

# **A Prototype Data Agent for Threat Modelling in Security Operation Centers**

**Owino Moses Odhiambo**

**171796**

**Submitted in Partial Fulfilment of the Requirements for the Degree of Master of Science  
in Information Technology at Strathmore University**



**School of Computing and Engineering Sciences**

**Strathmore University**

**Nairobi, Kenya**

**June, 2025**

## Declaration and Approval

I declare that this work has not been previously submitted and approved for the award of a degree by this or any other University. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made in the thesis itself.

© No part of this thesis may be reproduced without the permission of the author and Strathmore University.

Candidate Name: Owino Moses Odhiambo

Signature:  \_\_\_\_\_

Date: 20 May, 2025

### Approval

The thesis of Owino Moses Odhiambo was reviewed and approved for examination by the following:

Dr. Nelson Ochieng

Lecturer, School of Computing and Engineering Sciences,

Strathmore University

Dr. Julius Butime,

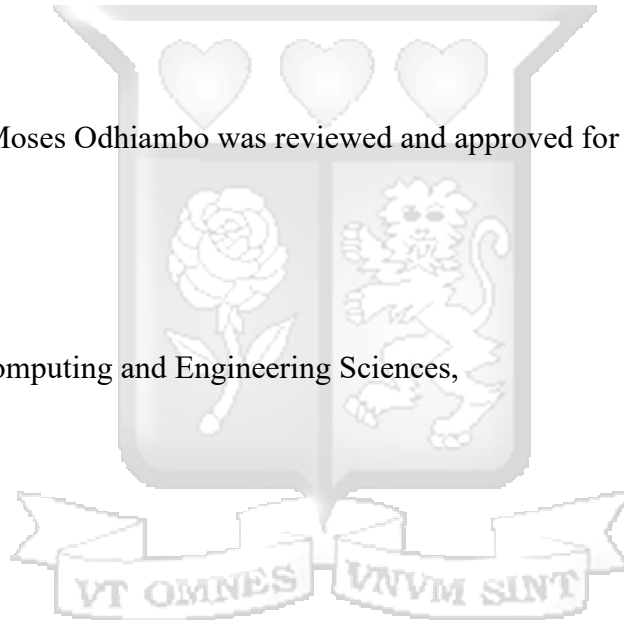
Dean, School of Computing and Engineering Sciences,

Strathmore University

Prof. Bernard Shibwabo,

Director of Graduate Studies,

Strathmore University



## Abstract

The rapidly evolving Cybersecurity threat landscape demands innovative threat modelling tools that can offer customized threat investigation in the complex and dynamic Security Operation Centers. This thesis outlines the development of a Data Agent designed to support Human Security Analysts in Security Operations Centers (SOCs). The system leverages advancements in generative artificial intelligence (GenAI) and in particular, Large Language Models (LLMs) and Retrieval Augmented Generation (RAG) to create an interactive Data Agent to augment human Security Analysts in investigating high risk threats. The main objective of this research is the design, development, and testing of a Large Language Model-Powered GenAI Agent that acts as a Data Agent, guiding human Security Analysts through the complexities of cybersecurity threat investigation. The intelligent Agent uses a conversational interface to provide explanations, answer questions and offer examples, thus engaging human Security Analysts in the Security Operation Centers. Methodologically, this research adopted a design science approach, involving the iterative development of the intelligent Agent system followed by rigorous testing in virtualized controlled environment. The system's effectiveness was evaluated based on its impact on threat investigation, accuracy levels, and user satisfaction.

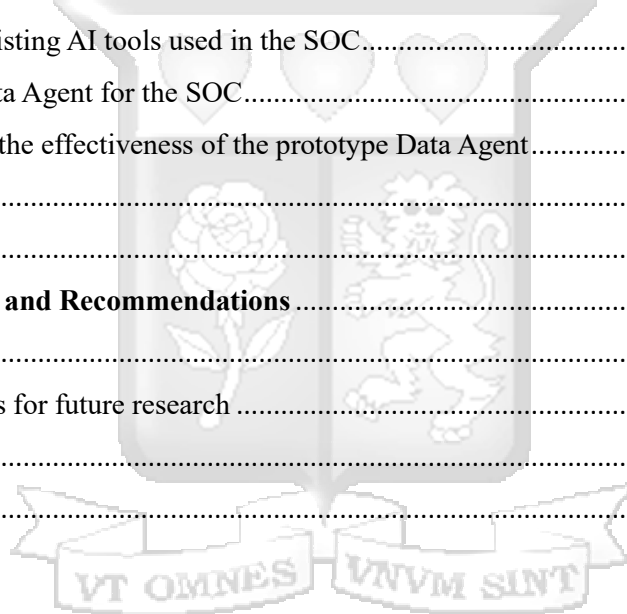
**Keywords:** artificial intelligence, cybersecurity, Data Agent, large language model, pre-trained transformer, false positives, retrieval augmented generation, threat investigation, security operations center

# Table of Contents

<b>Declaration</b> .....	ii
<b>Abstract</b> .....	iii
<b>Table of Contents</b> .....	iv
<b>List of Tables</b> .....	vii
<b>List of Figures</b> .....	viii
<b>Abbreviations/Acronyms</b> .....	ix
<b>Definition of Terms</b> .....	x
<b>Dedication</b> .....	xii
<b>Acknowledgements</b> .....	xiii
<b>Chapter 1: Introduction</b> .....	1
1.1 Background and State of the Art.....	1
1.2 Problem Statement.....	2
1.3 Aim of the Study .....	3
1.4 Research objectives.....	3
1.5 Research Questions .....	3
1.6 Scope and Limitation .....	3
<b>Chapter 2: Literature Review</b> .....	5
2.1 Introduction.....	5
2.2 Empirical Literature.....	5
2.3 Theoretical Literature.....	8
2.3.1 Large Language Models.....	8
2.3.2 Data Agents.....	12
2.4 Algorithms.....	13
2.4.1 Generative Pre-trained Transformers .....	13
2.4.2 Bidirectional Encoder Representations from Transformers .....	15
2.4.3 Reinforcement Learning from Human Feedback.....	16
2.5 Architectures and Designs.....	16
2.6 Existing Solutions .....	20
2.7 Research Gap .....	21
2.8 Conceptual Framework.....	21
<b>Chapter 3: Research Methodology</b> .....	23
3.1 Introduction.....	23

3.2 Research Design and Philosophy .....	23
3.3 Target Population / Sampling .....	23
3.4 Design Science Research (DSR) .....	23
3.4.1 The Rationale for Choosing Design Science Research .....	24
3.4.2 Components of Design Science Research .....	24
3.5 Data Collection .....	25
3.6 Data Analysis .....	25
3.7 System Development Methodology .....	25
3.7.1 Agile Development Methodology .....	25
3.7.2 Design-Based Research (DBR) .....	26
3.8 Data Agent System Development .....	26
3.8.1 Requirements Analysis .....	27
3.8.2 System Design .....	27
3.8.3 Tools Selection .....	27
3.8.4 Development .....	27
3.8.5 Testing .....	27
3.9 Ethical Considerations .....	27
3.10 Utilization of the Research Results .....	28
3.11 Dissemination of the Research Results .....	28
<b>Chapter 4: System Design and Architecture</b> .....	<b>29</b>
4.1 Introduction .....	29
4.2 System Requirements .....	29
4.2.1 Functional requirements .....	29
4.2.2 Non-functional Requirements .....	29
4.3 System Architecture .....	30
4.3.1 Inputs .....	31
4.3.2 Key processes .....	31
4.4 System Design Tools .....	31
4.4.1 Flow Chart Diagram .....	31
4.4.2 Use Case Diagram .....	32
4.5 Web Chat Interface .....	33
4.6 Storage .....	33
<b>Chapter 5: System Implementation and Testing</b> .....	<b>34</b>
5.1 Introduction .....	34
5.2 System Implementation .....	34
5.2.1 Hardware Environment .....	34

5.2.2 Software Environment .....	35
5.2.3 Virtual Environment.....	35
5.3 System components and modules .....	35
5.3.1 Deploying the CALDERA Server.....	36
5.3.2 Deploying the Wazuh Server.....	39
5.3.3 The Data Agent implementation .....	42
5.4 System Testing .....	42
5.4.1 Functional Testing.....	47
5.4.2 Non-functional testing.....	48
5.5 System Validation .....	48
<b>Chapter 6: Discussion .....</b>	<b>50</b>
6.1 Introduction.....	50
6.1.1 Review of existing AI tools used in the SOC.....	50
6.1.2 Prototype Data Agent for the SOC.....	50
6.1.3 Validation of the effectiveness of the prototype Data Agent.....	51
6.2 Conclusion .....	54
6.3 Study Limitations.....	54
<b>Chapter 7: Conclusion and Recommendations .....</b>	<b>56</b>
7.1 Conclusions.....	56
7.2 Recommendations for future research .....	56
<b>References .....</b>	<b>58</b>
<b>Appendices.....</b>	<b>65</b>



## List of Tables

Table 2.1: Existing Solutions .....	20
Table 5.1: Virtual Machines Specifications.....	35
Table 5.2: Admin Login and Logout of the Servers.....	47
Table 5.3: Deploying Agents on the Servers.....	47
Table 5.4: Adversary Attack Emulation .....	48
Table 5.5: Generation of Wazuh Alerts .....	48
Table 5.6: Extraction and Processing of Alerts .....	48
Table 5.7: Perform Alert Investigation by Querying the Data Agent.....	48



## List of Figures

Figure 2.1: Generic Components of a RAG Agent .....	18
Figure 2.2: Non-Autonomous RAG System .....	19
Figure 2.3: Autonomous RAG System .....	20
Figure 2.4: Conceptual Framework .....	22
Figure 4.1: Alert Triage Architecture .....	30
Figure 4.2: Data Agent Architecture .....	32
Figure 4.3: Process Flow Diagram.....	33
Figure 5.1: Starting the Caldera Server.....	36
Figure 5.2: Caldera Admin Login Page .....	37
Figure 5.3: Caldera Dashboard without Agents .....	37
Figure 5.4: Deploying a Linux Agent .....	38
Figure 5.5: Starting the Linux Agent .....	38
Figure 5.6: Caldera Dashboard Showing the Deployed Agent .....	39
Figure 5.7: Logging into the Wazuh Server .....	39
Figure 5.8: Wazuh SIEM Admin Login Page .....	40
Figure 5.9: Main Wazuh Dashboard Showing Deployed Agents.....	40
Figure 5.10: Wazuh Endpoints Dashboard.....	41
Figure 5.11: Wazuh Alert Page.....	41
Figure 5.12: Code Snippet for Alerts Extraction and Processing Module .....	42
Figure 5.13: Initiating a Brute-force attack.....	43
Figure 5.14: Wazuh Showing Alerts Triggered by the Emulated Brute-force Attack .....	43
Figure 5.15: Generic login attempts query and response .....	45
Figure 5.16: Specific login attempts query and response .....	45
Figure 5.17: Generic Windows registry query and response .....	46
Figure 5.18: Specific Windows registry query and response .....	47
Figure 6.1: Overall Alerts Classification and Endpoints.....	51
Figure 6.2: Labelled Windows Alerts .....	52
Figure 6.3: Labelled Linux Alerts.....	53

## Abbreviations/Acronyms

<b>AI</b>	Artificial Intelligence
<b>API</b>	Application Programming Interface
<b>BERT</b>	Bidirectional Encoder Representations from Transformers
<b>CI/CD</b>	Continuous integration and continuous deployment
<b>CNN</b>	Convolutional Neural Networks
<b>CoT</b>	Chain of Thought
<b>DBR</b>	Design Based Research
<b>DSR</b>	Design Science Research
<b>GenAI</b>	Generative Artificial Intelligence
<b>GPT</b>	Generative Pre-trained Transformer
<b>IoC</b>	Indicator of Compromise
<b>LLM</b>	Large Language Model
<b>LSTM</b>	Long Short-Term Memory
<b>MTTI</b>	Mean-Time-To-Investigate
<b>NLG</b>	Natural Language Generation
<b>NLP</b>	Natural Language Processing
<b>NLU</b>	Natural Language Understanding
<b>RAG</b>	Retrieval Augmented Generation
<b>RLHF</b>	Reinforcement Learning with Human Feedback
<b>RNN</b>	Recurrent Neural Networks
<b>SOC</b>	Security Operation Center
<b>UAT</b>	User Acceptance Testing
<b>UCD</b>	User Centered Design
<b>UI</b>	User Interface
<b>UML</b>	Unified Modeling Language
<b>VM</b>	Virtual Machine

## Definition of Terms

<b>Alert Fatigue</b>	A phenomenon that occurs when cybersecurity professionals are overwhelmed with such a high volume of security alerts that it leads to a diminished ability to react effectively to and investigate real threats (proofpoint, 2024).
<b>Artificial Intelligence</b>	Artificial intelligence (AI) is a set of technologies that enable computers to perform a variety of advanced functions, including the ability to see, understand and translate spoken and written language, analyze data, and make recommendations (Google Cloud, 2024).
<b>Attention mechanism</b>	This is a machine learning technique that directs deep learning models to prioritize (or attend to) the most relevant parts of input data (Bergmann and Stryker, 2024).
<b>Data Agent</b>	An Artificial Intelligence system consisting of planning capabilities, memory, and tools to perform tasks requested by a user (Varshney, 2024).
<b>Generative Artificial Intelligence</b>	This is artificial intelligence (AI) that can create original content such as text, images, video, audio or software code in response to a user's prompt or request (Stryker and Scapicchio, 2024).
<b>Hallucination</b>	Within the context of GPT, refers to the propensity of the model to fabricate information or generate text that is not grounded in factual reality (Alkaissi and McFarlane, 2023).
<b>Indicator of Compromise</b>	A piece of digital forensics that suggests that an endpoint or network may have been breached. (Baker, 2025).
<b>Large Language Model</b>	A category of foundation models trained on immense amounts of data making them capable of understanding

and generating natural language and other types of content to perform a wide range of tasks (IBM, 2023).

**Mean-Time-To-Investigate** This is the average time from the time a cyber threat is detected until the SOC team initiates investigation (Splunk, 2025).

**Retrieval-Augmented Generation** This is the process of optimizing the output of a large language model, so it references an authoritative knowledge base outside of its training data sources before generating a response (AWS, 2025).

**Security Operations Center** A facility where teams of security professionals, supported by advanced technologies and processes, work together to monitor, detect, and respond to cybersecurity incidents (Tariq et al. 2025).

**Self-attention** Type of attention mechanism used in machine learning models to weigh the importance of tokens or words in an input sequence to better understand the relations between them (Winland, 2025).

**Self-training** A technique where a model trains itself on labeled data and then uses its own predictions on unlabeled data to continue the learning process (Yadav, 2024).

**Sycophantic behaviour** This is where models may exhibit an inclination to produce content that excessively flatters or conforms to the preferences of the training data, potentially compromising the integrity and impartiality of generated text (Sufi, 2024).

**Threat Modelling** A proactive process of identifying, classifying and describing threats, that provide visibility of an attacker or a campaign of attacks (Tatam et al., 2021).

## Dedication

I dedicate this research project to the Strathmore University fraternity, my friends, family, and classmates whose help has been essential in creating a climate that is conducive to learning, innovation, and development.



## Acknowledgements

I am very grateful to the Almighty God for the grace He has bestowed upon me during the time I have spent in researching and composing this thesis.

I would like to thank Prof. Ismail Ateya Lukandu for his knowledge and advice in Research Methodology and Presentation skills which helped me to develop this thesis.

Further, I wish to thank Dr. Nelson Ochieng Odunga for his knowledge and guidance in conceptualizing and prototyping the solution.



## Chapter 1: Introduction

### 1.1 Background and State of the Art

The growing complexity and volume of cybersecurity threats have placed immense pressure on the security operation centers (SOC) to effectively investigate and respond to security incidents (Gartner, 2020). Traditional security approaches, which rely heavily on human security analysts, have struggled to keep pace with the evolving threat landscape, leading to the exploration of artificial intelligence (AI) and automation to enhance the capabilities of SOCs. One of the key challenges faced by SOCs is the shortage of skilled security personnel and the high turnover rate within the industry (Ponemon Institute, 2019). Security analysts are often overwhelmed by the sheer amount of security alerts generated by various tools and systems, making it increasingly difficult to properly investigate incidents, and make informed decisions in a timely manner (Lucic et al., 2021).

The emergence of large language models (LLMs), such as OpenAI's GPT series and Meta's Llama series, has revolutionized the field of generative artificial intelligence (GenAI) and opened up new possibilities for enhancing cybersecurity operations (Brown et al., 2020; Devlin et al., 2019). These powerful AI models can be fine-tuned and integrated into Data Agents, or "security bots," to automate repetitive tasks, augment human analysts, and provide real-time, contextual threat investigation and response capabilities. LLM-powered Data Agents can leverage their advanced natural language understanding (NLU) and text generation capabilities to analyse vast amounts of security data, including incident reports, threat intelligence, and unstructured security logs (Biernacki, 2019). These virtual analysts can identify patterns, extract relevant information, generate summaries, recommendations, and response strategies, thereby assisting human analysts in investigating security incidents (Lucic et al., 2021).

Moreover, LLM-powered Data Agents can engage in interactive conversations with human analysts, allowing for seamless collaboration and knowledge sharing. They can understand and respond to security-related queries, provide explanations for their actions, and even learn from the feedback and guidance provided by human experts (Sommer & Brown, 2011). However, the successful implementation of an LLM-powered Data Agent requires addressing several challenges, such as the integration of diverse security data sources, the development of robust and trustworthy AI models, and the effective collaboration between human analysts and AI systems (Lucic et al., 2021). Additionally, ensuring the transparency, interpretability, and ethical deployment of such AI-driven solutions is important in the cybersecurity domain. The

application of LLMs in cybersecurity operations has seen a steady rise over the past few years, driven by advancements in generative artificial intelligence, retrieval augmented generation and the growing need for automated, intelligent security solutions.

One of the early pioneers in this field was the research team at OpenAI, who in 2019 released GPT-3, a groundbreaking LLM that demonstrated remarkable language understanding and generation capabilities (Brown et al., 2020). Cybersecurity researchers quickly recognized the potential of GPT-3 and similar LLMs to enhance various aspects of security operations, including threat investigation and response automation. In 2020, a team of researchers from the University of Maryland and the National Security Agency (NSA) published a study on the use of BERT, another prominent LLM, for automated vulnerability detection and patch recommendation (Devlin et al., 2019; Shen et al., 2020). Their work showcased how BERT's deep understanding of natural language could be leveraged to analyse security advisories, technical documentation, and vulnerability reports, and generate relevant mitigation strategies.

As the adoption of LLM-powered security solutions gained momentum, several cybersecurity startups and technology companies began to develop and deploy Data Agents, or "security bots," to assist human teams in the SOC (Gartner, 2021). These virtual analysts utilized LLMs to engage in natural language interactions, understand security contexts, and provide real-time threat detection, analysis, investigation and response.

For example, in 2023, the cybersecurity firm CrowdStrike introduced Charlotte AI; an LLM-powered Data Agent capable of autonomously investigating security incidents, generating incident reports, and suggesting remediation actions (CrowdStrike, 2023). Charlotte AI was designed to augment human analysts, allowing them to focus on more complex and strategic security tasks.

## **1.2 Problem Statement**

Security Analysts are overwhelmed with such a high volume of security alerts that it leads to a diminished ability to prioritize and investigate real threats (Fortinet, 2024). A Data Agent can assist human analysts in effectively investigating and responding to security alerts, thereby reducing the burden of alert fatigue (Lucic et al., 2021; Biernacki, 2019).

This Data Agent uses retrieval augmented generation (RAG) to solve the limitations of existing LLM-powered agents such as hallucination and sycophantic behaviour. Hallucination, within the context of LLMs, refers to the tendency of the model to fabricate information or generate text that is not grounded in factual reality (Alkaissi & McFarlane, 2023). Sycophantic

behaviour on the other hand, is where models may exhibit an inclination to produce content that excessively flatters or conforms to the preferences of the training data, potentially compromising the integrity and impartiality of generated text (Sufi, 2024). RAG enables the connection of the LLM-powered agent to live systems and tools such as the SIEM to allow real-time data access and retrieval. Using RAG is an effective strategy for ensuring LLMs agent responses are up-to-date and not hallucinated (Bagade, 2024).

### **1.3 Aim of the Study**

Develop an LLM-Powered Data Agent to augment human Security Analysts' threat investigation and response in the SOC.

### **1.4 Research objectives**

- i. Identify the challenges associated with cyber threat investigation.
- ii. Review existing technologies and tools used in threat investigation.
- iii. Design an LLM-Powered Data Agent.
- iv. Develop the LLM-Powered Data Agent.
- v. Test the LLM-Powered Data Agent.

### **1.5 Research Questions**

- i. What are the challenges associated with threat modelling?
- ii. What are the existing technologies and tools used in threat modelling?
- iii. How can an LLM-Powered Data Agent be designed?
- iv. How can the LLM-Powered Data Agent be developed?
- v. How can the LLM-Powered Data Agent be tested?

### **1.6 Scope and Limitation**

Alert fatigue is a significant challenge faced by SOC Analysts due to the overwhelming volume of security alerts and notifications generated by various monitoring tools and systems. Alert fatigue occurs when security analysts become overwhelmed by the sheer number of alerts, leading to decreased vigilance, increased likelihood of missed threats, and slower incident response times (Fortinet, 2024). This phenomenon is particularly concerning as it can result in critical security events being overlooked or inadequately addressed, ultimately compromising organizational security.

This thesis focuses on the design and development of an LLM-powered Data Agent designed to enhance the capabilities of Security Operation Centres (SOCs). The Data Agent aims to address key challenges faced by SOCs, including alert fatigue, the complexity of threat

investigation and the efficiency of incident response. The scope of this research encompasses three core areas: The design and development of the Data Agent, its integration with an open-source security information and event management (SIEM) solution, and the evaluation of its performance in real-world scenarios. However, this thesis will not focus on the development of new security detection algorithms and complete overhaul of SOC practices and technologies.



## Chapter 2: Literature Review

### 2.1 Introduction

In an era where cyber threats are increasingly sophisticated and persistent, the need for enhanced security measures within the SOC has never been more critical. The integration of advanced technologies, particularly those powered by Artificial Intelligence (AI), is emerging as a vital strategy to bolster cybersecurity defenses. Among these technologies, LLMs offer promising capabilities for augmenting cyber threat investigation and incident response in the SOC (Brown et al., 2020).

Existing literature highlights a range of applications where LLMs and similar AI tools have been employed to enhance threat investigation, streamline incident response, and improve the overall efficiency of security operations (Joulin et al., 2017; Moustafa et al., 2019). However, despite the notable advancements in AI-driven cybersecurity solutions, there remains a significant gap in the research concerning the design and development of LLM-powered Data Agents specifically tailored for cyber threat investigation in the SOC.

This literature review aims to explore the current state of research on LLMs in cybersecurity, examining their capabilities, limitations, and potential applications within the SOC. Additionally, it will analyze existing frameworks and methodologies employed in the design of Data Agents, identifying best practices and areas for improvement. By synthesizing insights from various studies, this review will provide a foundational understanding that informs the design and development of an LLM-powered Data Agent tailored for use in the SOC.

### 2.2 Empirical Literature

In the 2024 paper titled "An Innovative GPT-Based Open-Source Intelligence Using Historical Cyber Incident Reports," Sufi presents a novel approach to leveraging Generative Pre-trained Transformers (GPT) for enhancing open-source intelligence (OSINT) in cybersecurity. The study focused on analyzing historical cyber incident reports to extract valuable insights that can inform current security practices and threat assessments. By employing a GPT-based model, Sufi (2024) aims to automate the extraction and synthesis of information from these reports, thereby facilitating a more efficient analysis process. Sufi's research underscores the potential of using GPT to process vast amounts of unstructured data contained in historical incident reports. The model is trained to identify key patterns, trends, and recurring vulnerabilities, which can serve as crucial indicators for predicting future cyber threats. The author conducted a series of experiments to validate the effectiveness of the GPT-based

approach, comparing it with traditional OSINT methods. The findings indicate that the GPT model significantly enhances the accuracy of threat identification and improves the speed of data processing.

One of the key contributions of the study is the development of a framework that integrates GPT's capabilities into existing OSINT workflows. Sufi (2024) emphasizes that this integration allows cybersecurity professionals to stay ahead of emerging threats by providing timely and relevant insights derived from past incidents. The results also highlight the model's ability to generate actionable intelligence, which can inform incident response strategies and enhance overall security posture.

The increasing volume and complexity of cyber threats have highlighted the need for more effective and efficient threat intelligence analysis. Traditional manual analysis of threat intelligence is time-consuming and prone to errors, making it challenging for security teams to respond quickly and effectively to emerging threats (Siracusano et al., 2023). This paper proposes an automated approach to threat intelligence analysis using machine learning and natural language processing techniques. The authors develop a system that can automatically collect, analyze, and prioritize threat intelligence from various sources, including open-source intelligence, social media, and dark web forums. The system uses machine learning algorithms to identify patterns and relationships between different threat indicators, such as IP addresses, domains, and malware samples (Siracusano et al., 2023). The authors demonstrate the effectiveness of the system in identifying and prioritizing high-risk threats, enabling security teams to respond quickly and effectively to emerging threats.

One of the key findings of this paper by Siracusano et al. (2023), is that automated threat intelligence analysis can significantly improve the speed and accuracy of threat detection and response. The authors find that the system can reduce the time required for threat analysis from hours to minutes, enabling security teams to respond more quickly to emerging threats. The paper also highlights the importance of integrating automated threat intelligence analysis with existing security systems, such as security information and event management (SIEM) systems and incident response platforms. The authors also identify several challenges and limitations associated with automated threat intelligence analysis, including the need for high-quality training data, the risk of false positives and negatives, and the importance of human oversight and validation (Siracusano et al.).

In the paper titled "SecureFalcon: The Next Cyber Reasoning System for Cyber Security," Ferrag et al. (2023) introduce an innovative framework aimed at enhancing cybersecurity through advanced reasoning capabilities. The authors argue that traditional cybersecurity measures often fall short in effectively predicting and mitigating sophisticated cyber threats due to the rapidly evolving nature of these attacks. SecureFalcon is proposed as a comprehensive cyber reasoning system designed to address these challenges by integrating machine learning, knowledge representation, and automated reasoning. The framework of SecureFalcon employs a combination of artificial intelligence techniques to analyze and interpret vast amounts of security data in real time. The authors highlight that by leveraging both structured and unstructured data, the system can identify vulnerabilities, detect anomalies, and predict potential attacks more effectively than conventional methods. The paper presents experimental results demonstrating SecureFalcon's ability to provide actionable insights and recommendations for threat mitigation, showcasing its superiority in threat detection and response time.

One of the key findings of the study is SecureFalcon's capability to enhance situational awareness for cybersecurity teams. By synthesizing information from various sources, the system can generate a holistic view of the threat landscape, helping organizations make informed decisions in real time. Additionally, the authors emphasize the importance of continuous learning in SecureFalcon, allowing the system to adapt to new threats and evolving attack patterns, thus ensuring long-term effectiveness.

Manocchio et al. (2024) in their paper titled "FlowTransformer: A Transformer Framework for Flow-Based Network Intrusion Detection Systems," introduce a novel framework designed to enhance the capabilities of intrusion detection systems (IDS) by leveraging the power of transformer architectures. The authors note that traditional flow-based IDS often struggle with high false positive rates and limited adaptability to evolving attack patterns. FlowTransformer aims to address these shortcomings by employing a transformer model to analyze network flow data more effectively. The study begins by detailing the architecture of FlowTransformer, which is specifically tailored to process flow data generated by network traffic. By using self-attention mechanisms, the model can capture long-range dependencies and relationships within the data, allowing it to identify complex attack patterns that may not be discernible with conventional methods. The authors conducted extensive experiments comparing FlowTransformer with existing IDS models, demonstrating its superior performance in terms of accuracy, detection speed, and robustness against various types of network attacks.

One of the key findings of the research is that FlowTransformer significantly reduces false positive rates while maintaining high detection accuracy. The model's ability to learn from contextual information within flow data enables it to better differentiate between legitimate traffic and potential threats. Additionally, the authors highlight the framework's scalability, indicating that it can handle large volumes of network traffic without compromising performance, making it suitable for deployment in real-world scenarios.

In the paper titled "Flow-Based Network Intrusion Detection Based on BERT Masked Language Model," Nguyen and Watabe (2022) explore the application of BERT, a powerful masked language model, for enhancing flow-based network intrusion detection systems (NIDS). The authors argue that traditional NIDS often face challenges in accurately identifying complex attack patterns due to the high-dimensional nature of flow data. By utilizing BERT's contextual understanding capabilities, the study aims to improve detection accuracy and reduce false positives. The research introduces a novel approach where network flow data is transformed into a format suitable for BERT processing. The authors explain how they leverage the masked language modeling technique, allowing the model to predict missing parts of the data and thus learn underlying patterns associated with both normal and malicious activities. The experimental results demonstrate that this BERT-based approach outperforms conventional flow-based detection methods in terms of accuracy and robustness against various attack scenarios.

One of the key findings of the study is that the BERT model significantly enhances the system's ability to capture complex temporal dependencies and contextual information within flow data. This capability allows the model to better distinguish between benign and malicious traffic, leading to a marked reduction in false positive rates. Furthermore, Nguyen and Watabe (2022) emphasize the scalability of their approach, making it suitable for deployment in real-time network environments with large volumes of traffic.

## **2.3 Theoretical Literature**

### **2.3.1 Large Language Models**

The Google Deep Mind team in the groundbreaking paper "Attention is All You Need" (Vaswani et al., 2017), presented the transformer architecture, which relies solely on self-attention mechanisms instead of traditional recurrent or convolutional neural networks for handling sequential data. The main findings of the paper are centered around the transformer architecture and the use of self-attention mechanisms to process input sequences. They proposed a novel multi-head self-attention mechanism that allowed the model to capture

various contexts and dependencies between elements in an input sequence (Vaswani et al., 2017). They demonstrated that using self-attention mechanisms could effectively replace recurrent neural networks (RNNs) and convolutional networks in various NLP tasks, such as machine translation. The transformer model consisted of encoder-decoder stacks, each containing multiple identical layers of multi-head self-attention and feed-forward neural networks (Vaswani et al., 2017).

The primary advantage of the transformer architecture is its ability to capture long-range dependencies between input elements more effectively than RNNs and the improved parallelization during training, enabling faster and more efficient processing of sequential data (Vaswani et al., 2017). The authors showed that the transformer-based model outperformed traditional RNN and convolutional networks in machine translation tasks, proving the efficacy of the self-attention mechanism. The transformer architecture, introduced by Vaswani et al. (2017), has become a fundamental component in large-scale language models such as BERT, RoBERTa, and GPT, which have significantly impacted NLP and cybersecurity applications, such as threat intelligence, anomaly detection, and incident response (Devlin et al., 2019; Liu et al., 2019; Brown et al., 2020). The transformer architecture's introduction paved the way for subsequent large-scale LLMs and their successful deployment in various cybersecurity tasks.

In the paper "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" (Devlin et al., 2019) introduced BERT, a groundbreaking language model using a deep bidirectional transformer architecture for understanding language and significantly advancing the state-of-the-art in various NLP tasks. The authors emphasized the importance of pre-training contextualized representations of language and proposed a novel method for training bidirectional models.

The main findings by Devlin et al. (2019) revolve around the bidirectional training of transformer-based models and the pre-trained language representations' effectiveness. BERT utilized a masked language modeling objective, wherein certain input tokens were randomly masked, and the model was tasked with predicting the masked tokens based on their surrounding context (Devlin et al., 2019). Additionally, the authors introduced the next sentence prediction task, enabling BERT to understand the relationships between pairs of sentences (Devlin et al., 2019). The authors demonstrated that BERT outperformed traditional models in various NLP tasks, such as question-answering, named entity recognition, and sentiment analysis, by fine-tuning the pre-trained model on specific task data (Devlin et al.,

2019). The bidirectional contextualized representations learned during pre-training enabled BERT to capture richer semantic and syntactic relationships between words, providing a better understanding of the input data (Devlin et al., 2019).

In the paper "Language Models are Unsupervised Multitask Learners," Radford et al. (2019) introduce GPT-2, a large-scale generative pre-trained transformer model that demonstrates significant advancements in the field of natural language processing (NLP). The authors build upon the original GPT architecture, emphasizing the model's capacity to perform a wide array of language tasks without requiring explicit task-specific training data. This ability is attributed to the model's training on diverse datasets drawn from the internet, enabling it to learn the complexities of human language through unsupervised learning.

A key finding by Radford et al. (2019) is that, GPT-2 can effectively generate coherent and contextually appropriate text across multiple tasks, including text completion, translation, and summarization, even when fine-tuning is not applied. The model's performance is showcased through various qualitative and quantitative evaluations, revealing its capability to produce high-quality text that often mimics human writing styles. The authors highlight that this versatility is a result of the model's architecture, which uses a large number of parameters (1.5 billion), allowing it to capture intricate patterns and nuances in language.

Radford et al. (2019) also discuss the implications of the model's capabilities for the broader NLP community. GPT-2's ability to generalize across tasks without supervised fine-tuning represents a significant shift in how language models can be utilized, suggesting that large unsupervised models can serve as robust starting points for various applications. However, the authors express concerns regarding the potential misuse of such powerful models, particularly in generating misleading information or deepfakes, leading to discussions about the ethical considerations of deploying language models in real-world applications.

In the influential paper "Language Models are Few-Shot Learners," Brown et al. (2020) introduce GPT-3, a large language model that demonstrates remarkable capabilities in few-shot learning, where the model can perform tasks with minimal task-specific examples. The authors detail the architecture of GPT-3, which consists of 175 billion parameters, making it one of the largest language models to date. This scale allows GPT-3 to generate coherent and contextually relevant text across a wide range of tasks, from translation to question-answering, often without the need for extensive fine-tuning.

One of the central findings by Brown et al. (2020) is that GPT-3 excels at few-shot, one-shot, and even zero-shot learning. This means that the model can generalize to new tasks effectively by simply providing it with a few examples or even just task descriptions. The authors demonstrate this capability across a variety of benchmarks, highlighting how GPT-3 achieves state-of-the-art performance on numerous language understanding and generation tasks. This ability to learn from minimal examples is particularly significant, as it reduces the data and time required for model training in specific applications, thereby increasing accessibility for developers and researchers.

The authors also discuss the implications of GPT-3's performance, emphasizing the importance of scale in model training. They illustrate that larger models tend to exhibit better performance across tasks, suggesting a positive correlation between model size and capability. However, this raises concerns about the resource-intensive nature of training such models, prompting discussions about the environmental and economic implications of large-scale AI development.

Brown et al. (2020) provides a comprehensive survey of prompting methods in natural language processing (NLP), which have gained significant attention in recent years due to their ability to adapt pre-trained language models to specific tasks without requiring task-specific labeled data (Liu et al., 2020). The authors define prompting as "the process of designing and using a prompt to elicit a specific response from a pre-trained language model" (Liu et al., 2020, p. 2). They identify three key components of prompting: the pre-trained language model, the prompt, and the prediction task.

The authors categorize prompting methods into three main categories: (1) manual prompting, where humans design prompts based on their understanding of the task and the language model; (2) automatic prompting, where prompts are generated using algorithms or models; and (3) hybrid prompting, which combines manual and automatic prompting (Liu et al., 2020, p. 5). They also identify several key challenges in prompting, including prompt engineering, prompt bias, and the trade-off between prompt simplicity and effectiveness (Liu et al., 2020, p. 7).

The survey covers a range of NLP tasks, including language translation, question answering, text classification, and sentiment analysis, among others (Liu et al., 2020, p. 9). The authors find that prompting methods have achieved state-of-the-art results in many of these tasks, often outperforming traditional fine-tuning approaches (Liu et al., 2020, p. 11). They also identify several key factors that influence the effectiveness of prompting methods, including the choice

of pre-trained language model, the design of the prompt, and the size and quality of the training data (Liu et al., 2020, p. 13).

### **2.3.2 Data Agents**

In their 2021 paper, "AI-Driven Cybersecurity: Leveraging LLMs for Threat Detection," Chung and Jang (2021) explore the transformative potential of Large Language Models (LLMs) in enhancing threat detection within cybersecurity frameworks. The authors argue that traditional methods of threat detection often struggle with the vast amounts of unstructured data generated in modern IT environments, which can include security logs, incident reports, and user-generated content. By integrating LLMs, organizations can leverage advanced natural language processing capabilities to analyze this data more effectively.

Chung and Jang (2021) present a series of case studies demonstrating how LLMs can identify patterns and anomalies that traditional rule-based systems might miss. They highlight that LLMs excel at processing natural language, allowing them to interpret complex threat intelligence reports and correlate them with real-time security data. This capability significantly enhances the accuracy and speed of threat detection, enabling organizations to respond proactively to potential security incidents. The authors also discuss the model's ability to learn from new data, which continuously improves its performance over time.

This paper by Chung and Jang (2021) concludes that the adoption of LLM-powered threat detection systems represents a significant advancement in cybersecurity strategy, offering the potential for reduced response times and improved threat mitigation. However, Chung and Jang (2021) caution that while LLMs provide powerful tools for cybersecurity, organizations must also address challenges such as data privacy, model interpretability, and the risk of algorithmic bias to ensure ethical deployment and effective use.

The increasing volume and complexity of security alerts have overwhelmed Security Operations Centers (SOCs), making it challenging for Tier 1 analysts to efficiently investigate and respond to threats as shown by Oniagbi et al. (2024) in their paper "Evaluation of LLM Agents for the SOC". Large Language Models (LLMs) have shown promise in augmenting human analysts in the investigation process, but their effectiveness in this context has not been thoroughly evaluated (Oniagbi, 2024). This paper presents an evaluation of LLM agents in supporting Tier 1 analysts in the triage process, focusing on their ability to reduce false positives, improve response times, and enhance overall analyst productivity (Oniagbi, 2024).

Oniagbi (2024) designed an experiment to compare the performance of human analysts with and without LLM agent support in triaging security alerts. The LLM agents were trained on a dataset of labeled security alerts and were integrated into the triage process to provide recommendations and insights to the analysts (Oniagbi, 2024). The results showed that the LLM agents significantly reduced the number of false positives, with an average reduction of 35% compared to human analysts working alone (Oniagbi, 2024).

Furthermore, the study found that the LLM agents improved response times by an average of 27%, allowing analysts to focus on higher-priority tasks (Oniagbi, 2024). The authors also observed a significant reduction in analyst fatigue and burnout, as the LLM agents helped to alleviate the cognitive load associated with manual triage (Oniagbi, 2024). Additionally, the LLM agents were found to be effective in identifying unknown threats and anomalies, which are often challenging for human analysts to detect.

The key findings of the paper by Oniagbi (2024) include the potential of LLM agents to augment human analysts in the triage process, improving the efficiency and effectiveness of SOC operations. The authors suggest that LLM agents can be used to support Tier 1 analysts in various ways, including alert prioritization, threat analysis, and incident response (Oniagbi, 2024). However, the study also highlights the need for further research to address the limitations and challenges associated with LLM agent integration, such as data quality, model bias, and explainability.

## **2.4 Algorithms**

The paper "Attention is All You Need" by Vaswani et al. proposes a novel neural network architecture, called the Transformer, which relies entirely on self-attention mechanisms to process input sequences. The authors argue that traditional recurrent and convolutional neural networks (RNNs and CNNs) are not well-suited for sequence-to-sequence tasks, such as machine translation, due to their sequential nature and limited parallelization capabilities.

The Transformer architecture, on the other hand, is designed to handle input sequences of arbitrary length and can be parallelized more easily, making it more efficient and scalable. The key innovation of the Transformer is the self-attention mechanism, which allows the model to attend to different parts of the input sequence simultaneously and weigh their importance.

### **2.4.1 Generative Pre-trained Transformers**

In the paper "Language Models are Unsupervised Multitask Learners," Radford et al. (2019) introduce GPT-2, a large-scale generative pre-trained transformer model that demonstrates

significant advancements in the field of natural language processing (NLP). The authors build upon the original GPT architecture, emphasizing the model's capacity to perform a wide array of language tasks without requiring explicit task-specific training data. This ability is attributed to the model's training on diverse datasets drawn from the internet, enabling it to learn the complexities of human language through unsupervised learning.

A key finding of the paper is that GPT-2 can effectively generate coherent and contextually appropriate text across multiple tasks, including text completion, translation, and summarization, even when fine-tuning is not applied. The model's performance is showcased through various qualitative and quantitative evaluations, revealing its capability to produce high-quality text that often mimics human writing styles. The authors highlight that this versatility is a result of the model's architecture, which uses a large number of parameters (1.5 billion), allowing it to capture intricate patterns and nuances in language.

The paper also discusses the implications of the model's capabilities for the broader NLP community. GPT-2's ability to generalize across tasks without supervised fine-tuning represents a significant shift in how language models can be utilized, suggesting that large unsupervised models can serve as robust starting points for various applications. However, the authors express concerns regarding the potential misuse of such powerful models, particularly in generating misleading information or deepfakes, leading to discussions about the ethical considerations of deploying language models in real-world applications.

In the influential paper "Language Models are Few-Shot Learners," Brown et al. (2020) introduce GPT-3, a large language model that demonstrates remarkable capabilities in few-shot learning, where the model can perform tasks with minimal task-specific examples. The authors detail the architecture of GPT-3, which consists of 175 billion parameters, making it one of the largest language models to date. This scale allows GPT-3 to generate coherent and contextually relevant text across a wide range of tasks, from translation to question-answering, often without the need for extensive fine-tuning.

One of the central findings of the study is that GPT-3 excels at few-shot, one-shot, and even zero-shot learning. This means that the model can generalize to new tasks effectively by simply providing it with a few examples or even just task descriptions. The authors demonstrate this capability across a variety of benchmarks, highlighting how GPT-3 achieves state-of-the-art performance on numerous language understanding and generation tasks. This ability to learn from minimal examples is particularly significant, as it reduces the data and time required for

model training in specific applications, thereby increasing accessibility for developers and researchers.

The authors also discuss the implications of GPT-3's performance, emphasizing the importance of scale in model training. They illustrate that larger models tend to exhibit better performance across tasks, suggesting a positive correlation between model size and capability. However, this raises concerns about the resource-intensive nature of training such models, prompting discussions about the environmental and economic implications of large-scale AI development.

In addition to performance metrics, Brown et al. address ethical considerations associated with GPT-3, such as the potential for generating harmful or misleading content. They highlight the necessity for responsible deployment and oversight, given the model's ability to produce text that can mimic human writing convincingly. Overall, Brown et al. (2020) provide compelling evidence that large language models like GPT-3 can function as few-shot learners, setting a new benchmark for future research in natural language processing and prompting important discussions about the ethical implications of deploying such powerful AI systems.

#### **2.4.2 Bidirectional Encoder Representations from Transformers**

In the paper "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" (Devlin et al., 2019), the authors introduced BERT, a groundbreaking language model using a deep bidirectional transformer architecture for understanding language and significantly advancing the state-of-the-art in various NLP tasks. The authors emphasized the importance of pre-training contextualized representations of language and proposed a novel method for training bidirectional models.

The main findings of the paper revolve around the bidirectional training of transformer-based models and the pre-trained language representations' effectiveness. BERT utilized a masked language modeling objective, wherein certain input tokens were randomly masked, and the model was tasked with predicting the masked tokens based on their surrounding context (Devlin et al., 2019). Additionally, the authors introduced the next sentence prediction task, enabling BERT to understand the relationships between pairs of sentences (Devlin et al., 2019).

The authors demonstrated that BERT outperformed traditional models in various NLP tasks, such as question-answering, named entity recognition, and sentiment analysis, by fine-tuning the pre-trained model on specific task data (Devlin et al., 2019). The bidirectional contextualized representations learned during pre-training enabled BERT to capture richer

semantic and syntactic relationships between words, providing a better understanding of the input data (Devlin et al., 2019).

### **2.4.3 Reinforcement Learning from Human Feedback**

The paper "Learning to Summarize with Human Feedback" by Stiennon et al. (2020) proposes a novel approach to training summarization models using human feedback (Stiennon et al., 2020). The authors argue that traditional summarization models are often trained using automatic metrics, such as ROUGE, which do not accurately capture the nuances of human judgment (Stiennon et al., 2020). Instead, they propose a framework that uses human feedback to train a summarization model, which they call "human-in-the-loop" summarization (Stiennon et al., 2020). The authors collect a dataset of human-annotated summaries, which they use to train a summarization model using reinforcement learning (Stiennon et al., 2020). The model is trained to maximize the reward function, which is based on the human feedback (Stiennon et al., 2020). The authors show that the human-in-the-loop approach leads to significant improvements in summarization quality, as measured by human evaluators (Stiennon et al., 2020).

One of the key findings of the paper is that human feedback is essential for training high-quality summarization models (Stiennon et al., 2020). Another key finding is that the human-in-the-loop approach can be used to train models that are more interpretable and transparent (Stiennon et al., 2020). The authors show that the model's attention weights, which indicate the importance of each input sentence, are more correlated with human judgment when trained using human feedback (Stiennon et al., 2020). This suggests that the human-in-the-loop approach can lead to more explainable and trustworthy summarization models (Stiennon et al., 2020).

## **2.5 Architectures and Designs**

A typical LLM-Powered Data Agent architecture is designed to facilitate a seamless interaction between the agent or swarm of agents and the human security analysts in the SOC. These systems are generally composed of several core components that work together to understand user inputs, determine the most appropriate responses, and deliver those responses in a way that is effective and engaging. The key components of a typical agent architecture include:

### **i. The Agent Core**

Think of it as the brain of the agent system. In a typical agent, according to Varshney (2024), the core acts the coordinator that manages the core logic and behaviour of the

agent in different situations. It is also where we define the overall goals and objectives of the agent, tools that will be used by the agent for execution, explanation for how to use different planning modules and in which situations, how to maintain relevance in memory from user queries, references past conversations, and define the persona of the agent based on the domain of interest (Varshney, 2024)

**ii. Memory Module**

Think of it as the store of the agent's internal logs as well as user interactions with agent (Varshney, 2024). According to Rana (2024), there are three types of agent memory namely: A short-term memory which stores the agent's current actions and thoughts allowing the agent to maintain a contextual understanding of the ongoing interaction, enabling seamless and coherent communication; a Long-term memory which records all the agent's interactions with users over an extended period, spanning weeks or even months and captures the history of conversations, preserving valuable context and insights gleaned from past exchanges; and a Hybrid memory that combines the advantages of both STM and LTM to enhance the agent's cognitive abilities.

**iii. Agent Tools**

In the context of LLM-Powered agents, tools refer to external resources, services, or APIs (Application Programming Interfaces) that the agent can utilize to perform specific tasks or enhance its capabilities. These tools serve as supplementary components that extend the functionality of the agent beyond its inherent language generation capabilities. Tools could also include databases, knowledge bases, and external models. For example, agents can employ a Retrieval Augmented Generation (RAG) pipeline for producing contextually relevant responses, a code interpreter for addressing programming challenges, an API for conducting internet searches, or even utilize straightforward API services such as those for weather updates or instant messaging applications (Rana, 2024).

**iv. The Planning Module**

This module enables the LLM-Powered agent to break down complex problems into small tasks that can be performed step by step (Varshney, 2024). For example, triage of a cyber threat to correlate it to an Indicator of Compromise (IoC) or an attacker may require a nuanced approach. It also enables the agent to engage in self-criticism and self-reflection over past actions, learns from mistakes, and refines future steps, thereby improving the quality of the final results (Rana, 2024). Techniques like ReAct, Reflexion, Chain of Thought, and Graph of thought have served as critic- or evidence-

based prompting frameworks. They have been widely used to improve the reasoning capabilities and responses of LLMs (Varshney, 2024).

**v. User Interface**

The user interface (UI) is the front-end component through which interactions between the user and the data occur. While the UI can be text-based, incorporating voice or even virtual reality elements can make the interaction more engaging and accessible for diverse users (Schroeder & Adesope, 2015).

The figures general architecture (2.1), user-driven (2.2) and event-driven (2.3) LLM-powered, RAG-based agent applications.

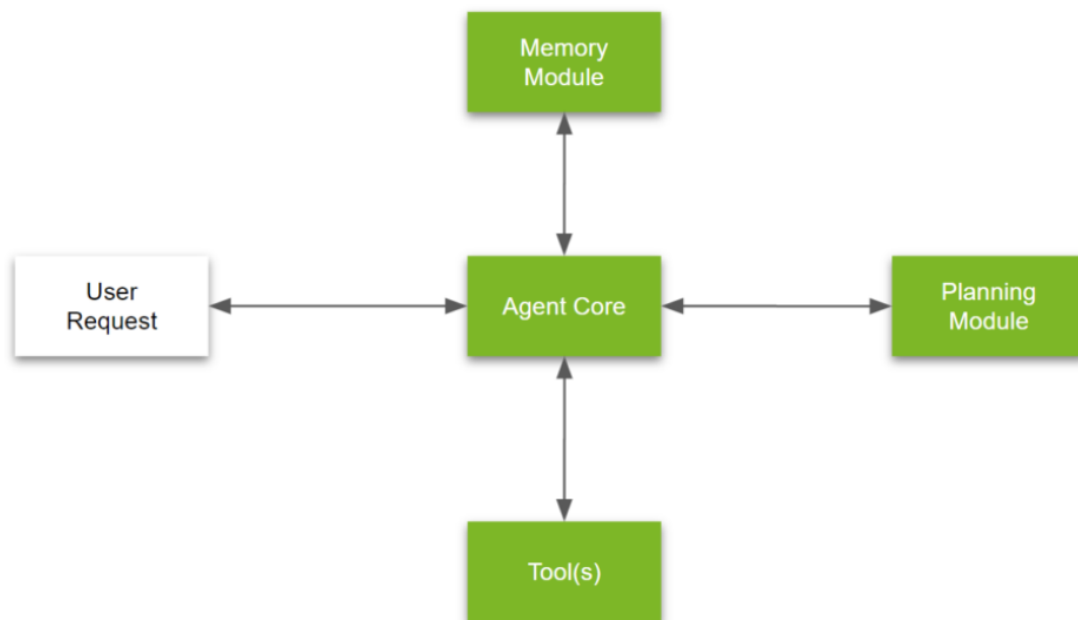


Figure 2.1: Generic Components of a RAG Agent (NVIDIA, 2024)

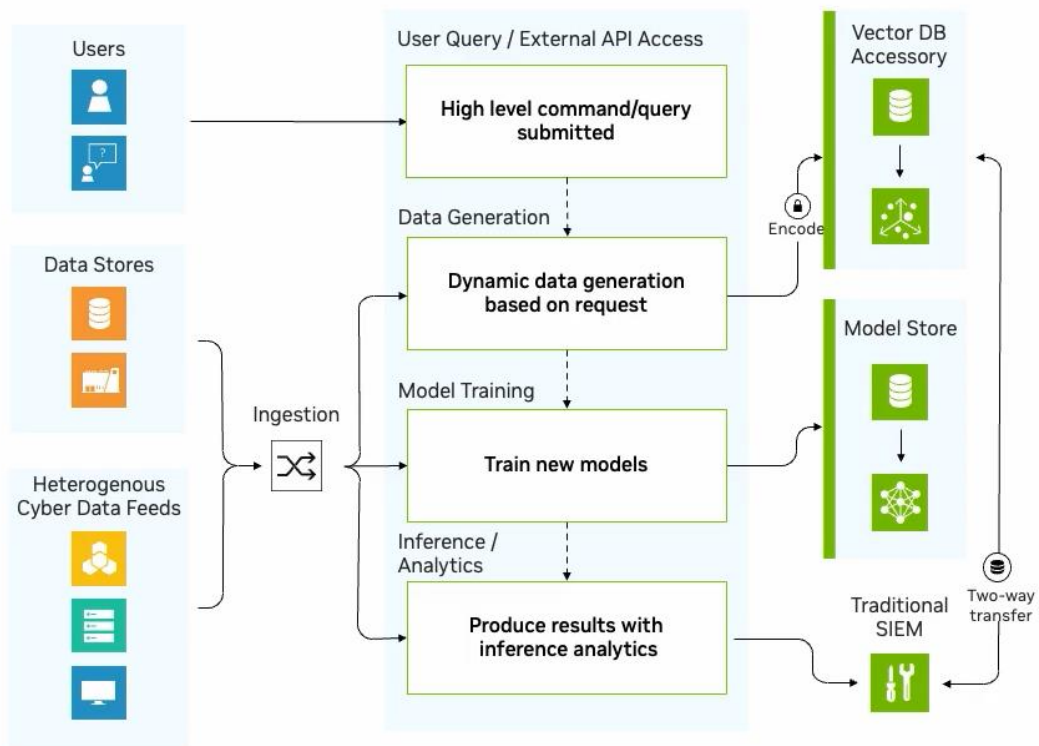
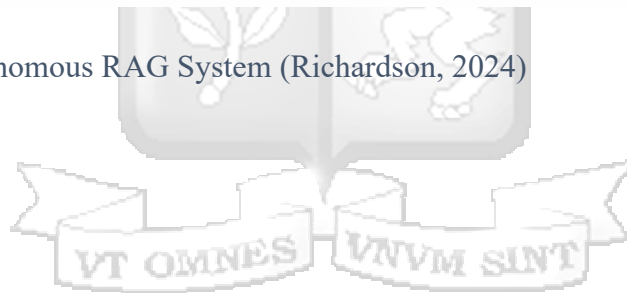


Figure 2.2: Non-Autonomous RAG System (Richardson, 2024)



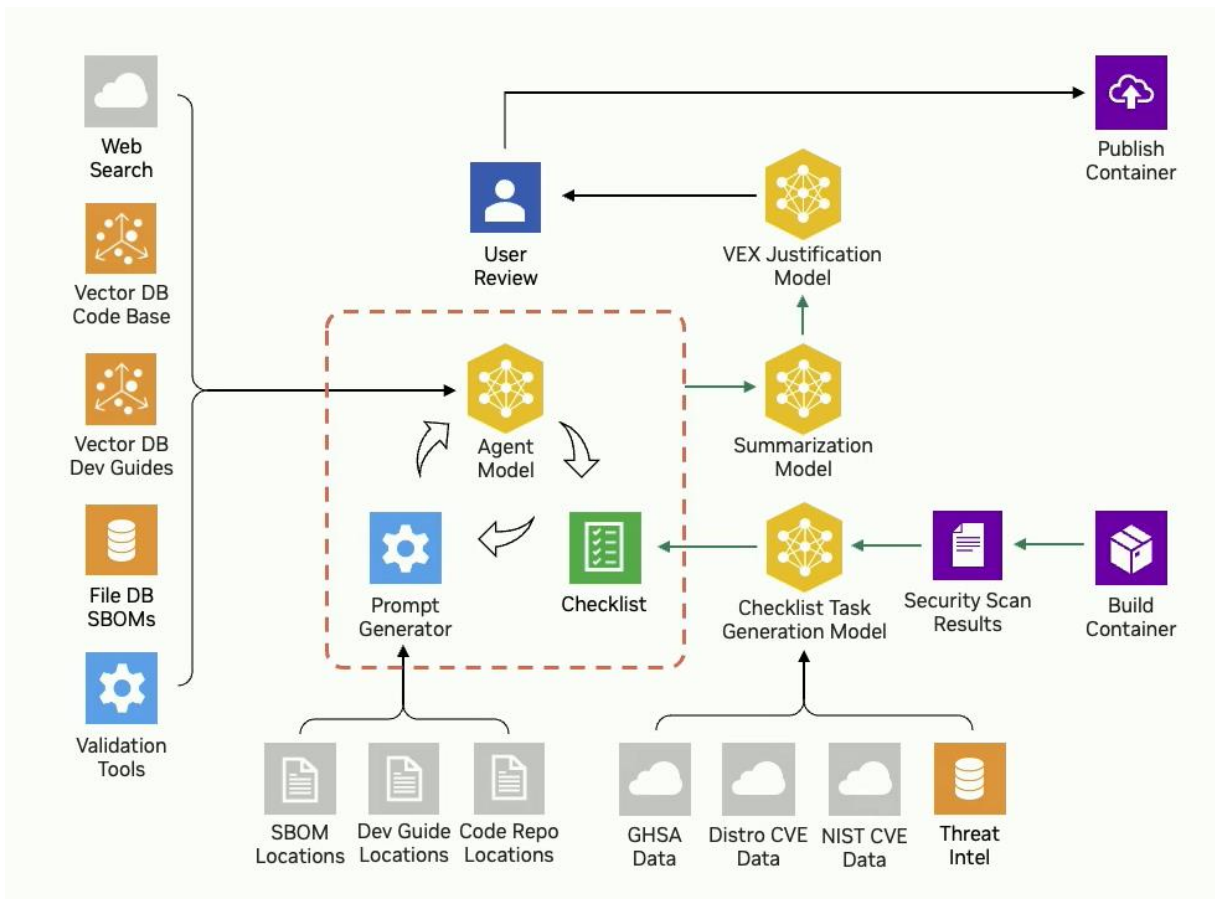


Figure 2.3: Autonomous RAG System (Allen and Demoret, 2024)

## 2.6 Existing Solutions

Table 2.1 indicates two (2) conversational Data Agents that are in use today especially in the SOC to support human Analysts.

Table 2.1: Existing Solutions

Solution	Description	Capabilities	Limitations
Charlotte AI	A generative AI Cybersecurity Analyst, CrowdStrike (Sentonas, 2023)	<ul style="list-style-type: none"> <li>Real-time insights.</li> <li>Faster decision making.</li> <li>Automated threat search and detection.</li> </ul>	<ul style="list-style-type: none"> <li>Enterprise-grade</li> <li>CrowdStrike-centric</li> <li>Not cloud-centric</li> </ul>

		<ul style="list-style-type: none"> <li>• Guided incidence response.</li> <li>• Third-party integration.</li> </ul>	
Security Co-Pilot	A generative AI cybersecurity solution to help Security Analysts (Caparas et al., 2024)	<ul style="list-style-type: none"> <li>• Incident summarization.</li> <li>• Impact analysis.</li> <li>• Reverse engineering of scripts.</li> <li>• Guided incidence response.</li> </ul>	<ul style="list-style-type: none"> <li>• Microsoft-centric.</li> <li>• Closed source.</li> <li>• Enterprise-grade.</li> </ul>

**2.7 Research Gap**

Many LLM-powered agents have limitations such as hallucination and sycophantic behaviour. Hallucination, within the context of GPT, refers to the tendency of the model to fabricate information or generate text that is not grounded in factual reality (Alkaissi & McFarlane, 2023). Sycophantic behaviour is where models may exhibit an inclination to produce content that excessively flatters or conforms to the preferences of the training data, potentially compromising the integrity and impartiality of generated text (Sufi, 2024). This research seeks to solve these problems by employing retrieval augmented generation (RAG). RAG enables the connection of the LLM-powered agent to live systems and tools such as the SIEM to allow real-time data access and retrieval. Using RAG is an effective strategy for ensuring LLMs agent responses are up-to-date and not hallucinated (Bagade, 2024).

**2.8 Conceptual Framework**

The conceptual framework for an LLM-powered Data Agent feature concepts such as Generative Artificial Intelligence, Human Centered Design, Retrieval Augmented Generation, Large Language Models, Cybersecurity, Threat modelling which have been reviewed in the previous sections. Figure 2.4 describes the LLM-powered conceptual framework for a Data Agent to augment human security analysts in the SOC.

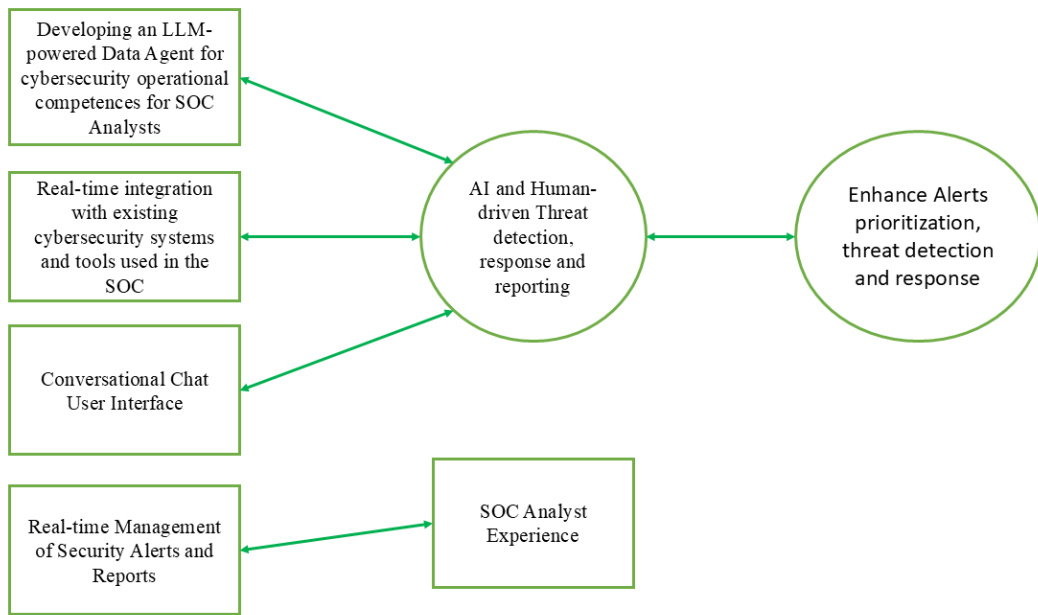


Figure 2.4: Conceptual Framework



## **Chapter 3: Research Methodology**

### **3.1 Introduction**

The methodology chapter of a research thesis is pivotal in delineating the framework through which the research questions are explored and answered. It serves not only to describe the methods employed in the investigation but also to justify their selection and to discuss their implementation in the context of the study. This introduction sets the stage for a detailed examination of the research methodology, grounding it within the broader discourse on research design and methodological rigor.

### **3.2 Research Design and Philosophy**

The development of a Data Agent aimed at enhancing the efficacy of human Analysts in the SOC, necessitates a methodological approach that is both innovative and rigorous. This research adopts a mixed methods design, integrating both qualitative and quantitative methodologies to explore the complexities inherent in the design, development, and evaluation of LLM-powered Data Agent. This mixed methods approach is underpinned by a pragmatic research philosophy, which prioritizes the research questions to guide the methodological choices, rather than adhering strictly to the conventions of either qualitative or quantitative paradigms (Creswell & Creswell, 2017).

### **3.3 Target Population / Sampling**

For the purpose of this study and specifically for pre- and post-evaluation of the Data Agent, a representative population sample of fourteen (14) security analysts was to be selected but that did not happen due to the sensitive nature of the SOC environment. This sampling process is crucial to ensure the agent's effectiveness in real-world scenarios. According to Patton (2002), the goal of sampling is to accurately represent the population of interest, in this case, security analysts working in the SOC. A stratified sampling method was to be employed to capture the diversity of security analysts in terms of their experience, expertise, and the types of threats they typically encounter. This approach is supported by Creswell (2014), who suggests that stratified sampling is suitable for ensuring the representation of diverse subgroups within a population.

### **3.4 Design Science Research (DSR)**

This research will adopt Design Science Research (DSR) which is an innovative and highly relevant methodology for the development and evaluation of a Data Agent. It provides a rigorous framework for creating and assessing artifacts designed to solve identified problems.

This approach is particularly suited to the field of Information Systems, where the development of practical and innovative solutions is paramount. By adopting DSR, the research not only contribute to theoretical knowledge but also provides tangible, practical outputs that enhance the performance of human analysts through automation (Hevner et al., 2004; Peffers et al., 2007).

### **3.4.1 The Rationale for Choosing Design Science Research**

DSR is grounded in the principle that knowledge is created through the building and evaluation of innovative artifacts. In the context of this thesis, the Data Agent represents such an artifact—a technological solution intended to address the challenge of alert fatigue and specifically, cyber threat investigation and incident response in the SOC. DSR is particularly suitable for this project because it emphasizes the importance of addressing real-world problems through the creation of innovative, practical solutions (Hevner, March, Park, & Ram, 2004).

### **3.4.2 Components of Design Science Research**

According to Henver et al. (2004), the DSR methodology encompasses several components, each of which contributed to the systematic development and testing of the Data Agent.

- i. **Problem Identification and Motivation:** This involved a detailed analysis of the SOC challenges that the Data Agent aims to address, such as threat investigation and incident response.
- ii. **Objectives of a Solution:** The goals of the Data Agent are defined in terms of functionality, effectiveness, and user satisfaction. These objectives are informed by the identified problems and the theoretical framework that guided this research.
- iii. **Design and Development:** This core component involved the actual creation of the Data Agent. It encompassed activities such as requirements gathering, design specification, prototyping, and testing. The design process was informed by best practices in GenAI, retrieval augmented generation and user interface design.
- iv. **Demonstration:** The Data Agent was tested in a controlled virtual environment mimicking a real-world setting. This stage assessed the artifact's functionality and its fit within the SOC environment.
- v. **Testing:** The effectiveness of the Data Agent was systematically tested against the objectives set forth in the solution's goals. The testing method used was user testing in a controlled virtual environment due to inaccessibility of a real SOC environment.

- vi. **Communication:** The knowledge gained from the development and testing of the Data Agent will be communicated to both management and practitioner audiences. This will include the publication of research findings in a peer reviewed journal and the sharing of practical insights and recommendations for the implementation of Data Agent in the SOC in workshops and conferences.

### **3.5 Data Collection**

The data collection phase of the thesis, focusing on the development of a Data Agent, was a crucial step in understanding the requirements, effectiveness, and user satisfaction associated with the use of the technology. Given the iterative nature of design science research (DSR), the data collection process was initially planned to be comprehensive, involving multiple methods to gather both qualitative and quantitative data. This approach was to ensure a deep understanding of the SOC operations and tools, user needs, and the impact of the Data Agent on the performance of human analysts. However, this exercise was not conducted due to inaccessibility of a real SOC environment.

The system used alerts data generated from the open-source Wazuh SIEM and processed through an API.

### **3.6 Data Analysis**

In the development of a Data Agent, data analysis plays a pivotal role in understanding user interactions, preferences, outcomes, and overall system performance. Given the complexity and multifaceted nature of this research, both qualitative and quantitative data was not collected from SOC analysts to inform the design and development of the Data Agent.

### **3.7 System Development Methodology**

The development of a Data Agent, a SOC tool designed to improve cyber threat investigation in the SOC, required a meticulous approach to system development. This section outlines the system development methodology that incorporates principles from both software engineering and Information Systems research that was used in the creation of an effective and user-friendly Data Agent.

#### **3.7.1 Agile Development Methodology**

The Agile Development Methodology, characterized by iterative and incremental development, was used for the development of the Data Agent. This approach allowed for continuous feedback and adaptation, which was crucial for aligning the system's features with the SOC goals and needs of human analysts (Beck et al., 2001; Highsmith, 2002).

- i. **Sprints and Iterations:** Development was divided into short sprints, enabling rapid prototyping and frequent reassessment of goals and priorities. This allowed for the iterative refinement of the Data Agent, incorporating feedback. (Schwaber & Beedle, 2002).
- ii. **User Stories and Backlogs:** User stories (i.e., brief descriptions of features from the perspective of the end user) guided development priorities. The product backlog—a prioritized list of user stories—ensured that the focus remained on delivering the most valuable features first (Cohn, 2004).
- iii. **Continuous Testing and Integration:** Continuous testing ensured that each component of the Data Agent functioned as expected, while continuous integration allowed for the seamless combination of new features with existing functionality (Duvall et al., 2007).

### 3.7.2 Design-Based Research (DBR)

Design-Based Research is particularly suitable for information systems projects like Data Agents. DBR involves iterative cycles of design, implementation, analysis, and revision, with the goal of developing practical technology solutions and advancing theoretical understanding (Wang & Hannafin, 2005; Anderson & Shattuck, 2012).

- i. **Collaborative Development:** Engaging stakeholders in the SOC especially human analysts in the development process was to ensure the Data Agent addresses real challenges and opportunities (Barab & Squire, 2004). However, this was not done as initially planned.
- ii. **Contextual Testing:** Evaluating the Data Agent in the SOC to understand its impact on threat detection and response, and identify areas for improvement (Cobb et al., 2003). Testing of the Data Agent was done by the researcher in a controlled virtual environment due to the sensitive nature of the SOC environments.
- iii. **Theoretical Integration:** Incorporating and testing information system theories within the design of the Data Agent to contribute to the academic field's body of knowledge (Sandoval, 2014) was done.

### 3.8 Data Agent System Development

The system development process for creating the Data Agent involved several key stages, from initial requirements gathering and analysis, design, development and testing. This process leveraged various programming tools and technologies to build a system that is both effective

in facilitating learning and robust in its technical implementation. The following is an overview of these stages and the tools used.

### **3.8.1 Requirements Analysis**

The initial stage involved gathering and analyzing requirements to understand what the users and other stakeholders would expect from the Data Agent. This included identifying the SOC goals, the agent functionalities, and any specific user needs.

### **3.8.2 System Design**

Based on the requirements analysis, the system's architecture, including its conversational capabilities, user interaction flow, and content delivery mechanisms, was designed. Tools like UML (Unified Modeling Language) diagrams and wireframing tools helped in visualizing the structure and user interface of the Data Agent as recommended by Rosenberg & Stephens (2007).

### **3.8.3 Tools Selection**

Identification of the most crucial tools required for the proper functioning of the Data Agent was important. The tools used to achieve this included open-source tools such as CALDERA adversary emulation platform, Suricata – Intrusion Detection System and Wazuh for Security Information and Events Management (SIEM). The Wazuh Indexer API was also used for alerts extraction and processing.

### **3.8.4 Development**

The Data Agent was programmed using a combination of AI models and LLM frameworks. The popular high-level programming language for AI development which is Python - was used with LLM frameworks such as langchain, langchain-community and streamlit which were used to develop the LLM engine and web-based chat interface. All these were installed in the Visual Studio (Code) integrated development environment.

### **3.8.5 Testing**

Testing was crucial to ensure the Data Agent is effective, user-friendly, and free of technical issues. This included unit testing and user testing.

## **3.9 Ethical Considerations**

The development of a Data Agent to augment SOC threat investigation raises several ethical considerations. One key concern is the potential for bias in the agent's decision-making processes, which could lead to inaccurate or unfair outcomes (Barocas et al., 2019). To mitigate

this risk, the agent was connected to an open-source Wazuh SIEM. Additionally, the agent's interactions with human analysts was designed to prioritize transparency and explainability, enabling analysts to understand the reasoning behind the agent's recommendations (Doshi-Velez et al., 2017). Furthermore, the agent's handling of sensitive threat intelligence and incident response data was subjected to strict confidentiality and data protection protocols, in accordance with industry standards and regulations (NIST, 2020).

The researcher sought ethical review and approval from the Strathmore University-Institutional Scientific Ethics Review Committee (SU-ISERC) which was granted to mitigate any ethical risks. A copy of the *Ethical Clearance Confirmation* letter is found in **Appendix B** of this Thesis.

### **3.10 Utilization of the Research Results**

The research results of this study are expected to contribute to the improvement of threat investigation processes in the SOC. According to Hevner et al. (2004), research in the field of information systems should aim to produce results that are applicable and usable in practice. The Data Agent developed in this study was designed to support security analysts in enhancing the efficiency and effectiveness of threat investigation. The results of this study will be utilized to inform further development of agents for use in the SOC and contribute to the body of knowledge of AI-driven cybersecurity, ensuring that they meet the needs and requirements of security analysts. Furthermore, the results will be used to evaluate the agent's performance and identify areas for improvement, as suggested by Sahu et al. (2018).

### **3.11 Dissemination of the Research Results**

The research results of this study will be disseminated through various channels to reach both academic and practitioner communities. According to Kitchenham et al. (2009), effective dissemination of research results is crucial to ensure their impact on practice. The results will be published in reputable academic journals and conferences, such as the Journal of Information Security and the International Conference on Information Systems Security. Additionally, the results will be presented at industry-focused conferences and workshops, such as the RSA Conference to reach security professionals and practitioners. Furthermore, the Data Agent developed in this study will be made available as an open-source tool, allowing SOC teams to integrate it into their existing threat investigation workflows. This approach is supported by von Hippel (2005) who argues that open-source software can facilitate the widespread adoption of innovative technologies.

## **System Design and Architecture**

### **3.12 Introduction**

This chapter details the manner in which the Data Agent was designed for use by Tier 1 SOC analysts during alerts triage in the security operation center (SOC), its general setup and the architecture employed to make the prototype ready for use, and the services offered by the Data Agent. The chapter will also discuss functional and non-functional requirements required by the system, the experimental procedures and designs used in this experiment.

### **3.13 System Requirements**

In brief, this includes a crucial three-sixty analysis of the problem or gap that this study sought to bridge by performing requirements gathering of existing information of systems while defining the actual requirements that make the Data Agent better compared to existing tools either in studies or in the enterprise. The features, components, services and limitations as far as these requirements are concerned is discussed under Functional and Non-functional requirements sections.

#### **3.13.1 Functional requirements**

These are requirements that outline the basic features that should be offered by the Data Agent. They include:

- i. The Data Agent should be deployable on premises, in the cloud or in a hybrid environment where the SOC is operated.
- ii. The Data Agent should not be detectable by malicious actors on the computer network.
- iii. The system should be a non-autonomous agent with human-in-the-loop or conversational interface.
- iv. The system should be able to give detailed explanations of attacks taking place or those that have been captured in the logs.
- v. The Data Agent should be connected to a live SIEM to for log extraction and analysis.

#### **3.13.2 Non-functional Requirements**

These are the requirements indicative of the performance metrics that the prototype Data Agent must satisfy.

- i. **Availability:** The Data Agent should be running and accessible by SOC Analysts whenever necessary.

- ii. Security: The Data Agent should be secured from malicious attacks especially AI vulnerabilities such as Prompt injection.
- iii. Capacity: The agent should be able to handle multiple and concurrent requests/interactions from SOC analysts and other systems.
- iv. Performance: By design, the Data Agent should service user requests seamlessly even with additional load.
- v. Scalability: The system must have the ability to expand or grow based on the demands or needs of the SOC.

### 3.14 System Architecture

The Data Agent is to be placed in the SOC environment and connected to a SIEM - in this case Wazuh and any other security tools to allow security logs to be ingested and to allow SOC analysts to query the alerts and generate reports within the computer network. This kind of setup ensures that the agent can learn from live data generated from these systems as SOC analysts interact with it.

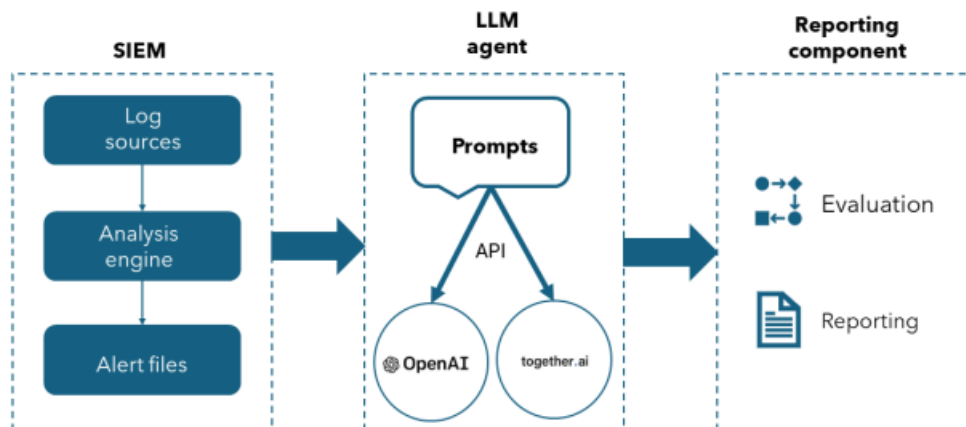


Figure 3.1: Alert Triage Architecture (Oniagbi, 2024)

The Data Agent design is largely based on the Human-in-the-Loop architecture for non-autonomous agents where the Human Analysts interacts with agent to guide its decisions. An example of this design was done by CrowdStrike for their non-autonomous Charlotte AI agent which interacts with their Falcon security solutions (CrowdStrike, 2023). Furthermore, as attested by Astle (2025), autonomous agents which operate without human oversight are not suitable for critical tasks that require precision, reliability and oversight such as security alerts triage and investigation.

This setup was done in a controlled lab environment with a Type 2 Hypervisor – VMWare Workstation which can be replicated in any networked environment depending the requirements of the SOC.

### **3.14.1 Inputs**

The prototype Data Agent design was done in a virtualized environment which essentially is host-level virtualization. The researcher used the MITRE ATT&CK framework which is used by cybersecurity teams to model attacks and threats. The MITRE ATT&CK framework, was created by the MITRE Corporation and released in 2013 (MITRE, 2025). It outlines the tactics, techniques and procedures used by adversaries to attack computer networks and systems.

### **3.14.2 Key processes**

The Data Agent has a conversational interface via chat where the SOC Analysts can enter their queries. This is done to allow implementation of the non-autonomous agent architecture and provide interface where the SOC analyst can interact and guide the agent in its decision-making process. The prototype Data Agent augments the human analyst who would have otherwise spent most of their time scrolling through thousands of alerts to find interesting alerts and then proceed to investigate the them. This is achieved through the following; an alert extraction and processing module, the LLM agent core and report generation and chat interface.

### **3.15 System Design Tools**

System design is meant to capture elements of the prototype Data Agent especially, the architecture, modules and components and data that goes through this agent. The prototype used a virtualized-controlled environment in order to overcome resource limitation that would not be the case for well-established SOCs.

#### **3.15.1 Flow Chart Diagram**

Similar to the process used by Ban et al. (2021) and in conventional SOC environments, logs were collected and forwarded to the Wazuh server from two different endpoints using Wazuh agents installed on them. The endpoints had different operating systems installed, Windows Desktop and Ubuntu Linux, the most popular desktop operating systems in use in the world today according to Statcounter (2024). The versions were Windows 11 and Ubuntu Linux 22. These logs are then loaded and prepared for processing by the agent's log processing engine. The human SOC Analyst can also use the search tool to perform general queries regarding for example attacks and techniques used by adversaries. Figure 4.2 shows the order of events from log generation to querying of the logs by the human SOC analyst.

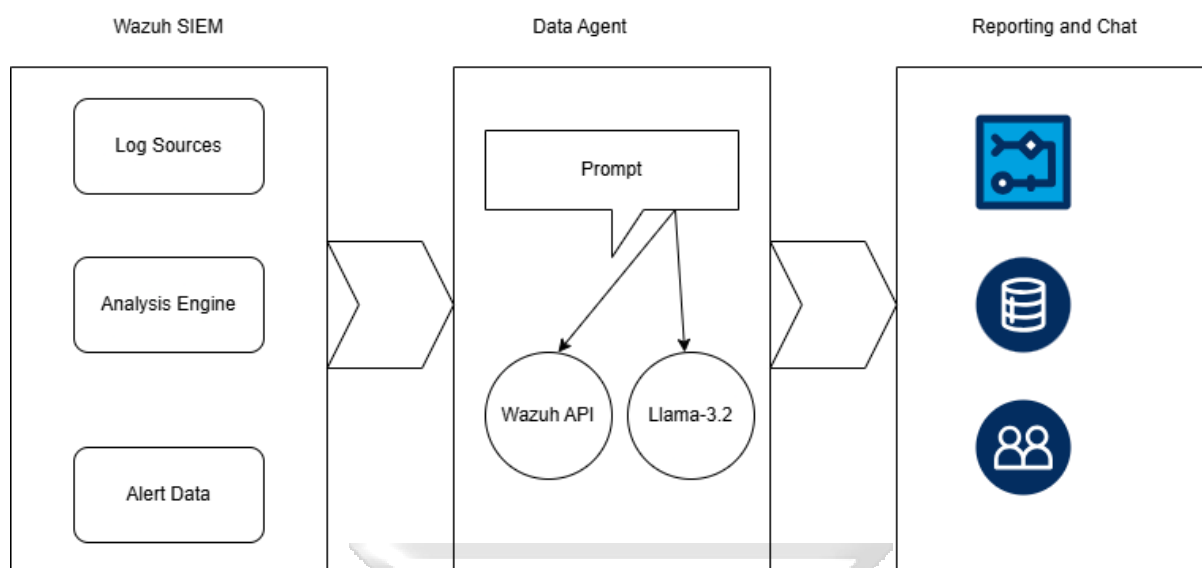


Figure 3.2: Data Agent Architecture

### 3.15.2 Use Case Diagram

In Unified Modelling Language (UML) the use case diagrams are used to show interactions between the actors and the system. This prototype had 2 primary actors and a 3rd secondary actor.

The first primary actor: the adversary who attacks the enterprise network or systems leading to alert generation, and the second primary actor: the human SOC Analyst who triage and investigate the alerts, and the secondary actor is the RAG which interacts with the SOC Analyst through a chat interface to classify the alerts. The adversary is considered a primary actor as she is responsible for triggering alerts, while the SOC Analyst interacts with and maintains the prototype as they gather more intelligence from the RAG. In case of an incident or attack as emulated by CALDERA, an alert is generated and forwarded to the Wazuh SIEM which is connected to the prototype Data Agent and the alert is extracted. The SOC analyst interacts with the Data Agent to triage the alerts and consults external knowledge bases and tools via the RAG for more intelligence. Figure 4.3 shows the flow of events together with the actors in the Data Agent.

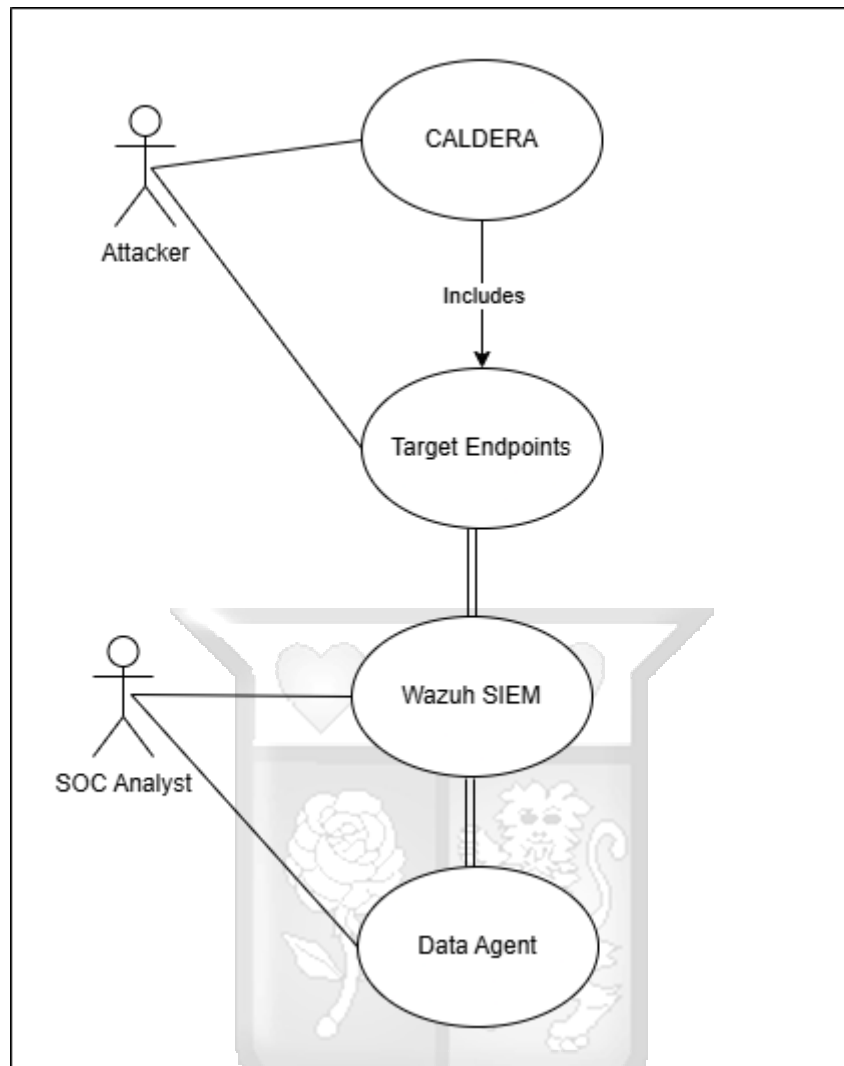


Figure 3.3: Process Flow Diagram

### 3.16 Web Chat Interface

The SOC Analyst who is a primary user of the prototype Data Agent accesses the system via a web interface using an IP to the LLM Agent server. Web login is not enabled in the Chat interface because this is a prototype and the assumption is the SOC Analyst will be able to access the agent once they are logged into the SOC ecosystem.

### 3.17 Storage

The prototype Data Agent was created on a virtualized-controlled environment with several Virtual Machines that had storage in the form of virtual hard disks. These VMs hosted the Wazuh SIEM, Adversary emulation tool and the Wazuh agents/endpoints to allow the agent to run smoothly albeit with a few challenges. The advantage with Virtualized infrastructure is the flexibility they offer in terms of either removing/creating or quickly adding up more storage space for the prototype on the go.

## Chapter 4: System Implementation and Testing

### 4.1 Introduction

The focus of this chapter is mainly on setting up and implementing the prototype Data Agent, which builds up from the previous chapter. The aim is to elaborate on the steps involved in the setup of the controlled virtual environment, the development and testing of the prototype, and presentation of results.

### 4.2 System Implementation

This prototype Data Agent runs partly in a controlled virtual environment built with VMWare Workstation which is a Type 2 Hypervisor, which suitable for prototyping. For purposes of this research, the Wazuh and CALDERA server, and the endpoints are all hosted on virtual machines in VMWare due to financial constraints but mostly for the flexibility virtual environments offer, in terms of access, scalability and portability. The Data Agent deployed used Wazuh SIEM to collect logs and generate alerts from the Windows and Ubuntu Linux endpoints. The SIEM is also connected to other tools such as Suricata for network and host Intrusion Detection. The alerts are generated by the adversary emulation tool - CALDERA which is based on the MITRE ATT&CK cybersecurity framework. The emulated attacks are targeted at the two endpoints which send their logs directly to Wazuh via the agents installed on them.

The LLM-powered Data Agent codebase is hosted on and run from the Windows host which also hosts the Type 2 hypervisor. This code includes the alerts extraction and processing module – log data extracted Wazuh Server API, the RAG system – agent core, and the web chat interface module. The following sections will discuss the hardware and software components of the prototype Data Agent.

#### 4.2.1 Hardware Environment

The researcher used his personal laptop which had the specifications as outlined. These hardware requirements were sufficient to support the controlled virtual environment used to host and run the prototype Data Agent and its components.

- i. 32GB RAM
- ii. Intel Core i7 Processor
- iii. 500GB Solid-state drive storage
- iv. Virtualization hardware

### 4.2.2 Software Environment

For the prototype Data Agent to be designed to function optimally in the virtualized test environment, the following software requirements were also met on the host machine.

- i. Windows 11 Professional 64-bit
- ii. Brave browser, 64-bit, however any web browser could be used.
- iii. VMWare Workstation Pro for hosting the Virtual Machines
- iv. VS Code for Windows, 64-bit
- v. Python for Windows - version 3.12 (64-bit)
- vi. Ollama for local LLM access

### 4.2.3 Virtual Environment

Due to financial constraints, the researcher used virtualized computing infrastructure hosted on a personal computer, which means not paying for compute and storage services, where one could create and delete virtual machines as necessary. This also supported agile development and continuous improvement/continuous development making the project quite portable. Since the prototype could not be built in a live environment, the research relied heavily on the local VMWare network connectivity similar to on-premise, cloud or hybrid networks in production environments. The building blocks included one Windows VM and one Ubuntu 22.04 desktop VM acting as Wazuh agents/endpoints, one Ubuntu 22.04 server VM hosting the Wazuh Server, one Debian 12 VM hosting CALDERA Server.

The virtual machines had the following minimum required specifications based on its function in Table 5.1.

Table 4.1: Virtual Machines Specifications

<b>VM</b>	<b>vCPUs</b>	<b>RAM (GB)</b>	<b>Storage (GB)</b>
Windows 11	1	2	20
Ubuntu 22.04 Desktop	1	2	20
Wazuh Server	2	4	40
CALDERA Server	1	4	20

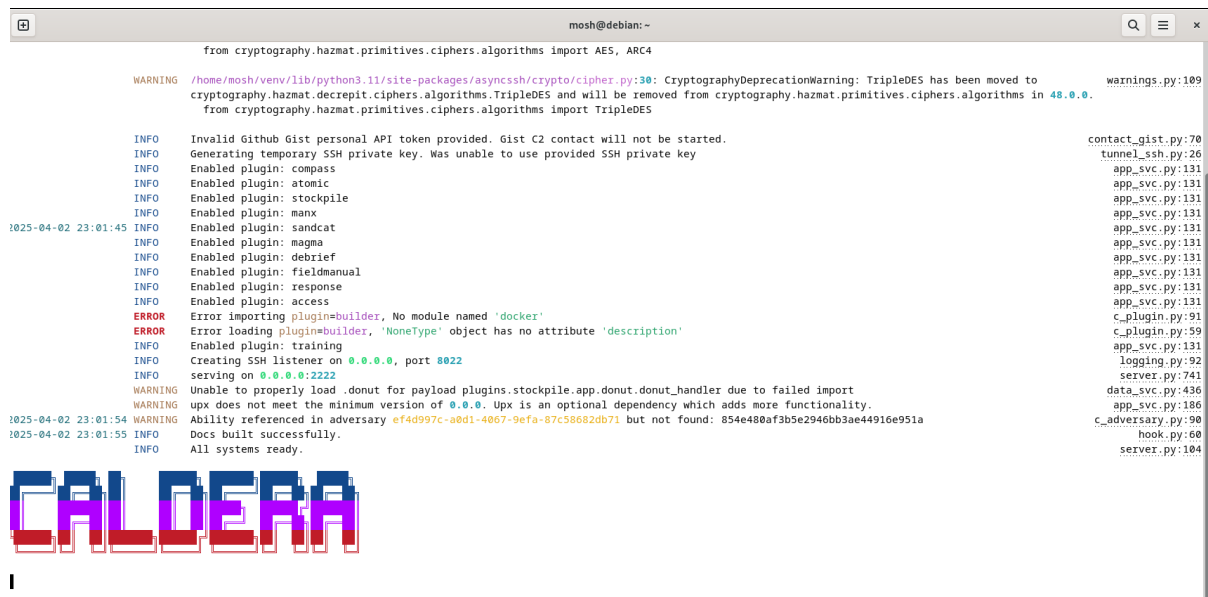
### 4.3 System components and modules

CALDERA was used to emulate adversarial attacks on the endpoints that send log data to the Wazuh Server generating alerts that are displayed on the Wazuh SIEM dashboard. This shows SOC Analysts what is happening on the network just like in a conventional SOC. The researcher was mostly interested in alerts that indicate potential security attacks on the systems and networks. Once the alerts are generated, the human SOC analyst can query the SIEM via

the prototype Data Agent to identify these potential attacks or incidences for further investigation.

### 4.3.1 Deploying the CALDERA Server

Virtual machines were used to host the CALDERA server which acts as the adversary Command and Control (C2) machine and the target agents/endpoints. For the purpose of this study, which is a proof of concept, the researcher added two agents to CALDERA that were used to emulate various types of attacks. The following figures 5.1 to 5.6 show the implementation of the CALDERA adversary emulation platform.



```
mosh@debian:~$ python3
from cryptography.hazmat.primitives.ciphers.algorithms import AES, ARC4

WARNING /home/mosh/venv/lib/python3.11/site-packages/asynssh/crypto/cipher.py:30: CryptographyDeprecationWarning: TripleDES has been moved to
cryptography.hazmat.decrepit.ciphers.algorithms.TripleDES and will be removed from cryptography.hazmat.primitives.ciphers.algorithms in 48.0.0.
from cryptography.hazmat.primitives.ciphers.algorithms import TripleDES

INFO Invalid Github Gist personal API token provided. Gist C2 contact will not be started.
INFO Generating temporary SSH private key. Was unable to use provided SSH private key
INFO Enabled plugin: compass
INFO Enabled plugin: atomic
INFO Enabled plugin: stockpile
INFO Enabled plugin: manx
1025-04-02 23:01:45 INFO Enabled plugin: sandcat
INFO Enabled plugin: magma
INFO Enabled plugin: debrief
INFO Enabled plugin: fieldmanual
INFO Enabled plugin: response
INFO Enabled plugin: access
ERROR Error importing plugin=builder, No module named 'docker'
ERROR Error loading plugin=builder, 'NoneType' object has no attribute 'description'
INFO Enabled plugin: training
INFO Creating SSH listener on 0.0.0.0, port 8022
INFO serving on 0.0.0.0:2222
WARNING Unable to properly load .donut for payload plugins.stockpile.app.donut.donut_handler due to failed import
upx does not meet the minimum version of 0.0.0. Upx is an optional dependency which adds more functionality.
1025-04-02 23:01:54 WARNING Ability referenced in adversary ef4d997c-a0d1-4067-9efa-87c58682db71 but not found: 854e480af3b5e2946bb3ae44916e951a
1025-04-02 23:01:55 INFO Docs built successfully.
INFO All systems ready.
```

**CALDERA**

Figure 4.1: Starting the Caldera Server

The researcher logged into CALDERA using a red team user name and password to mimic an adversary and to deploy agents as shown in figure 5.2.

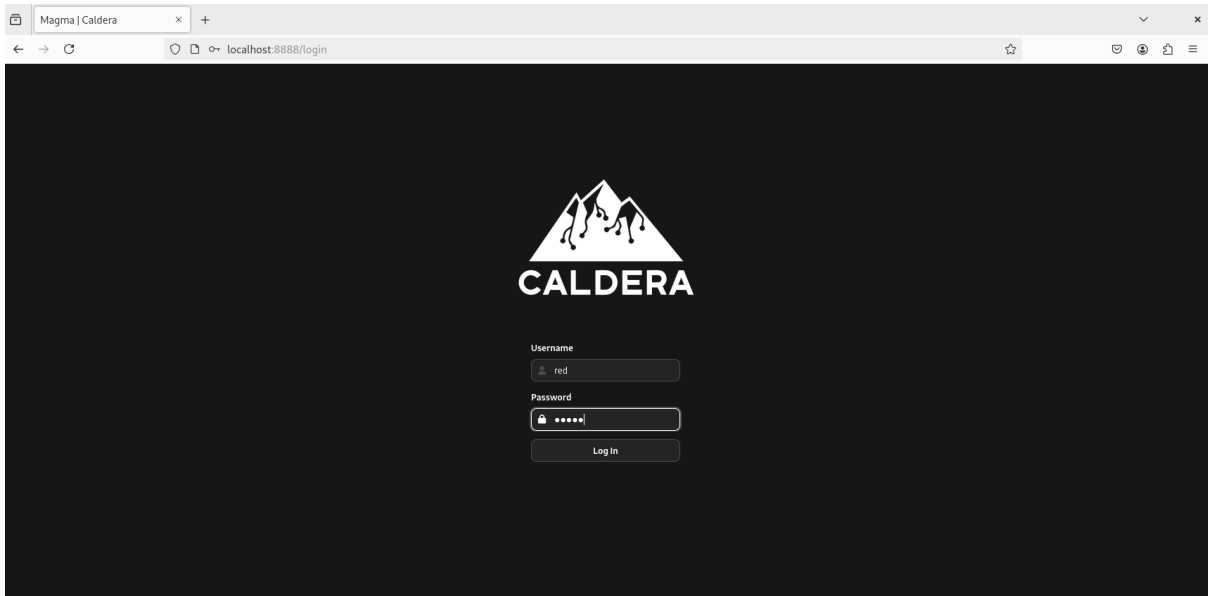


Figure 4.2: Caldera Admin Login Page

Once logged in, the researcher added the agents as shown in figure 5.3 and 5.4.

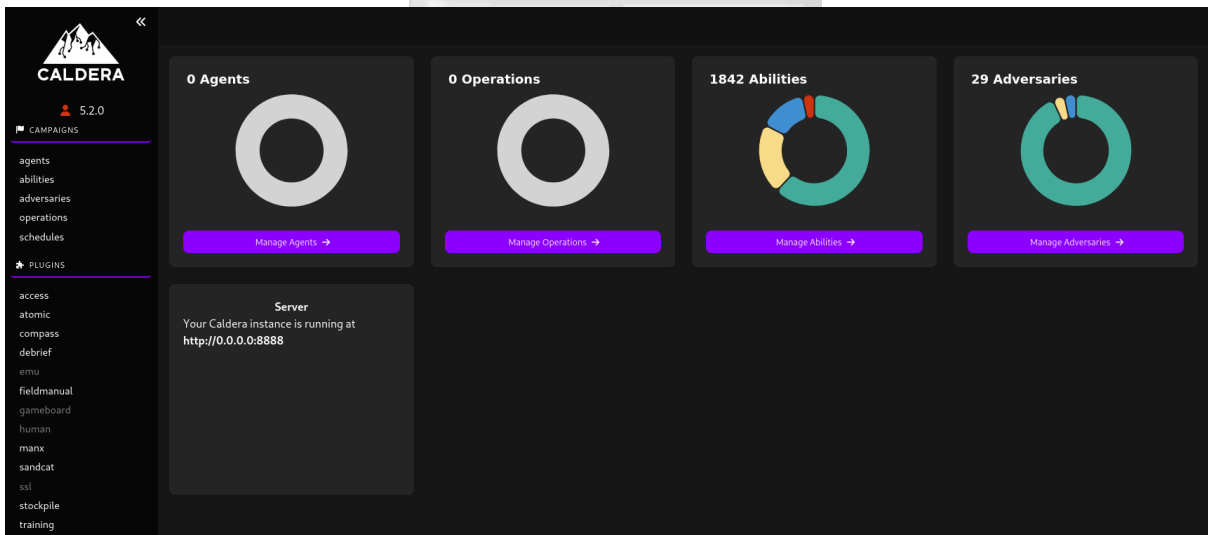


Figure 4.3: Caldera Dashboard without Agents

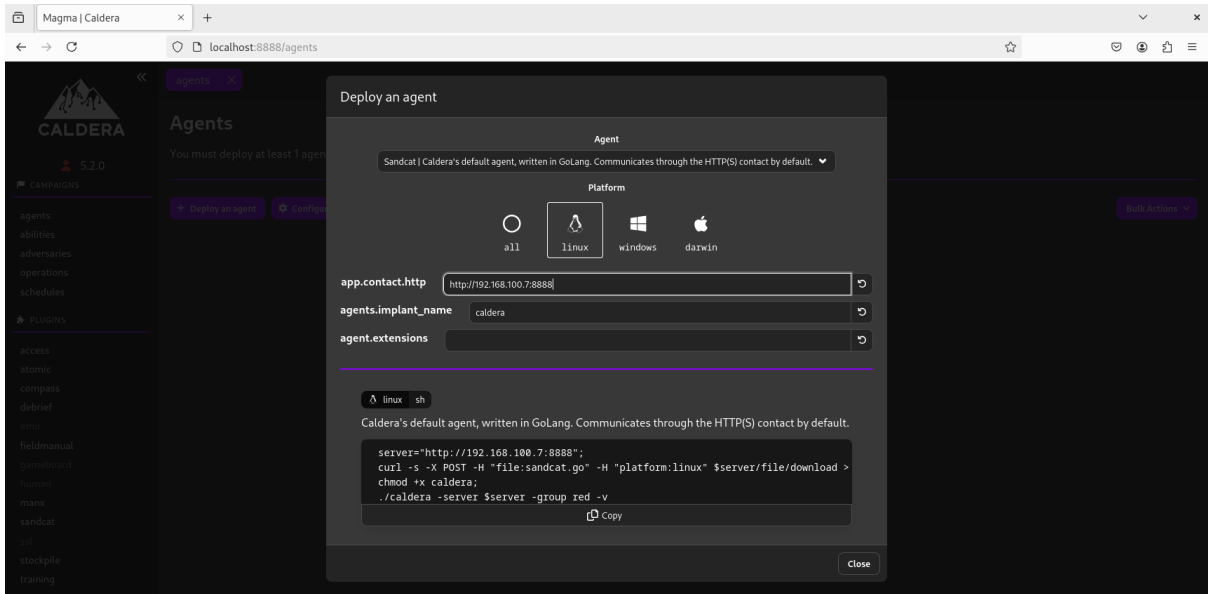


Figure 4.4: Deploying a Linux Agent

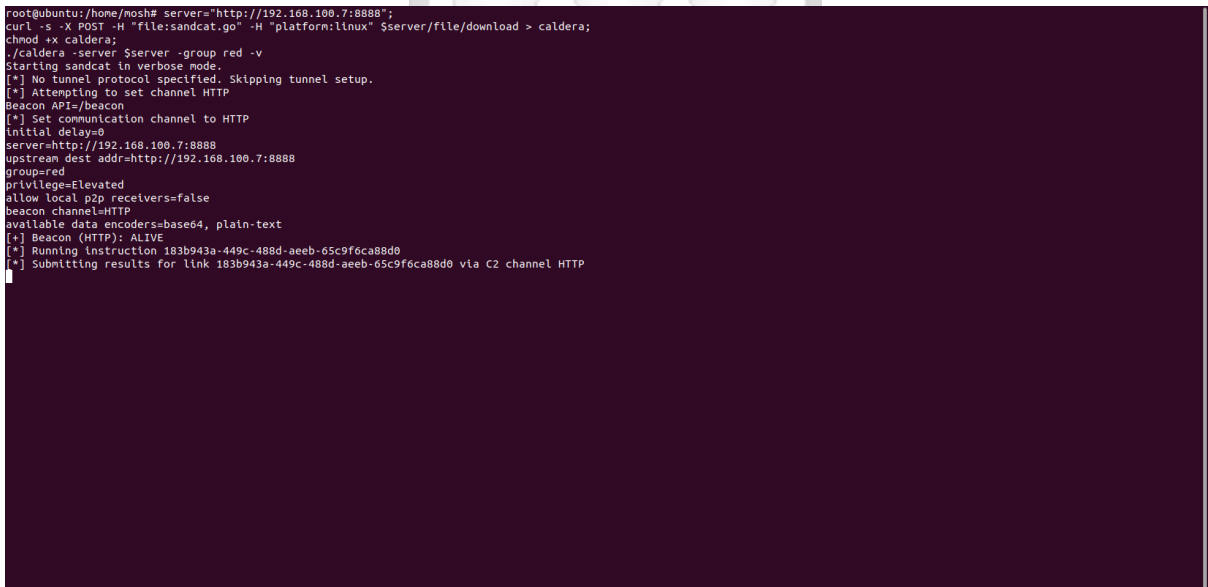


Figure 4.5: Starting the Linux Agent

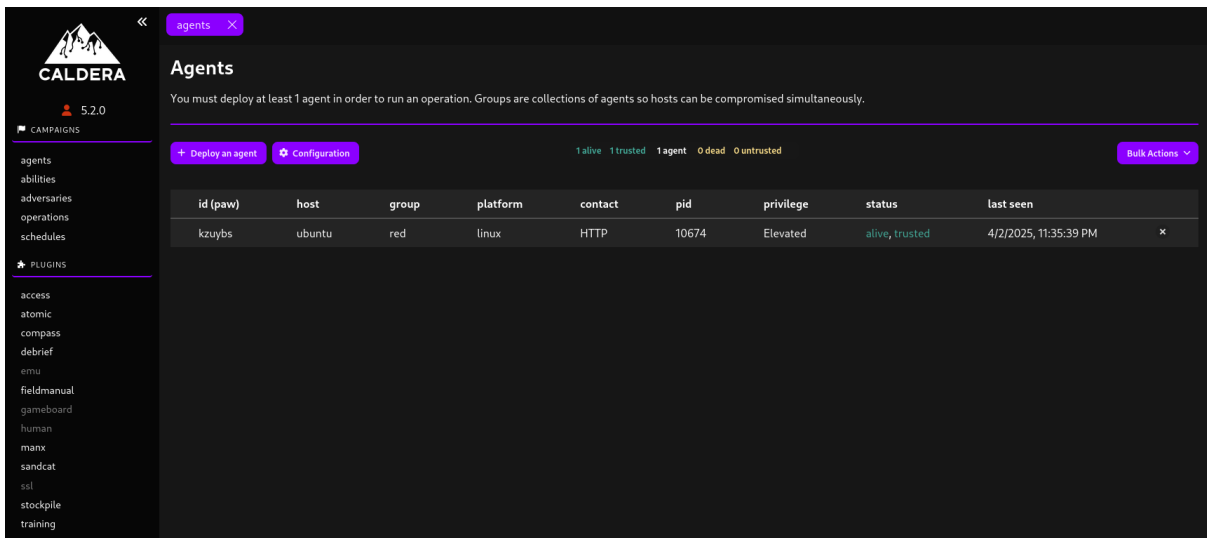


Figure 4.6: Caldera Dashboard Showing the Deployed Agent

### 4.3.2 Deploying the Wazuh Server

Wazuh SIEM was also deployed in a virtual machine hosted on VMWare Workstation. Again, two agents were deployed to the Wazuh Server to enable the generation and sending of log data. The Wazuh dashboard showed insights with confidence on what each alert and log was all about, from which the SOC analyst can choose what to monitor. The researcher logged into the Wazuh server, first to deploy the agents and second to view the alerts generated as a result. The following figures 5.7 to 5.11 show the implementation of the Wazuh SIEM.



Figure 4.7: Logging into the Wazuh Server

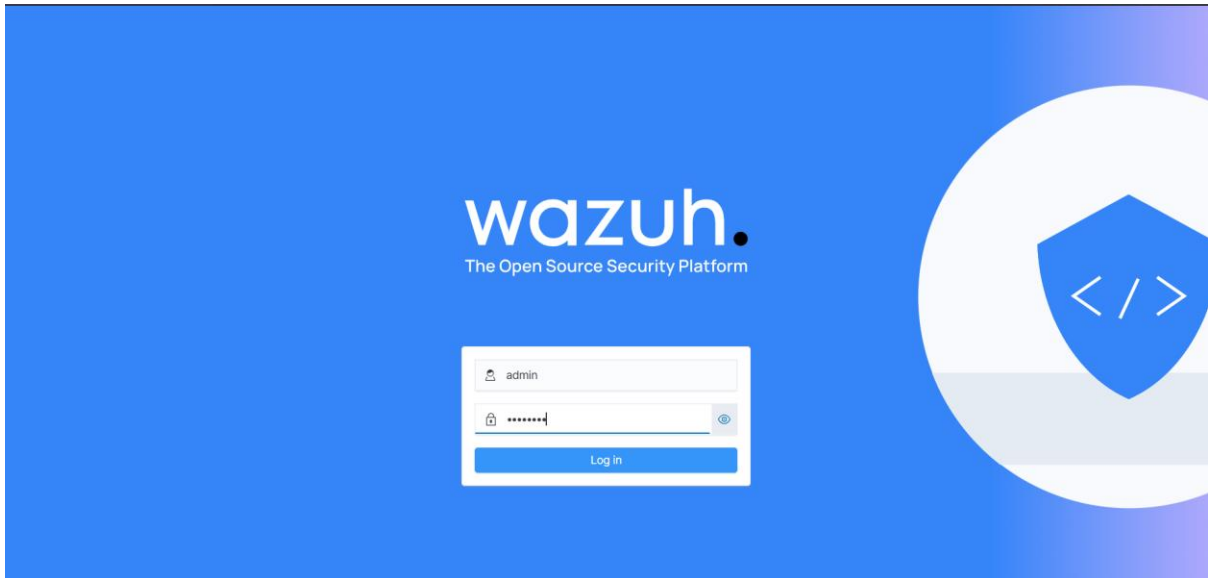


Figure 4.8: Wazuh SIEM Admin Login Page

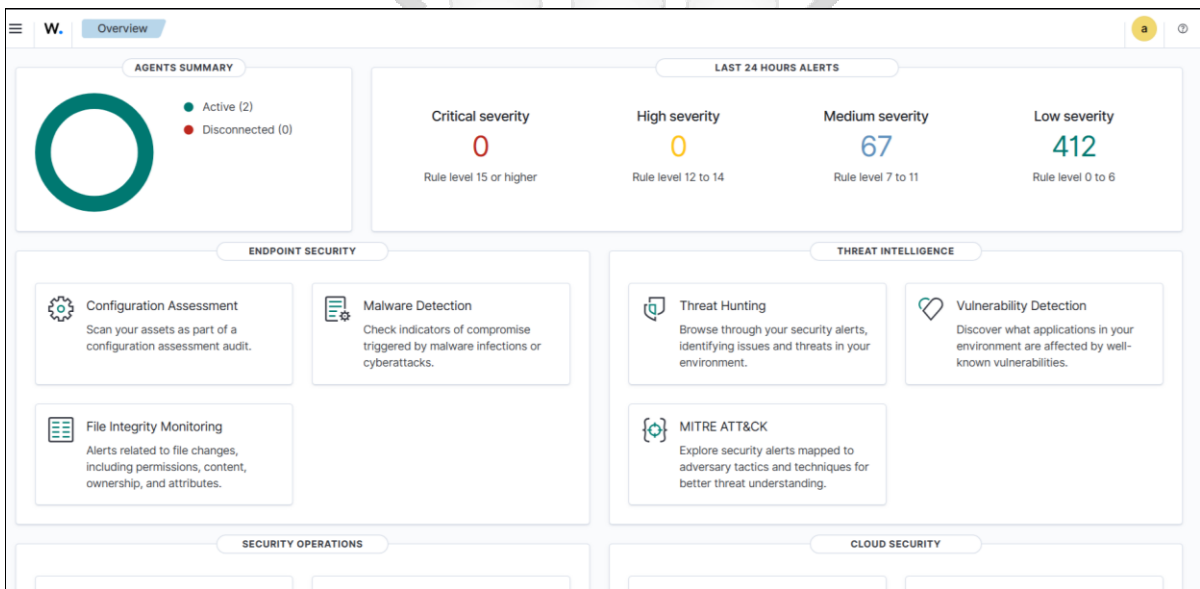


Figure 4.9: Main Wazuh Dashboard Showing Deployed Agents

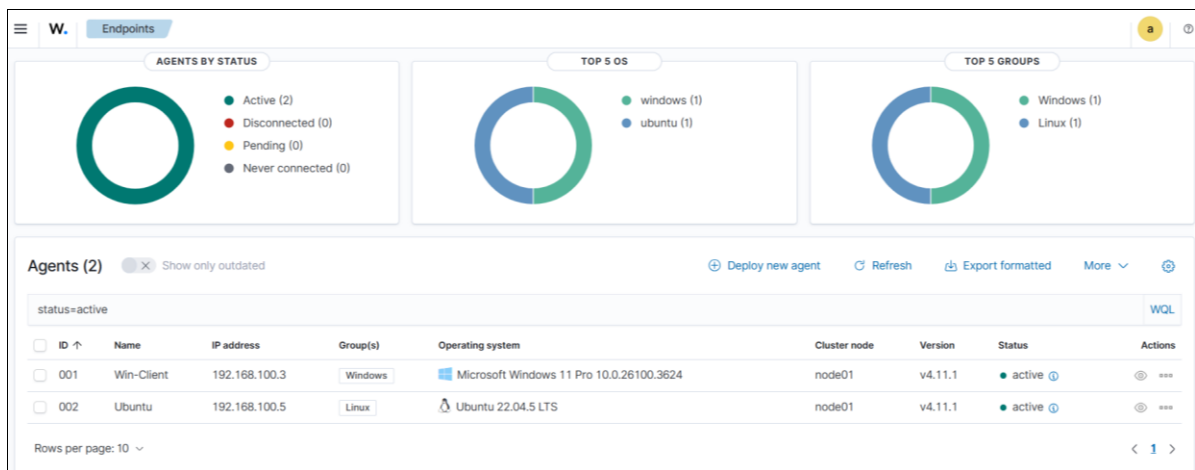


Figure 4.10: Wazuh Endpoints Dashboard

t agent.id	002
t agent.ip	192.168.100.5
t agent.name	Ubuntu
t data.dstuser	mosh(uid=1000)
t data.uid	0
t decoder.name	pam
t decoder.parent	pam
t full_log	Apr 03 03:28:47 ubuntu systemd[1982]: pam_unix(systemd-user:session): session opened for user mosh(uid=1000) by (uid=0)
t id	1743652088.31249
t input.type	log
t location	journald
t manager.name	ubuntu
t predecoder.hostname	ubuntu
t predecoder.program_name	systemd
t predecoder.timestamp	Apr 03 03:28:47
t rule.description	PAM: Login session opened.
# rule.firedtimes	1
t rule.gdpr	IV_32.2
t rule.gpg13	7.8, 7.9

Figure 4.11: Wazuh Alert Page

The Admin can also change the username and password of CALDERA and Wazuh servers at their prerogative.

### 4.3.3 The Data Agent implementation

The researcher used Python programming language which is a general-purpose language suitable for building AI applications. Visual Studio code (VS Code) was used as the Integrated Development Environment (IDE). Several frameworks and libraries were installed including but not limited to langchain, langchain-community, langchain-ollama to allow interaction with Ollama, Requests Python library for making http requests to the Wazuh Server Application Programming Interface (API).

The agent was implemented in three modules – alerts extraction and processing, the RAG system which is the agent core and the frontend web-based chat interface which was implemented using the streamlit framework. Figure 5.7 shows code snippet for the implementation of the alert extraction and processing module from the Wazuh Indexer API.

```
app.py > get_wazuh_alerts
19 urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
20
21 # Declare Wazuh Indexer API variables - URL, Username and Password
22 WAZUH_API_URL = "https://192.168.100.15:9200/wazuh-alerts-4.x-2025.05.07/_search"
23 USERNAME = "BASIC_AUTH_USERNAME"
24 PASSWORD = "BASIC_AUTH_PASSWORD"
25
26 # Groq API Key configuration from the .env file
27 GROQ_API_KEY = os.getenv("GROQ_API_KEY")
28
29 # The Function for fetching alerts from the Wazuh Indexer API
30 def get_wazuh_alerts():
31     """Retrieve alerts from Wazuh indexer API"""
32     try:
33         # Define the query to filter by specific rule IDs
34         query = {
35             "query": {
36                 "terms": {
37                     "rule.id": ["751", "752", "5503", "60106", "60122"]
38                 }
39             }
40         }
41         response = requests.get(WAZUH_API_URL, auth=(USERNAME, PASSWORD), verify=False, json=query)
42         response.raise_for_status()
43         # Save to JSON file
44         alerts = response.json()
45         with open('wazuh_alerts.json', 'w') as f:
46             json.dump(alerts, f, indent=2)
47
48         return alerts
49     except Exception as e:
50         return f"Error retrieving alerts: {str(e)}"
```

Figure 4.12: Code Snippet for Alerts Extraction and Processing Module

## 4.4 System Testing

The system was tested and seen to emulate attacks, send the alerts to the Wazuh SIEM, extract the alerts and query the alerts using the Data Agent providing information about both interesting and not interesting alerts. The Ubuntu Linux and Windows endpoints connected to

the Wazuh Server were made vulnerable by having open ports such as SSH which is commonly used in various networked environments to not only perform operations legitimate admin tasks but also adversarial attacks such as brute-force attacks. Brute-forcing is a common tactic used by adversaries to gain unauthorized access to vulnerable Windows and Linux systems that have an open Secure Shell (SSH) port (Wazuh, 2025). The whole system was seen to work as demonstrated in figure 5.10 where it gives details of alerts as the emulated attacks are conducted. Figures 5.13 to 5.16 show the full testing process of the Data Agent query and response with an emulated brute-force attack.

```

mosh@debian: ~
root@debian:/home/mosh# ssh mosh@192.168.100.5
The authenticity of host '192.168.100.5 (192.168.100.5)' can't be established.
ED25519 key fingerprint is SHA256:PVj10Jf7059nLEJqgIng3bNCHSVhIQ+cURySRrFnp0.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '192.168.100.5' (ED25519) to the list of known hosts.
mosh@192.168.100.5's password:
Permission denied, please try again.
mosh@192.168.100.5's password:
Permission denied, please try again.
mosh@192.168.100.5's password:
mosh@192.168.100.5: Permission denied (publickey,password).
root@debian:/home/mosh# ssh mosh@192.168.100.5
mosh@192.168.100.5's password:
Permission denied, please try again.
mosh@192.168.100.5's password:
Permission denied, please try again.
mosh@192.168.100.5's password:
mosh@192.168.100.5: Permission denied (publickey,password).
root@debian:/home/mosh# ssh mosh@192.168.100.5
mosh@192.168.100.5's password:
Permission denied, please try again.
mosh@192.168.100.5's password:
Permission denied, please try again.
mosh@192.168.100.5's password:
mosh@192.168.100.5: Permission denied (publickey,password).
root@debian:/home/mosh# hydra -l badguy -P password_list.txt 192.168.100.5 ssh
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-04-03 00:56:30
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 8 tasks per 1 server, overall 8 tasks, 8 login tries (1:1/p:8), ~1 try per task
[DATA] attacking ssh://192.168.100.5:22/
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-04-03 00:56:33
root@debian:/home/mosh#

```

Figure 4.13: Initiating a Brute-force attack

timestamp	agent.name	rule.description	rule.level	rule.id
Apr 3, 2025 @ 07:56:33.392	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710
Apr 3, 2025 @ 07:56:33.392	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710
Apr 3, 2025 @ 07:56:33.392	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710
Apr 3, 2025 @ 07:56:33.392	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710
Apr 3, 2025 @ 07:56:33.392	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710
Apr 3, 2025 @ 07:56:33.392	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710
Apr 3, 2025 @ 07:56:33.392	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710
Apr 3, 2025 @ 07:56:33.347	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710
Apr 3, 2025 @ 07:56:31.398	Ubuntu	PAM: User login failed.	5	5503
Apr 3, 2025 @ 07:56:31.394	Ubuntu	PAM: Multiple failed logins in a small period of time.	10	5551
Apr 3, 2025 @ 07:56:31.389	Ubuntu	PAM: User login failed.	5	5503
Apr 3, 2025 @ 07:56:31.385	Ubuntu	PAM: User login failed.	5	5503
Apr 3, 2025 @ 07:56:31.380	Ubuntu	PAM: User login failed.	5	5503
Apr 3, 2025 @ 07:56:31.376	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710
Apr 3, 2025 @ 07:56:31.374	Ubuntu	ssh: Attempt to login using a non-existent user	5	5710

Figure 4.14: Wazuh Showing Alerts Triggered by the Emulated Brute-force Attack

The Data Agent was also tested by asking basic queries to summarize login attempts and windows registry events as shown in figures 5.15 to 5.18. The agent's responses are also shown.

Before the development of the Data Agent, it was found that most solutions mentioned in the Related works section often suffered from hallucination and sycophantic behavior. According to Alkaissi & MacFarlane (2023), hallucination within the context of LLMs, refers to the tendency of the model to fabricate information or generate text that is not grounded in factual reality. Sycophantic behaviour on the other hand, is where models may exhibit an inclination to produce content that excessively flatters or conforms to the preferences of the training data, potentially compromising the integrity and impartiality of generated text (Sufi, 2024). This Data Agent was connected to a live SIEM to solve these problems.

When the Data Agent was queried by asking it to summarize login attempts resulting from the emulated brute-force attack, the agent was able to summarize the attacks by providing details of the target agent affected – Windows or Linux, the source IP address of the machine that initiated the attack, and the username of the potential attacker. This information is mostly used by a SOC analyst during threat investigation to gather more information about attackers or to identify if an alert is a False positive. This alone reduces the Mean-Time-To-Investigate (MTTI) alerts from an average of fifteen to five minutes where the Data Agent was able to provide accurate details of the alerts and identify information missing from the alert. Mean-Time-To-Investigate is the average time from the time a threat has been detected until the SOC team initiates investigation. It bridges the gap between MTTD (Mean-Time-To-Detect) and the start of MTTR (Mean-Time-To-Resolve), outlining the initial response phase (Splunk, 2025).

# Agent TOI

Hello 🙌. I'm your Data Agent. How can I help you today?

Refresh Alerts

Enter your query:

summarize the login attempts

Results of the Analysis:

There were multiple login attempts from the IP address 10.10.10.67 to a Debian system using SSH. The attempts were made with an invalid username "badguy" from different ports (48676, 48688, and 48680). All attempts were unsuccessful. The attempts are suspected to be part of a password guessing attack.

Figure 4.15: Generic login attempts query and response

# Agent TOI

Hello 🙌. I'm your Data Agent. How can I help you today?

Refresh Alerts

Enter your query:

summarize the login attempts including number of times fired

Results of the Analysis:

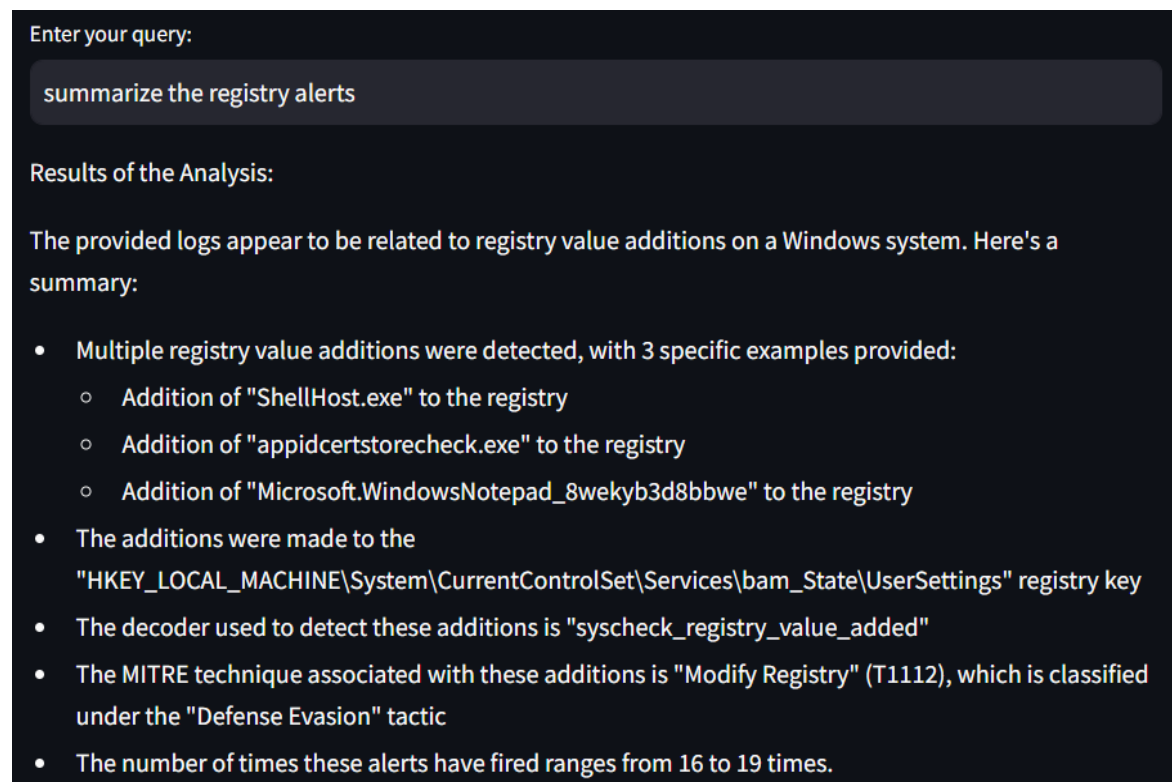
There were 3 login attempts from the IP address 10.10.10.67, all with the username "badguy". The attempts were made through SSH and were unsuccessful due to the username being invalid. The attempts were made within a short time frame, with the following details:

- 1 attempt was disconnected from the invalid user "badguy" (port 48676)
- 2 attempts were made with the invalid user "badguy" (ports 48680 and 48688)

All attempts were detected as potential "Password Guessing" and "SSH" techniques, categorized under "Credential Access" and "Lateral Movement" tactics.

Figure 4.16: Specific login attempts query and response

In addition, when asked to summarize the Windows Registry events, the Data Agent was able to provide information about the type of registry event that is, addition, deletion or modification and even went ahead to correlate the alerts by indicating whether they are related to the same Registry file or Key Entry. It would have taken a SOC Analyst an average of fifteen minutes to manually go through each of these alerts to gather context but it only took the agent two minutes to summarize this data for the researcher.



Enter your query:

summarize the registry alerts

Results of the Analysis:

The provided logs appear to be related to registry value additions on a Windows system. Here's a summary:

- Multiple registry value additions were detected, with 3 specific examples provided:
  - Addition of "ShellHost.exe" to the registry
  - Addition of "appidcertstorecheck.exe" to the registry
  - Addition of "Microsoft.WindowsNotepad\_8wekyb3d8bbwe" to the registry
- The additions were made to the "HKEY\_LOCAL\_MACHINE\System\CurrentControlSet\Services\bam\_State\UserSettings" registry key
- The decoder used to detect these additions is "syscheck\_registry\_value\_added"
- The MITRE technique associated with these additions is "Modify Registry" (T1112), which is classified under the "Defense Evasion" tactic
- The number of times these alerts have fired ranges from 16 to 19 times.

Figure 4.17: Generic Windows registry query and response

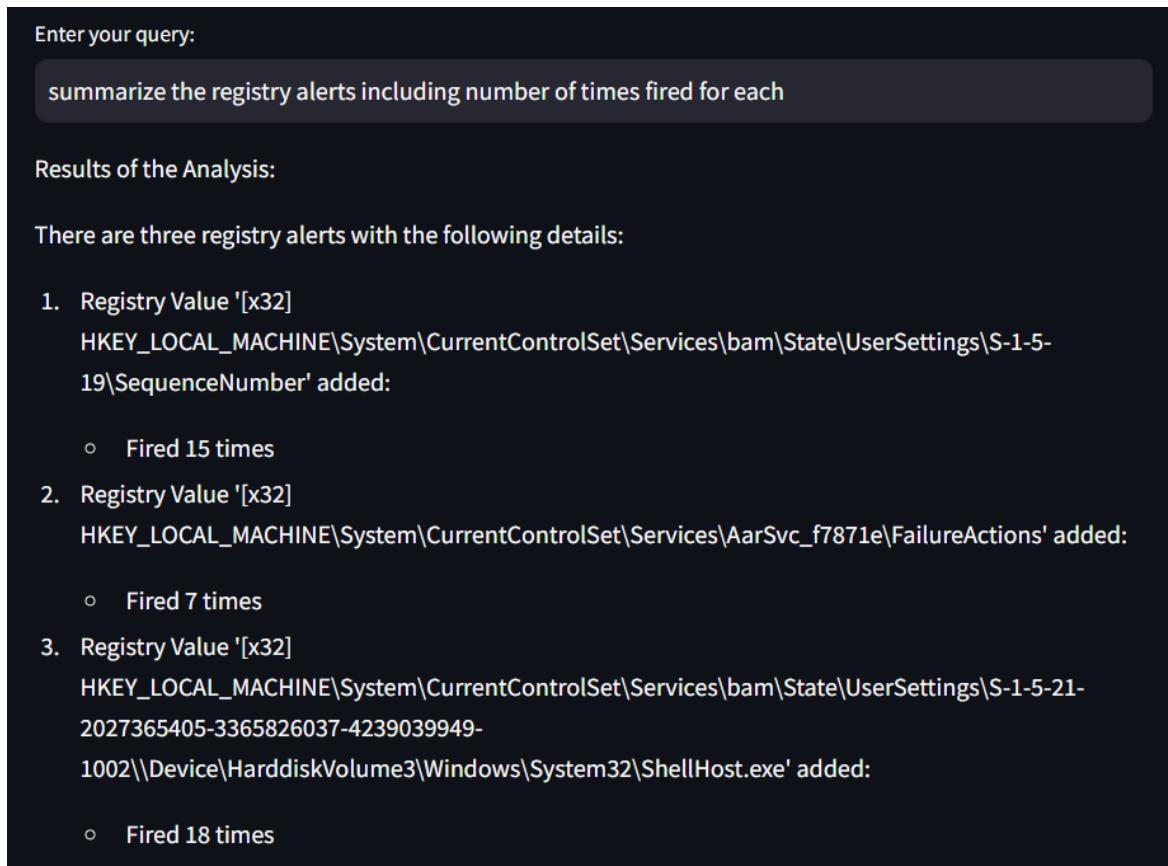


Figure 4.18: Specific Windows registry query and response

Additional tests were carried out to ascertain the functional and non-functional goals of the Data Agent as discussed in the following sections.

#### 4.4.1 Functional Testing

A functional test was performed on the system to ascertain that all modules and components worked as expected, according to the system requirements. This was done based on various use cases on the prototype to confirm if the results give the expected performance as shown in Tables 5.2 to 5.7:

Table 4.2: Admin Login and Logout of the Servers

<b>Test Case</b>	Testing login/logout of Caldera and Wazuh Servers
<b>Description</b>	Admin login/logout of the servers using a username and password
<b>Use Case</b>	Login/logout
<b>Results</b>	Access to the dashboards is granted after successful login
<b>Pass/Fail</b>	Pass

Table 4.3: Deploying Agents on the Servers

<b>Test Case</b>	Testing the deployment of Wazuh and Caldera agents
<b>Description</b>	The researcher successfully deployed the agents to the servers
<b>Use Case</b>	Agent deployment

<b>Results</b>	Agents successfully connected to the servers and visible on the dashboard
<b>Pass/Fail</b>	Pass

Table 4.4: Adversary Attack Emulation

<b>Test Case</b>	Emulating an attack on the target agents
<b>Description</b>	Successfully launched a brute-force attack from Caldera
<b>Use Case</b>	Brute-force attack
<b>Results</b>	Caldera agent successfully sent the alerts to Wazuh
<b>Pass/Fail</b>	Pass

Table 4.5: Generation of Wazuh Alerts

<b>Test Case</b>	Testing to check if alerts are generated
<b>Description</b>	The brute-force attack triggered alerts on the Wazuh SIEM
<b>Use Case</b>	Alert generation
<b>Results</b>	Alerts generated are visible on the Wazuh SIEM
<b>Pass/Fail</b>	Pass

Table 4.6: Extraction and Processing of Alerts

<b>Test Case</b>	Testing the alerts extraction and processing module
<b>Description</b>	HTTP request made to the Wazuh Server API
<b>Use Case</b>	Alerts extraction and processing
<b>Results</b>	HTTP request to Wazuh API returned a response with alerts data in JSON
<b>Pass/Fail</b>	Pass

Table 4.7: Perform Alert Investigation by Querying the Data Agent

<b>Test Case</b>	Testing the Data Agent RAG system connected to the SIEM
<b>Description</b>	A natural language query via the web-based chat interface
<b>Use Case</b>	Log analysis
<b>Results</b>	The Data Agent responded with an accurate output of the query
<b>Pass/Fail</b>	Pass

#### 4.4.2 Non-functional testing

The developed prototype was further tested to check on the non-functional requirements and was found to meet the requirements. These requirements tested were on the availability of the Data Agent and its components where it was confirmed to be up and running at all times. The prototype was developed on a local network using a local LLM for security. The servers were password protected but the web-based chat interface was not password protected. Secure coding was also implemented during basic authentication of the Wazuh Indexer API. It was noted that all the components of the system functioned optimally as expected. The hardware specifications were also performing as expected without glitches.

#### 4.5 System Validation

Due to the nature of the study, and the sensitive nature of SOC environments, the prototype was not launched in a real SOC environment, but on a controlled virtual lab environment for the proof of concept. The Data Agent is designed to only be deployed in the SOC to augment

SOC analysts alerts investigation of potential threats including zero-day attacks. This study assumes that all SOC environments are configured the same with SOC Analysts using single-sign-on that will include the Data Agent.



## Chapter 5: Discussion

### 5.1 Introduction

This study was conducted with the aim of developing a prototype Data Agent that would play a key role in the SOC to augment human SOC Analysts in understanding security alerts and investigate attacks that target their computer systems and networks. The researcher learned that this approach which only adds an important tool to the toolset of the SOC analyst, but also goes a long way in adding efficiency in alert triage by removing context gathering from the SOC analyst's plate letting them concentrate on the tasks of accurately investigating alerts. After the whole system was setup, alerts data generated from the Wazuh SIEM and performing an in-depth literature review related to this study, the expected outcome was achieved. The researcher uses this chapter to provide results and interpretations of the study, which will be used to give conclusions and recommendations in the last chapter.

#### 5.1.1 Review of existing AI tools used in the SOC

The researcher conducted extensive research on available security tools used in the SOC and from the findings and due to financial constraints settled on the Wazuh SIEM which is an open-source tool. Furthermore, most of the existing solutions similar to this prototype are ecosystem-specific, enterprise-grade and closed-source. This prototype goes a long way in helping bridge this gap as it will be open-source and target small and medium-sized businesses. This will allow researchers and these businesses get access to this tool without cost limitations. Further research was conducted confirming that SOCs have been slow in adopting AI-driven security solutions due to security vulnerabilities associated with AI tools such as prompt injection attacks.

#### 5.1.2 Prototype Data Agent for the SOC

The implementation and testing of this prototype was to be done within a real SOC environment as was initially planned. However, due to challenges such as the sensitive nature of the SOC environment with possible cybersecurity risks, the study opted for an implementation and testing was done on the controlled virtual environment. The researcher created virtual machines on the VMWare workstation Type 2 hypervisor. These machines are what were installed and configured to act as the different components of the prototype Data Agent. This made the whole system portable, and easily accessible from wherever the researcher was, without having to always seek permission to access a real SOC environment. The prototype was tested by the number of alerts it was able to generate and process and not forgetting the analysis that it was able to perform on the alerts data. Browser based platforms such as draw.io was used to design

the Data Flow Diagrams, flow charts and sequence diagrams. This prototype data agent is meant to be deployed in a SOC and integrated with other security tools apart from just the SIEM to fully unlock its capabilities.

### 5.1.3 Validation of the effectiveness of the prototype Data Agent

Similar to methods used by Kersten et al. (2023) and Rosso et al. (2020), a tailored alert dataset was extracted from the Wazuh Indexer API, processed and used to query the Data Agent. The dataset contained 50 alerts generated by the SIEM using logs from the Windows and Linux operating system endpoints. Of the alerts, twenty-two were labeled as interesting – those considered to be actual threats or incidences, while the rest were generic events with little or no security significance. The alerts deemed as interesting were generated using adversary emulation techniques performed by the CALDERA adversary emulation platform. The twenty-eight alerts deemed as not interesting were generated from normal system operations.

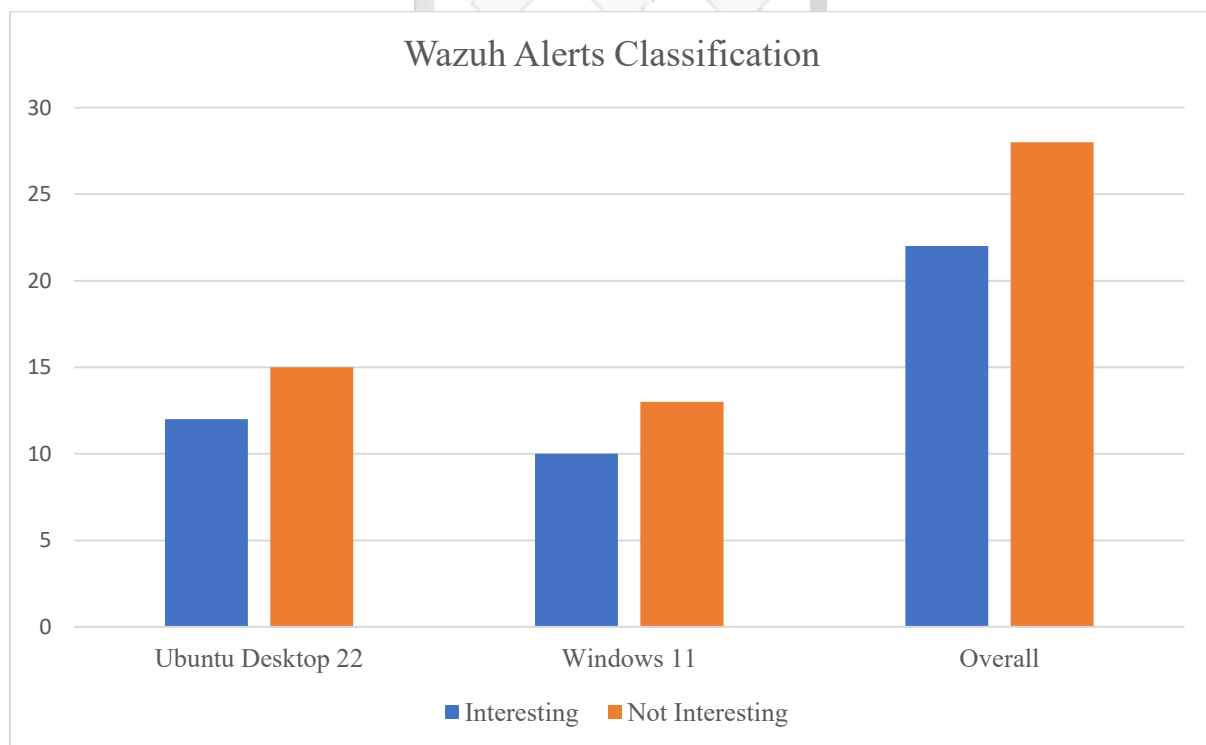


Figure 5.1: Overall Alerts Classification and Endpoints

#### 5.1.3.1 Windows Alerts

The not interesting Windows alerts includes normal Wazuh agent activities, process activities by genuine Microsoft binary processes, and normal system activities. The interesting alerts triggered by CALDERA included process activities by suspicious binaries, Windows registry key modifications and deletions, user account activities involving the Administrator and Guest

accounts (which should always be disabled in Windows endpoints), and clearing of audit logs, which is an activity performed by adversaries to hide their footprint.

Alert ID	Description	Label
1716460351.2348	New Wazuh agent connected	Not-interesting
1716460446.1542036	The start type of the Background Intelligent Transfer Service service was changed from auto start to demand start.	Not-interesting
1716460457.1543868	Windows logon success (eventID 4624) on host DESKTOP-I1PBMQ1.	Not-interesting
1716461131.1560198	Wazuh agent started.	Not-interesting
1716461133.1560539	Discovery activity executed	Not-interesting
1716461783.1683166	Software protection service scheduled successfully.	Not-interesting
1716461784.1684765	Process loaded taskschd.dll module. May be used to create delayed malware execution	Not-interesting
1716465387.1851263	Application Compatibility Database launched	Not-interesting
1716484801.2290155	WSearch was unavailable to handle a notification event.	Not-interesting
1716484801.2291659	Windows User Logoff.	Not-interesting
1716553887.166930	Printer spooler service loaded a dll file. Possible PrintNightmare exploit: CVE-2021-34527	Not-interesting
1716553886.161133	Sysmon - Suspicious Process - svchost.exe	Not-interesting
1716553907.185312	The Windows search service started.	Not-interesting
1716579663.1598575	Windows logon success.	Not-interesting
1716579671.1615206	Explorer process was accessed by powershell.exe, possible process injection	Not-interesting
1716579897.1687901	Windows command prompt started by an abnormal process	Interesting
1716579897.1705930	User account enabled or created.	Interesting
1716580030.2239372	Scripting interpreter spawned a new process	Interesting
1716580845.2470765	Suspicious Windows cmd shell execution - "wmic useraccount get /ALL /format:csv"	Interesting
1716581007.2585021	Value added to registry key has Base64-like pattern	Interesting

Figure 5.2: Labelled Windows Alerts (Oniagbi, 2024)

### 5.1.3.2 Linux Alerts

The not interesting Ubuntu Linux alerts included authentication success and failure alerts usually grouped as PAM Login activities. The not interesting alerts included creating privileged login sessions using by a valid user account, normal Wazuh agent activities such as Agent starting, a netstat port change alert, and generic log and Debian package activities. The interesting alerts included user and group modifications, root account exploitation, and brute-force attacks on existing user accounts.

Alert ID	Description	Label
1716123945.13961	sshd: authentication success.	Not-interesting
1716123945.14470	PAM: Login session opened.	Not-interesting
1716124422.24818	Successful sudo to ROOT executed.	Not-interesting
1716124706.25744	PAM: Login session closed.	Not-interesting
1716126525.33374	sshd: Timeout while logging in.	Not-interesting
1716125093.30331	PAM: Login session opened.	Not-interesting
1716128884.33634	Log file rotated.	Not-interesting
1716134605.0	Wazuh agent started.	Not-interesting
1716134607.321	Listened ports status (netstat) changed (new port opened or closed).	Not-interesting
1716134616.13709	Wazuh server started.	Not-interesting
1716154961.54191	Dpkg (Debian Package) half configured.	Not-interesting
1716327886.22797	New group added to the system.	Interesting
1716327886.22212	New user added to the system.	Interesting
1716327954.24067	Information from the user was changed.	Interesting
1716329950.41028	Group (or user) deleted from the system.	Interesting
1716330807.42863	Integrity checksum changed.	Interesting
1716407215.30217	Root's crontab entry changed.	Interesting
1716407561.34643	Successful sudo to ROOT executed.	Interesting
1716407661.46394	PAM: User changed password.	Interesting
1716410397.125073	Group (or user) deleted from the system.	Interesting
1716410399.125547	Group (or user) deleted from the system.	Interesting
1716411003.145038	PAM: Multiple failed logins in a small period of time.	Interesting
1716411005.153348	sshd: brute force trying to get access to the system. Authentication failed.	Interesting

Figure 5.3: Labelled Linux Alerts (Oniagbi, 2024)

### 5.1.3.3 LLM model and prompts

Due to cost constraints, most LLM models accessed through Groq and OpenAI API exceeded token request limits for specific API requests. Therefore, this research settled on using a local LLM model accessed through Ollama – a platform that allows for downloading and running LLMs locally on a computer. The model used in this research is Meta Llama 3.2 that allows for local API calling. The prompts that were used adopted the optimization techniques recommended by Zhao et al. (2023) and Hadi et al. (2023). The prompts include positive reinforcement with human feedback and a clear description of the required tasks. The description of the task also contains the required outputs expected of the Data agent. The prompts instruct the Data agent to access the alerts extracted from the Wazuh Indexer API and processed by the RAG engine, evaluate them, and return feedback based on their evaluation. All this was done using Python, a high-level general-purpose programming language.

Therefore, the results indicate that the prototype Data Agent was able to successfully extract the alerts from the Wazuh Server and respond to basic queries performed by the researcher through the web-based chat interface.

## **5.2 Conclusion**

The researcher learnt that this approach which not only adds an important tool to the toolset of the SOC analyst, but also goes a long way in adding efficiency in alert triage by removing context gathering from the SOC analyst's plate letting them concentrate on the tasks of accurately investigating alerts. After the whole system was setup, alerts data generated from the Wazuh SIEM and performing an in-depth literature review related to this study, the expected outcome was achieved. While it is not uncommon in the cybersecurity sector to find implementations of AI-powered security tools in the SOC, this research emphasizes the idea of removing barriers to increased adoption, considering they have been around for a while now. According to Agnaou et al, (2017), it is evident that due to the fact that there is exponential increase in ICT infrastructure such as digital technologies, networks and other systems becoming even more vulnerable as this makes them exposed to more attacks, resulting to damages such as widespread breach to confidentiality, integrity and availability, this research proves an effective solution that gives insight into analysis and investigation of attacks.

## **5.3 Study Limitations**

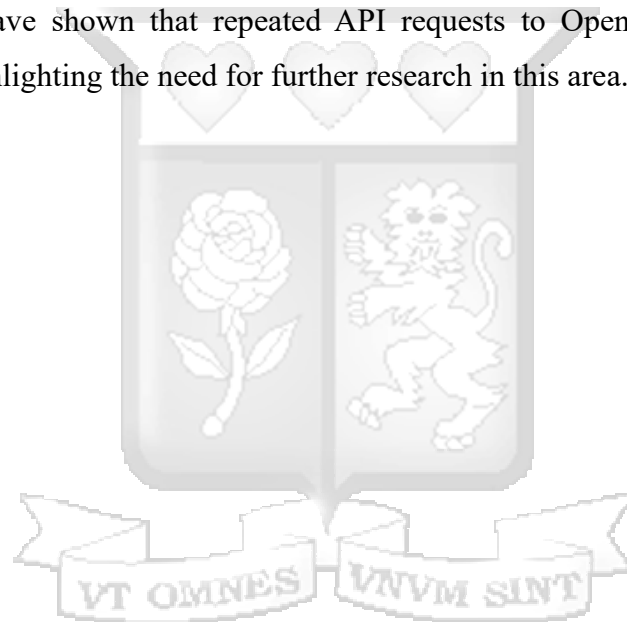
One of the primary constraints of this study is the limited size of the alert dataset utilized, which is a common challenge in the field of threat detection and investigation. Although the generated dataset represented a reasonable range of MITRE ATT&CK framework scenarios, it fell short of the average number of daily alerts generated by organizations. To mitigate this issue, a custom dataset was created, but it comprised only 50 alerts, a relatively small sample size compared to other studies, such as Kersten et al.'s (2023) work, which tested 200 alerts with human Tier 1 analysts. This highlights the need for more comprehensive and representative alert datasets that reflect real-world attack scenarios.

Another significant limitation of this research is the reliance on general-purpose large language models (LLMs), which were not specifically trained for the task at hand. While these models possess emergent capabilities due to their extensive training, they lack domain-specific expertise. For instance, custom LLMs trained on medical datasets have shown improved performance in that field (Shah et al., 2023). However, training custom models incurs

significant computational costs, although it may be more cost-effective in the long run than relying on third-party APIs.

Furthermore, this study did not explore the impact of prompt engineering on LLM performance in alert triage. The formulation of prompts can significantly influence LLM performance, as discussed in Chapter 2. This suggests that optimizing prompts could enhance LLM performance, but a thorough analysis was not conducted to determine the effect of different prompt variations on different models.

Lastly, this research did not investigate the consistency of LLM responses when running the same tests repeatedly. SOC alerts often involve repetitive scenarios, making it essential to test whether LLM agents produce consistent results regardless of the number of times a request is sent. Observations have shown that repeated API requests to OpenAI's GPT-4 can yield different answers highlighting the need for further research in this area.



## Chapter 6: Conclusion and Recommendations

This chapter discusses the conclusion of the study and recommendations for future work based on the limitations highlighted in the previous chapter.

### 6.1 Conclusions

The general objective of this study was to design, develop and test a prototype Data Agent to augment human SOC analysts' investigation process in the SOC. This is important to enable the SOC to enrich their existing cybersecurity toolset and boost their efficiency. From the review of literature, it was evident that SOC's suffer adversely from various challenges such as alert fatigue, high employee turnover, many standalone cybersecurity tools, budget constraints that prevent acquisition of the right cybersecurity solutions among others.

Results got from this study, indicate that this type of solution is not widely adopted due to issues like perceived cybersecurity risks that are still associated with AI-driven solutions in general.

This study adds to the expanding body of research on LLM agents by utilizing them in a unique manner. Additionally, it contributes by developing and evaluating a system that uses LLM agents to carry out the SOC tier 1 analyst investigation process.

### 6.2 Recommendations for future research

This section outlines potential avenues for future research building upon this study. To advance the field, future studies should prioritize the creation and collection of a comprehensive dataset comprising a large number of alerts with diverse signatures, reflective of real-world attack scenarios. Such a dataset would facilitate the benchmarking of LLM performance in alert triage and provide a valuable resource for future research.

Another promising direction for future research is the development and fine-tuning of custom LLM models specifically designed for the alert triage process. As demonstrated in previous studies, custom models can outperform general-purpose models when trained on task-specific datasets (Shah et al. 2023). Therefore, creating a custom SOC Tier 1 analyst model would involve using a training set consisting of alerts, and subsequently evaluating its performance compared to general-purpose models.

Additionally, future research should investigate the impact of prompt engineering on LLM performance in alert triage. It is essential to quantify the effects of different prompts on LLM decision-making to optimize their performance.

Lastly, future studies should examine the reliability of LLM agents in the SOC Tier 1 analyst triage process by assessing the consistency of their decisions when presented with repeated tests on the same alerts. This research would provide valuable insights into the trustworthiness of LLM agents in real-world security operations.



## References

- Adamopoulou, E., & Moussiades, L. (2020). An overview of chatbot technology. In *Artificial Intelligence Applications and Innovations* (pp. 373-383). Springer.
- Anderson, T., & Shattuck, J. (2012). Design-Based Research: A Decade of Progress in Education Research? *Educational Researcher*, 41(1), 16-25.
- Barab, S., & Squire, K. (2004). *Design-Based Research: Putting a Stake in the Ground*. *The Journal of the Learning Sciences*, 13(1), 1-14.
- Barr, J. (2016). AWS in 2016: More Services, More Features. Amazon Web Services Blog.
- Beck, K., Beedle, M., Bennekum, A. van, et al. (2001). Manifesto for Agile Software Development.
- Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly Media.
- Bocklisch, T., Faulkner, J., Pawlowski, N., & Nichol, A. (2017). Rasa: OpenSource Language Understanding and Dialogue Management. arXiv:1712.05181.
- Boo, V. (2020). *Building Chatbots with Python: Using Natural Language Processing and Machine Learning*. Apress.
- Brandtzaeg, P. B., & Følstad, A. (2017). Why people use chatbots. In *International Conference on Internet Science* (pp. 377-392). Springer.
- Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology*, 3(2), 77-101.
- Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P. & Amodei, D. (2020). Language Models are Few-Shot Learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems (NeurIPS 2020)*.
- Campbell, D. & Milmo, D. (2024, June 14). London hospitals cancel nearly 1,600 operations and appointments in one week due to hack. *The Guardian: International Edition*.  
<https://www.theguardian.com/technology/article/2024/jun/14/london-hospitals-cancelled-nearly-1600-operations-and-appointments-in-one-week-due-to-hack>
- Carnegie, A., & Dhirendra, S. (2019). *Hands-On Chatbot Development with Alexa Skills and Amazon Lex*. Packt Publishing.

- Chung, M., & Iorga, M. (2019). Enhancing Customer Experience with AI at H&M Group. H&M Group.
- Clancey, W. J. (1987). Knowledge-based agenting: The GUIDON program. MIT Press.
- Clark, K., Luong, M. T., Lewis, M., & Zettlemoyer, L. (2020). ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. arXiv preprint arXiv:2003.10555.
- Cohn, M. (2004). User Stories Applied: For Agile Software Development. Addison-Wesley Professional.
- Creswell, J. W., & Creswell, J. D. (2017). *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches (4th ed.)*. Sage Publications.
- Creswell, J.W. (2014). *A Concise Introduction to Mixed Methods Research*. Sage publications.
- Cronbach, L. J., & Meehl, P. E. (1955). Construct validity in psychological tests. *Psychological Bulletin*, 52(4), 281-302.
- Demoret, M. & Allen, R. (2024, March). Cybersecurity Developer Day [Video]. NVIDIA. <https://www.nvidia.com/en-us/on-demand/session/gtc24-se62821/>
- Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv preprint arXiv:1810.04805.
- Duvall, P. M., Matyas, S., & Glover, A. (2007). *Continuous Integration: Improving Software Quality and Reducing Risk*. Addison-Wesley Professional.
- Falmagne, J. C., Cosyn, E., Doignon, J. P., & Thiéry, N. (2006). The assessment of knowledge, in theory and in practice. In *Formal Concept Analysis* (pp. 61-79). Springer, Berlin, Heidelberg.
- Fitzgerald, B., & Stol, K.-J. (2017). *Continuous Software Engineering: A Roadmap and Agenda*. *Journal of Systems and Software*, 123, 176-189.
- Fowler Jr, F. J. (2013). *Survey Research Methods*. Sage publications.
- Garg, S., Kaur, K., Kumar, N., Kaddoum, G., Zomaya, A. Y., & Ranjan, R. (2019). A Hybrid Deep Learning-Based Model for Anomaly Detection in Cloud Datacenter Networks," in *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 924-935.
- Graesser, A. C., VanLehn, K., Rose, C. P., Jordan, P. W., & Harter, D. (2004). Intelligent agenting systems with conversational dialogue. *AI Magazine*, 22(4), 39-51.

- Gupta, N., & Saini, R. (2018). *Practical Bot Development: Designing and Building Bots with Node.js and Microsoft Bot Framework*. Apress.
- Heale, R., & Twycross, A. (2015). Validity and reliability in quantitative studies. *Evidence-Based Nursing*, 18(3), 66-67.
- Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, 28(1), 75-105.
- Highsmith, J. (2002). *Agile Software Development Ecosystems*. Addison-Wesley.
- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735-1780.
- Holtzblatt, K., & Beyer, H. (2017). *Contextual Design: Design for Life*. Morgan Kaufmann.
- Huang, S., Yang, K., Qi, S., & Wang, R. (2024). When Large Language Model Meets Optimization. South China University of Technology. <https://arxiv.org/pdf/2405.10098v1>
- Huang, Y., Zhang, J., & Li, Z. (2019). *Conversational AI for Security Operations Centers*. *Journal of Intelligent Information Systems*, 53(2), 257-275.
- Hussain, A., Wenbi, R., da Silva, A. L., Nadher, M., & Mudhish, M. (2019). *Health and wellness chatbots: A technology acceptance model analysis*. *Journal of Technology in Human Services*, 37(4), 282-293.
- James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An Introduction to Statistical Learning*. Springer.
- Jouhki, J., Jauho, M., Penttinen, M., & Isotalo, J. (2020). Chatbot ethics: An overlooked necessity. In *International Conference on Human-Computer Interaction* (pp. 202-210). Springer.
- Joulin, A., Mikolov, T., Grave, E., Bojanowski, P., Mikolov, P., & Ruder, S. (2017). Bag of Tricks for Efficient Text Classification. arXiv preprint arXiv:1607.01759.
- Klopfenstein, L. C., Delpriori, S., Malatini, S., & Bogliolo, A. (2017). The rise of bots: A survey of conversational interfaces, patterns, and paradigms. In *Proceedings of the Conference on Designing Interactive Systems*, 555-565.
- Knox, W.B. and Stone, P., 2009. Interactively shaping agents via human reinforcement: The TAMER framework. In *Proceedings of the fifth international conference on Knowledge capture* (pp. 9-16).

- Krueger, R. A., & Casey, M. A. (2015). *Focus Groups: A Practical Guide for Applied Research*. Sage publications.
- Kumar, V. (2019). *Building Chatbots with Google Dialogflow*. Packt Publishing.
- Lafferty, J., McCallum, A., & Pereira, F. C. N. (2001). Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*.
- Lazar, J., Feng, J. H., & Hochheiser, H. (2017). *Research Methods in Human-Computer Interaction*. Morgan Kaufmann.
- Li, J., Monroe, W., Ritter, A., Jurafsky, D., Galley, M., & Gao, J. (2016). Deep Reinforcement Learning for Dialogue Generation. arXiv preprint arXiv:1606.01541.
- Li, L., Chu, W., Langford, J., & Schapire, R. E. (2010). A Contextual-Bandit Approach to Personalized News Article Recommendation. *Proceedings of the 19th International Conference on World Wide Web*.
- Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., ... & Levy, O. (2019). RoBERTa: A Robustly Optimized BERT Pretraining Approach. arXiv preprint arXiv:1907.11692.
- Lopez, G., & Quesada, L. (2020). *How Voice Technology Will Influence Marketing Strategies*. *Journal of Marketing Trends*.
- McTear, M., Callejas, Z., & Griol, D. (2016). *The Conversational Interface: Talking to Smart Devices*. Springer.
- Merriam, S.B., & Tisdell, E.J. (2015). *Qualitative Research: A Guide to Design and Implementation*. Jossey-Bass.
- Miao, Y., Yu, L., & Blunsom, P. (2017). Neural Variational Inference for Text Processing. In *Proceedings of the 34th International Conference on Machine Learning*.
- Milmo, D. (2024, May 17). UK engineering firm Arup falls victim to £20m deepfake scam. *The Guardian*: International Edition.  
<https://www.theguardian.com/technology/article/2024/may/17/uk-engineering-arup-deepfake-scam-hong-kong-ai-video>
- Miner, A. S., Laranjo, L., & Kocaballi, A. B. (2020). Chatbots in the fight against the COVID-19 pandemic. *npj Digital Medicine*, 3(1), 1-4.

- Morgan, D.L. (2007). *Paradigms lost and pragmatism regained: Methodological implications of combining qualitative and quantitative methods*. *Journal of Mixed Methods Research*, 1(1), 48-76.
- Morse, J.M., Barrett, M., Mayan, M., Olson, K., & Spiers, J. (2002). *Verification strategies for establishing reliability and validity in qualitative research*. *International Journal of Qualitative Methods*, 1(2), 13-22.
- Norman, D. A., & Draper, S. W. (Eds.). (1986). *User Centered System Design: New Perspectives on Human-Computer Interaction*. Lawrence Erlbaum Associates.
- Oates, B. J. (2006). *Researching Information Systems and Computing*. Sage.
- Peppers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). *A design science research methodology for information systems research*. *Journal of Management Information Systems*, 24(3), 45-77.
- Preece, J., Rogers, Y., & Sharp, H. (2015). *Interaction Design: Beyond Human-Computer Interaction (4th ed.)*. Wiley.
- Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language Models are Unsupervised Multitask Learners. *OpenAI Blog*, 1(8), 9-24.
- Radziwill, N., & Benton, M. C. (2020). Evaluating Quality of Chatbots and Intelligent Conversational Agents. *arXiv preprint arXiv:2004.04574*.
- Resnik, D. B. (2011). What is ethics in research & why is it important? National Institute of Environmental Health Sciences.
- Richardson, B. (2024, March). How to Apply Generative AI to Improve Cybersecurity [Video]. NVIDIA. <https://www.nvidia.com/en-us/on-demand/session/gtc24-s62173/>
- Rosenberg, D., & Stephens, M. (2007). *Use Case Driven Object Modeling with UML: Theory and Practice*. Apress.
- Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2019). DistilBERT: Distilled Version of BERT for Faster and Lighter Natural Language Processing. *arXiv preprint arXiv:1910.01108*.
- Saunders, M.N.K., Lewis, P. and Thornhill, A. (2009). *Research Methods for Business Students*. London: Pearson Education
- Schwaber, K., & Beedle, M. (2002). *Agile Software Development with Scrum*. Prentice Hall.

- Shadish, W. R., Cook, T. D., & Campbell, D. T. (2002). *Experimental and Quasi-Experimental Designs for Generalized Causal Inference*. Houghton Mifflin.
- Shenton, A.K. (2004). Strategies for ensuring trustworthiness in qualitative research projects. *Education for Information*, 22(2), 63-75.
- Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. *Advances in Neural Information Processing Systems*.
- Tatam, M., Shanmugam, B., Azam, S. & Kannoopatti, K. (2021). A review of threat modelling approaches for APT-style attacks. Vol. 1, Issue 1.
- Tavakol, M., & Dennick, R. (2011). *Making Sense of Cronbach's Alpha*. *International Journal of Medical Education*, 2, 53-55.
- Teddlie, C., & Tashakkori, A. (2009). *Foundations of Mixed Methods Research: Integrating Quantitative and Qualitative Approaches in the Social and Behavioral Sciences*. Sage.
- Titterington, A. (2024, February 20). Ransomware: The most high-profile attacks of 2023. *Kaspersky daily*. <https://www.kaspersky.com/blog/ransowmare-attacks-in-2023/50634/>
- Trochim, W. M. K. (2006). *The Research Methods Knowledge Base*. Atomic Dog Publishing.
- VanLehn, K. (2011). The relative effectiveness of human agenting, intelligent agenting systems, and other agenting systems. *Educational Psychologist*, 46(4), 197-221.
- Vasan, A. (2018). *Designing Bots: Creating Conversational Experiences*. O'Reilly Media.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., & Polosukhin, I. (2017). Attention is All You Need. arXiv preprint arXiv:1706.03762.
- Wang, F., & Hannafin, M. J. (2005). Design-based research and technology-enhanced learning environments. *Educational Technology Research and Development*, 53(4), 5-23.
- Watson, M. R., Shirazi, N., Marnierides, A. K., Mauthe, A. & Hutchison, D. (2016). Malware Detection in Cloud Computing Infrastructures. *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 2, pp. 192-205.
- Young, T., Hazarika, D., Poria, S., & Cambria, E. (2018). Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine*, 13(3), 55-75.

Zhang, Y., Sun, S., Galley, M., Chen, Y.-C., Brockett, C., Gao, X., ... & Dolan, B. (2018). Dialogpt: Large-scale Generative Pre-training for Conversational Response Generation. arXiv preprint arXiv:1911.00536.



# Appendices

## Appendix A: Similarity Report

171796 - Owino Moses Odhiambo\_IT Thesis\_Final.pdf

ORIGINALITY REPORT

<b>22%</b> SIMILARITY INDEX	<b>21%</b> INTERNET SOURCES	<b>13%</b> PUBLICATIONS	<b>12%</b> STUDENT PAPERS
--------------------------------	--------------------------------	----------------------------	------------------------------

PRIMARY SOURCES

<b>1</b>	<a href="http://su-plus.strathmore.edu">su-plus.strathmore.edu</a> Internet Source	<b>6%</b>
<b>2</b>	<a href="http://www.utupub.fi">www.utupub.fi</a> Internet Source	<b>2%</b>
<b>3</b>	Submitted to Strathmore University Student Paper	<b>1%</b>
<b>4</b>	Fahim Sufi. "An innovative GPT-based open-source intelligence using historical cyber incident reports", Natural Language Processing Journal, 2024 Publication	<b>1%</b>
<b>5</b>	<a href="http://www.truefoundry.com">www.truefoundry.com</a> Internet Source	<b>1%</b>
<b>6</b>	<a href="http://www.researchgate.net">www.researchgate.net</a> Internet Source	<b>&lt;1%</b>
<b>7</b>	<a href="http://developer.nvidia.com">developer.nvidia.com</a> Internet Source	<b>&lt;1%</b>
<b>8</b>	<a href="http://aiforsocialgood.ca">aiforsocialgood.ca</a> Internet Source	<b>&lt;1%</b>
<b>9</b>	<a href="http://www.mdpi.com">www.mdpi.com</a> Internet Source	<b>&lt;1%</b>
<b>10</b>	<a href="http://kt.ijs.si">kt.ijs.si</a> Internet Source	<b>&lt;1%</b>
<b>11</b>	<a href="http://mlmodels.dev">mlmodels.dev</a> Internet Source	<b>&lt;1%</b>

12	<a href="https://researcharchive.vuw.ac.nz">researcharchive.vuw.ac.nz</a> Internet Source	<1 %
13	<a href="https://ger.mercy.edu">ger.mercy.edu</a> Internet Source	<1 %
14	Shahroz Tariq, Mohan Baruwal Chhetri, Surya Nepal, Cecile Paris. "Alert Fatigue in Security Operations Centres: Research Challenges and Opportunities", ACM Computing Surveys, 2025 Publication	<1 %
15	Submitted to University of Hertfordshire Student Paper	<1 %
16	<a href="https://www.machinelearninghelp.org">www.machinelearninghelp.org</a> Internet Source	<1 %
17	<a href="https://idus.us.es">idus.us.es</a> Internet Source	<1 %
18	<a href="https://wrap.warwick.ac.uk">wrap.warwick.ac.uk</a> Internet Source	<1 %
19	Submitted to University of Hong Kong Student Paper	<1 %
20	<a href="https://biblos.hec.ca">biblos.hec.ca</a> Internet Source	<1 %
21	Song Gao, Yingjie Hu, Wenwen Li. "Handbook of Geospatial Artificial Intelligence", CRC Press, 2023 Publication	<1 %
22	<a href="https://dokumen.pub">dokumen.pub</a> Internet Source	<1 %
23	<a href="https://media.suub.uni-bremen.de">media.suub.uni-bremen.de</a> Internet Source	<1 %
24	Submitted to Roehampton University Student Paper	<1 %



27<sup>th</sup> February 2025

Mr Owino Moses,  
odhiambo.owino@strathmore.edu

Dear Mr Owino,

**RE: A Prototype Data Agent for Threat Modelling in Security Operation Centers (SOCs)**

This is to inform you that SU-ISERC has reviewed and approved your above SU-masters proposal. Your application reference number is SU-ISERC2681/25. The approval period is from 27<sup>th</sup> February 2025 to 26<sup>th</sup> February 2026.

This approval is subject to compliance with the following requirements:

- i. Only approved documents including (informed consents, study instruments, MTA) will be used.
- ii. All changes including (amendments, deviations, and violations) are submitted for review and approval by SU-ISERC.
- iii. Death and life-threatening problems and serious adverse events or unexpected adverse events whether related or unrelated to the study must be reported to SU-ISERC within 72 hours of notification.
- iv. Any changes anticipated or otherwise that may increase the risks or affected safety or welfare of study participants and others or affect the integrity of the research must be reported to SU-ISERC within 72 hours.
- v. Clearance for the export of biological specimens must be obtained from relevant institutions.
- vi. Submission of a request for renewal of approval at least 60 days prior to the expiry of the approval period. Attach a comprehensive progress report to support the renewal.
- vii. Submission of an executive summary report within 90 days of completion of the study to SU-ISERC.

Before commencing your study, you will be expected to obtain a research license from National Commission for Science, Technology, and Innovation (NACOSTI) <https://research-portal.nacosti.go.ke/> and obtain other clearances needed.

Yours sincerely,

Mr Ambrose Rachier,  
Chairperson; SU-ISERC