How the notes linked evidences.
- Information such as source IP, username, abuse score and etc.
- Additional contexts from previous cases.
- Automated lookups, automated root cause writing, historical incident reference.
- Initial quick and dirty analysis.
- Integrations with community TI platforms, the initial root cause being ready from the very beginning.
- Summary.

- This helps reduce the time spent writing the investigation notes, and increase the focus and the time for the analysis itself.
- Multiple reasons behind why something is and the potential risks as a result.

Improvements Needed for Production Suitability

Open question:

Briefly, what improvements would be necessary before you would consider a system like this suitable for production use?

Responses:

- Evidence (double blind study and all that) that the system's error rate is significantly lower than human error rate.
- I would take a closer look for the multi agent system where multiple layers of checks are done to make sure the right decision is taken by the first system. Other then that, the system is beautiful and I would love to test it. Great Job!
- Some notes were missing a few steps that could improve their overall quality.
- More testing, compatibility with different tools.
- Lack of enrichment/guesswork, simplified verifiability of evidence (such as links to event and field or user profile etc), ability to report exact details that are wrong, clearly marked fields that are unable to be filled automatically, strong reputation of being correct.
- Malware investigation capabilities need improvement; performance on less technical tasks is satisfactory.
- Human in the loop.
- Avoid using unnecessary calls, like IP reputation call in MS Defender alert. Avoid providing inexperienced L1 analysts with root causes which may underestimate the criticality of incidents. Again the case with MS Defender alert is a perfect example. The facts that the user is a Java Developer and that the file is not marked

as malicious by VirusTotal, does not mean that it is not malicious. It may confuse L1 analysts and lead to the incident being investigated poorly, being escalated later than it should have been or even not being escalated at all.

- Alert source based classification would improve accuracy.
- More model training on real work data, and a review of the LLM output. Everything has to be reviewed to the last details before going full production. Besides that, I would really appreciate a tool like this in production, but not fully autonomous.
- Even though I didn't exactly notice hallucinations in the current investigation notes, I would say reduction in AI hallucinations and absolute certainty that it is reliable by rigorous testing.

Per-Alert Omission Comments

Per-alert open question:

Did this investigation note omit any critical information you would expect a Tier 1 analyst to document? If yes, please describe what was missing.

Alert 1: Data Exfiltration Anomaly (HP) - 3 out of 14 reported omissions

- Authentication details (MFA status, login anomalies, impossible travel) were missing. Device information (managed/unmanaged, risky device signals) were missing. Any possible exfiltrations.
- Paths, file example, URL, IP identification, Device info (ID, managed yes/no).
- Files were downloaded from Executive_Board_Confidential in SharePoint, but that info is not present in the investigation notes.

Alert 2: Suspicious PowerShell Script (LP) - 1 out of 14 reported omissions

- (Participant indicated yes but did not provide details.)

Alert 3: Suspicious Login (HP) - 5 out of 14 reported omissions

- User job role is not mentioned. Source IP geolocation should be in the evidence section as well.
- User verification needs to be done. Application correlation to check whether the other applications were accessed from the same location. Suggesting response actions if it is confirmed to be malicious.
- Whether any login failures observed prior.
- The root cause in fact didn't miss any important details, it just described them in a hardly understandable manner. In the "Description" section everything is written properly, but in the part with key value pairs it would be better to rename keys. I think that "MFA Status" must be changed to "MFA Enabled", because if I was a client I would have problems understanding what would "Status: false" mean. Also "IP Reputation Score" field should be described better. It is not clear if 100 score is good or bad. It is written later in description, but it would be better if it is easily understandable only from key value pair naming.
- (Participant indicated yes but did not provide details.)

Alert 4: EDR Malware (LP) - 7 out of 14 reported omissions

- An explanation of what a lock file is in the context of Gradle.
- In my personal opinion, that could be wrong, the file information was missing. Why it was detected by Defender, what is the source of the file and why is it there. It could be cache.
- File origin. Sign additional files or persistence artifacts on host.
- Code snippet.
- The note should mention that the detected file is a .lock file in a Gradle cache path, which is an unusual fit for genuine MacKeeper software and increases the likelihood of a false positive or mislabeled PUA detection. It

would also help to document whether the file was quarantined or merely blocked, whether the file could be restored/re-scanned, whether other detections occurred on the same host, and whether Defender showed any additional artifact details such as signer, origin, or download source.

- I would prefer to take a closer look at the file itself and investigate the origin of the file on the host. The sole fact of hash being unrecognized by community TI platforms does not mean that file is not malicious. I would submit file to online tools for static and dynamic analysis. Additionally, I would try to look into it on my own by extracting it via MS Defender functionality and running "file" and "strings" commands on Linux sandbox. The file may be masqueraded as .lock file and can be a totally another file type with other aims. Also, if in doubt or if lacking certain knowledge or skills, I would expect an L1 analyst to ask advice and assistance from an L2 analyst. In order to attempt to confirm an assumption about false positive detection, I would also attempt to search for similar problem encountered by other people on the internet. Usually, on forums there are conversations where people discover and discuss such false positives. I would definitely try to find at least one same case with .lock file being falsely detected by MS Defender. Additionally, in case of source IP being an internal private address, it doesn't make any sense to try to collect reputation score for the IP. The system should be fine-tuned to avoid such calls.
- (Participant indicated yes but did not provide details.)

Alert 5: Threat Intelligence: Leaked Documents (HP) - 4 out of 14 reported omissions

- Not an omission exactly, but "Source IP" makes no sense in this context.
- CTI alerts are usually LP unless there is critical breach.
- There could be a bypass for the old domain to contain a phishing. There could be more tools added to capture more complex phishing attempts, for example using domain screenshot analysis.

- The issue is again in formatting of root cause. Potentially malicious URLs must always be defanged (Cyberchef tool could help with this). Also, I would investigate the webpage with a sandbox. I would like to click different buttons, see the final page after completing the form, etc.