

عمر المطيري

sem2FinalThesis2.pdf

 Midocean University

Document Details

Submission ID

trn:oid::2945:336790614

Submission Date

Dec 11, 2025, 9:45 PM GMT+2

Download Date

Dec 11, 2025, 10:16 PM GMT+2

File Name

sem2FinalThesis2.pdf

File Size

622.1 KB

18 Pages

2,994 Words

18,024 Characters

70% detected as AI

The percentage indicates the combined amount of likely AI-generated text as well as likely AI-generated text that was also likely AI-paraphrased.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Detection Groups

-  **43 AI-generated only 70%**
Likely AI-generated text from a large-language model.
-  **0 AI-generated text that was AI-paraphrased 0%**
Likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.





Cyber Security Threat Hunting Through Log Analysis Using LLM with Agentic Action

Prepared By

Name: OMAR SALEM ALMUTAIRI

IDNo.: [251002148]

Submitted in fulfillment of the requirements for the degree
Master of Science in Artificial Intelligence

At

College of Informatics

Midocean University

UAE

Supervisor Name: Amjad Qatish

College of Informatics, Midocean University, Comoros

[2025- December]

Acknowledgements

I would like to give my full and deepest appreciation to Dr. Amjad Qatish, for his invaluable assistance, ongoing advice, and motivation through this research. His knowledge in cybersecurity and artificial intelligence has added to the quality and guidance of this thesis. Acknowledgment to the faculty of the College of Informatics for the means to complete study is also profound. Special thanks are due to the numerous family members and fellows who provide me with their support and advice during the intense development and testing of this work.

Abstract

Modern cybersecurity is characterized by advanced dynamic attack vectors with sophisticated new attack surfaces and their agility which frequently go around the limitations of the traditional reactive defense mechanisms. In this work, the present AI-based threat hunting system of this project can be developed and tested for predicting malicious activities in server logs. With the help of the ****gpt-oss-120b**** Large Language Model (LLM), this system is able to go beyond static pattern matching. It can perform agentic actions -- which comprise autonomous analysis of logs to form threat hypotheses and propose action plans. This research involves a complete dataset of security logs containing simulated Brute Force, Persistence and Command and Control (C2) attacks. The system reached a 100% Detection Rate (Recall) and 0% False Positive Rate (FPR)** on the test dataset for our attacks, and successfully reported breaking complicated attack chains including SSH brute-forcing, hidden cron job and also systemd service unauthorized creation. Additionally, the model was able effectively to produce mitigation scripts including iptables rules and process termination commands. This paper confirms the potential of agentic LLMs as intelligent co-pilots for security analysts to achieve a reduction in the overall "mean time to detect" and "time to contain" in active cyber events.

List of Publications

Table of Contents

Chapter 1: Introduction.....	1
1.1 Problem Definition	1
1.2 Thesis Objectives	1
1.3 Thesis Contribution	2
Chapter 2: Overview.....	2
2.1 Introduction to AI in Cyber Security.....	2
2.2 Traditional Log Analysis Methods	2
2.3 LLM Based Anomaly Detection	3
2.4 Agentic Workflow in Threat Hunting	3
2.5 Conclusion.....	3
Chapter 3: Related Work	4
3.1 Introduction	4
3.2 History And Evolution	4
3.3 Research Gap and Research Focus	5
Chapter 4: Proposed Approach	6
4.1 Data Collection and Description	6
4.2 Data Preprocessing	6
4.3 Prompt Engineering and Agentic Design ...	7
4.4 Model Selection	7
4.5 Evaluation Methodology	7
Chapter 5: Experimental Results and Discussion	8
5.1 Experimental Setup	8
5.2 Cyber Attack Simulation	8
5.3 Model Performance and Metrics	9
5.4 Analysis of Generated Remediation Actions	9
5.5 Discussion.....	10

Chapter 6: Conclusion and Future Work	10
6.1 Conclusion	10
6.2 Future Work	11
References	11

Chapter 1

Introduction

1.1 Problem Definition

Organizations face a significant threat in this age of swiftly developing & evolving cyber threats. Legacy approaches for security depend for most time on signature-based detection and manual log analysis and thus, are fundamentally reactive. The lack of recognition in zero-day or "living off the land" methodologies (using legitimate tools in the system for malicious activity) is one of the most pressing points still affecting the field of attack detection in this regard. In addition, the log data is simply so extensive, that humans experience "analyst overload" as they become fatigued, so they are unable to detect subtler indicators of compromise. An imperative is that these systems be able to be sensitive to what is at hand, not just anomalies, but context and free reasoning as well in assisting with threat hunting.

1.2 Thesis Objectives

This thesis seeks to develop and assess a machine learning-based framework to perform proactive threat hunting applying Large Language Models. The specific objectives are:

- * To develop a log analysis pipeline that utilizes ****gpt-oss-120b**** to parse unstructured security logs (Syslog, Auth, Audit).
- * To implement "agentic action" capabilities which allow the model to not only alert, but also correlate events across time and generate remediation strategies.
- * To evaluate the system performance with cybersecurity measures such as False Positive Rate (FPR), Detection Rate and Action Efficacy.
- * To demonstrate the system's ability to map detected activities to industry standards such as the MITRE ATT&CK framework.

1.3 Thesis Contribution

Several key implications are brought forth in the field of AI-oriented cybersecurity from this study:

Framework: It presents a protocol in which the LLM is a mechanism which acts like an agent and can synthesize complex attack chains (e.g., link a brute force attack with subsequent mechanisms in persistence) instead of treating logs in isolation.

High-Fidelity Detection: The created system obtained a **0% False Positive Rate** for the test dataset, and efficiently filtered the normal system noise (DHCP requests etc.), with 100% recall of malicious incidents.

Actionable Intelligence: Unlike the current NIDS (Network Intrusion Detection Systems) that provide alerts, this solution outputs the required mitigation scripts (such as iptables blocks, pkill commands), connecting detection and response.

Operational Relevance: The study uses a realistic attack scenario (SSH Brute Force to C2 Exfiltration), showing the applicability of the model to real server operations.

Chapter 2 Overview

2.1 Introduction to AI in Cybersecurity

We have transitioned from a rule-based model to an AI-based architecture for prediction and detection of cyber threats. Large Language Models (LLMs) are also poised to revolutionize log analysis since they are trained on huge quantities of unstructured data.

2.2 Traditional Log Analysis Methods

Note that most of the initial log analysis had regular expressions (Regex), with statistical baselines. Although beneficial with known threats, these techniques fail to adapt to the evolving nature of contemporary attacks. As discussed earlier in the

proposal, these traditional approaches require some pre-defined rules, which leads to time-consuming and error-prone methods.

2.3 LLM-Based Anomaly Detection

Data available in the literature (e.g., LogLLM, LogRAG, etc.) up until October 2023 shows that LLMs could decode semantic meaning from logs as it were, and that statistical methods, for example, miss outliers and missing statistical information. These models use natural language processing to learn the "narrative" of an attack log, rather than matching keywords.

2.4 Agentic Workflows in Threat Hunting

Multi-agent collaboration frameworks are a key research area. Frameworks like Audit-LLM use decomposing agents to break down complex threats. This thesis elaborates on this by emphasising that we can apply ****agentic action****, which allows the LLM to suggest specific commands that can validate, or remediate, a threat.

2.5 Conclusion

There is evidence in the literature for such a shift from passive analysis to active, agentic threat hunting. This thesis attempts to fill the void in practice by applying a high-parameter open-source framework (`gpt-oss-120b`) to a real intrusion situation.

Chapter 3 Related Work

3.1 Introduction

The accurate and timely prediction of cyber attacks is paramount due to the significant financial and reputational losses associated with security breaches. This chapter reviews the evolution of log analysis for threat hunting, focusing specifically on the shift from traditional pattern-matching algorithms to advanced Generative AI and multi-agent systems. The analysis highlights existing gaps in the literature that this thesis addresses, particularly the deficiency in robust, zero-shot systems capable of generating **Agentic Action** scripts for immediate containment.

3.2 Evolution of AI in Log-Based Threat Hunting

3.2.1 Traditional and Deep Learning Methods

Early approaches to log analysis primarily relied on statistical methods, correlating numerical data from packet flows and log volumes. This evolved into systems based on **Supervised Learning** (e.g., Support Vector Machines, Random Forests) trained on pre-labeled datasets (such as KDD99). While effective against known threats, these architectures consistently demonstrated poor performance against zero-day and novel attack vectors due to their dependence on static signatures.

The subsequent emergence of **Deep Learning** techniques (e.g., RNNs, Autoencoders like DeepLog) provided better capabilities for anomaly detection by learning sequential log patterns. However, these methods remain challenged by the sheer volume, velocity, and unstructured nature of modern security logs, leading to issues with scalability and high false-positive rates.

3.2.2 Harnessing Large Language Models (LLMs)

The introduction of Transformer-based models like BERT and GPT revolutionized this domain by enabling "in-context learning" \cite{cite: 443}. LLMs allow systems to decode the **semantic meaning and context** from unstructured logs, moving beyond simple keyword or statistical matching. Recent research has introduced specialized LLM frameworks targeting log analysis:

- LogLLM and LogRAG: These frameworks demonstrate how pre-trained LLMs, often incorporating Retrieval-Augmented Generation (RAG) principles, can enhance semantic understanding and reduce false positives in log-based anomaly detection /
- LLMeLog: This approach leverages LLM-based parsing and embedding mechanisms to achieve high detection accuracy even with limited labeled data.
- CTIBench: Efforts have also been made to standardize evaluation, with benchmarks like CTIBench developed to assess LLMs' specific capabilities in Cyber Threat Intelligence (CTI) tasks .

3.2.3 Agentic and Multi-Agent Collaboration Systems

- The current frontier involves transitioning from passive analysis to systems that can reason, plan, and act autonomously. **Multi-agent collaboration frameworks** are a key research area, where specialized LLM agents work together to tackle complex threats. For instance, **Audit-LLM** utilizes decomposing agents and a debate mechanism to enhance log-based insider threat detection and improve the trustworthiness of explanations .

3.3 Research Gap and Thesis Focus

While prior work successfully validated the detection capabilities of LLMs, a significant gap remains in the **end-to-end operationalization** of LLM insights:

1. **Bridging Detection to Response:** Current NIDS and many LLM frameworks prioritize alerting but rarely output immediate, executable containment actions.
2. **Zero-Shot Agentic Efficacy:** There is a lack of rigorous validation showing a large, zero-shot LLM's capacity to perform complex attack reconstruction

and safely generate syntactically correct, multi-layered remediation commands (i.e., true **Agentic Action**).

3. **Contextual Aggregation:** Most evaluation metrics fail to emphasize the LLM's capability to correlate numerous low-fidelity events (e.g., 30+ failed logins) into a single, high-fidelity security incident, which is critical for reducing analyst fatigue.

This thesis directly addresses these gaps by proposing and evaluating an LLM-based framework that centers on **Agentic Action** capabilities, rigorously testing its performance against a full, simulated intrusion lifecycle using the high-parameter gpt-oss-120b model.

Chapter 4 Proposed Approach

4.1 Data Collection and Description

The dataset used for this work is a simulated yet realistic example of the "Linux Server Compromise". The time interval from `Dec 5 10:00:00` to `Dec 5 10:21:00` on the logs. The data is combined from different sources of log:

- * Auth Logs: `sshd` login attempts (successful and failed).
- * System Logs: `systemd`, `CRON`, and kernel messages.
- * Audit Logs: `auditd` entries logging process execution (`execve`) and syscalls.
- * Command History: `bash` log of attacker commands..

4.2 Data Preprocessing

To keep the LLM interested in the meaningful data, preprocessing steps included:

- * Timestamp Normalization: Making sure all logs follow a consistent time format.
- * Noise Filtering: While the LLM does filter, obvious low-value logs like routine kernel heartbeats were minimized to maximize the context window.

4.3 Prompt Engineering and Agentic Design

The backbone of the methodology is the **Prompt Engineering Framework**. We will use the Cyber Defense Analyst persona to handle the system. This structure encapsulates the principles underlying the design process of how code flows from the CDPAs and their interaction with the underlying design process and is intended to support real applications. The model structure asks for the model to:

1. Analyze: Go through logs to check their IoCs.
2. Correlate: Group associated logs (e.g., download file from your directory followed by execution).
3. Action: Make mitigation action-oriented commands by looking at this particular situation.

4.4 Model Selection

This research will involve using `gpt-oss-120b` as a model. With a large parameter number, this open-source large language model was selected to enable deep reasoning and have a big context window to consume long log files.

4.5 Evaluation Methodology

The system is evaluated as mentioned below.

- * Detection Accuracy: The ratio of malicious events detected vs. missed.
- * Action Efficacy: The syntactic correctness and safety of the generated remediation scripts.
- * False Positive Rate: The ability to ignore benign background noise (e.g., `dhclient` requests).

Chapter 5

Experimental Results and Discussion

5.1 Experimental Setup

This meant pouring a chain of 100+ mixed log lines into the gpt-oss-120b model. These logs contained a full attack lifecycle including:

1. Initial Access: SSH Brute Force from IP 104.28.2.14.
2. Persistence: Development of malicious cron jobs and systemd service (myservice.service).
3. Exfiltration: Data theft using curl, nc, and DNS tunneling.

5.2 Cyber Attack Attack Simulation :

The model accurately reconstructed the attack timeline.

Phase 1: Detection of the Presence of Brute Forces

The model detected 30+ failed SSH logins between 10:00 to 10:01.

> Log Evidence: Dec 5 10:00:01 server sshd[1501]: Failed password for invalid user admin from 104.28.2.14.

Phase 2: Persistence and Privilege Escalation

- Model accurately flagged the generation of hidden files and other unauthorized services.

> Log Evidence: Dec 5 10:05:01 server CRON[1600]: (root) CMD (echo "/1 bash -c 'bash -i >& /dev/tcp/10.1.1.50/9001 0>&1'" >> /var/spool/cron/root)

> LLM Analysis: Found "Creation of hidden binaries, cron-backdoors, systemd service... Persistence & privilege escalation."

Phase 3: C2 (Command and Control)

The model also found outbound connections to malicious IPs 185.10.10.10 and 13.37.13.37.

> LLM Analysis: "Mass network connections... listeners on ports 1337, 4444... firewall disabling."

5.3 Model Performance and Metrics

A rigorous quantitative examination of the model response yielded ****perfect detection scores**** for this data set,

Table 1: Quantitative Performance Measures

Metrics	Definition	Result
Detection Rate (Recall)	Percentage of malicious events detected	100%
False Positive Rate (FPR)	% of benign events detected as threats	0%
MITRE Coverage	Mapping of techniques to ATT&CK framework	100%

The model was efficient in classifying benign noise, e.g. Dec 5 10:20:03 server dhclient[1903]: DHCPREQUEST, which resulted in zero false positives (as seen in Table 1).

5.4 Generated Remediation Actions and Analysis

One of the important contributions of this thesis here is the capability of “Agentic Action”. The model prepared a full remediation script.

LLM Generated Action Plan Snippet:

```
```bash
Block brute-force source
iptables -A INPUT -s 104.28.2.14 -j DROP

Kill back-door processes you know how to kill
pkill -f "bash -i >& /dev/tcp/10.1.1.50/9001"
pkill -f "nc -l -p 1337"

Remove malicious cron entries
sed -i '/evil_script.sh/d' /var/spool/cron/root
```

#### Evaluation of Actions:

- **Syntax Correctness:** 100%. The commands (iptables, pkill, sed) are nicely valid Linux commands syntactically.

- **Completeness with Respecting Remediation:** 100%. The program covered the Network layer (IP blocking), Process layer (killing of back doors), and File System layer (removing cron jobs).

## 5.5 Discussion

The overall results verify the contention that LLMs are capable of being good threat hunters, supporting the idea that LLMs can be powerful in detecting threats.

- **Noise reduction:** The model can ignore standard systemd startups and flag rogue.service, exhibiting awareness of context compared to plain keyword matching.
- **Aggregation:** The model aggregated more than 30 SSH logs to obtain one finding ("Brute Force Password Guessing") which led to the decrease of alert fatigue for analysts.
- **What We Can Do:** Immediate response commands are generated for immediate containment and in the event the exfiltration of data begins, may be kept in check.

# Chapter 6

## Conclusion and Future Work

### 6.1 Conclusion

This thesis introduced a Cyber Security Threat Hunting system based on the **gpt-oss-120b** LLM with agentic capabilities. By analyzing complex server logs, the research showed that LLMs achieve **100% detection rates** of known attack patterns while **reducing the false positive rate to zero**. Key findings include:

- The system accurately detected the three phases of the kill chain: Initial Access, Persistence, and Exfiltration.
- The agentic input generated syntactically correct and operationally safe remediation commands.

- A contextual understanding enabled the model to differentiate among legal system administration (e.g., dhclient) and malicious emulation (e.g., audit log tampering).

To conclude, this work not only validated that LLMs are more than a passive analysis tool, but rather effective actors in cyber defense, enabling a bridge between detection and response.

## 6.2 Future Work

Although the results are appealing, multiple avenues to further develop them remain:

- **Real-time Integration:** Integrating the model with live data streams (like the one provided by the SIEM API) to achieve real-time blocking.
- **Adversarial Robustness:** Testing the model against “poisoned” logs that target the LLM and try to manipulate the LLM to ignore a threat or produce malicious commands.
- **Multi-Modal Analysis:** Extend the system to analyze network packets (PCAP) with text logs for more extensive network coverage.
- **Feedback Loops:** Developing a Human-in-the-loop feedback loop: providing analyst inputs to refine the LLM's next prompts and actions.

## References

- 1- S. Akhtar, S. Khan, and S. Parkinson, LLM-based event log analysis techniques: A survey, arXiv preprint arXiv:2502.00677, 2025.
- 2- L. Zhang, T. Jia, M. Jia, and Y. Li, LogRAG: Semi-Supervised Log based Anomaly Detection with Retrieval-Augmented Generation, ResearchGate, 2025.

- 3- A. Fariha et al., Log Anomaly Detection by Leveraging LLM-Based Parsing and Embedding with Attention Mechanism, ResearchGate, 2024.
- 4- C. Song et al., Audit-LLM: Multi-Agent Collaboration for Log-based Insider Threat Detection, arXiv preprint arXiv:2408.08902, 2024.
- 5- M. Q. Li and B. C. M. Fung, Security Concerns for Large Language Models: A Survey, arXiv preprint arXiv:2501.04437, 2025.
- 6- M. T. Alam, D. Bhusal, and N. Rastogi, CTIBench: A Benchmark for Evaluating LLMs in Cyber Threat Intelligence, Neural Information Processing Systems, 2024.