VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI-590018, KARNATAKA

<u>TECHNICAL SEMINAR REPORT</u>

ON

"Revolutionizing Cybersecurity: Implementing Large Language Models as Dynamic SOAR Tools"

<u>Submitted by</u>

Name: SUHAS A C

USN: 1CR21EC217

<u>Under the guidance of</u>

Mr. Madhu G C

Assistant Professor

Department Of Electronics and Communication Engineering

Feb – June 2025

Department Of Electronics and Communication Engineering

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI,

BENGALURU-560037

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify the Technical Seminar Report entitled "**Revolutionizing Cybersecurity: Implementing Large Language Models as Dynamic SOAR Tools**" prepared by **Mr. SUHAS A C**, bearing **USN 1CR21EC217**, a bona fide student of **CMR Institute of Technology, Bengaluru** in partial fulfillment of the requirements for the award of **Bachelor of Engineering in Electronics and Communication Engineering** of the V**isvesvaraya Technological University, Belagavi-590018** during the academic year 2024-25.

This is certified that all the corrections and suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The seminar report has been approved as it satisfies the academic requirements prescribed for the said degree.

----------------------	-----------------------
Signature of Guide	**Signature of HOD**
Mr. Madhu G C	**Dr. Pappa M**
Asst. Professor	**Professor & HoD**
Dept. of ECE, CMRIT	**Dept. of ECE, CMRIT**

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without mentioning the people whose proper guidance and encouragement has served as a beacon and crowned my efforts with success. I take an opportunity to thank all the distinguished personalities for their enormous and precious support and encouragement throughout the duration of this seminar.

I take this opportunity to express my sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing me an opportunity to present my technical seminar.

I have a great pleasure in expressing my deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant encouragement.

I would like to thank **Dr. Pappa M,** HoD, Department of Electronics and Communication Engineering, CMRIT, Bangalore, who shared his opinion and experience through which I received the required information crucial for the seminar.

I consider it a privilege and honor to express my sincere gratitude to my guide **Mr. Madhu G C**, Assistant Professor, Department of Electronics and Communication Engineering, for the valuable guidance throughout the tenure of this review.

I also extend my thanks to the faculties of Electronics and Communication Engineering Department who directly or indirectly encouraged me.

Finally, I would like to thank my parents and friends for all their moral support they have given me during the completion of this work.

SUHAS A C

(1CR21EC217)

ABSTRACT

This research explores the potential of Large Language Models (LLMs), explicitly using ChatGPT Actions as dynamic SOAR tools to address evolving cybersecurity threats. Traditional SOAR systems, though effective, demand significant time and resources for development and maintenance. The study evaluates their ability to autonomously detect, analyze, and respond to threats by integrating LLMs into a controlled environment and simulating various cybersecurity incidents. Findings reveal that LLM-driven SOAR tools reduce development time, enhance response effectiveness, and improve communication clarity. However, challenges such as continuous model updates and staff training were noted. This research provides a framework for implementing LLM-driven SOAR tools, highlighting their transformative potential in cybersecurity operations and suggesting areas for further study.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

1.1. The Growing Need for Enhanced Cybersecurity Automation

In today's digital age, the pace at which cybersecurity threats evolve demands equally dynamic defense mechanisms. Security Orchestration, Automation, and Response (SOAR) systems are crucial in managing these threats by automating complex workflows and responses. Despite their efficacy, the traditional SOAR tools are often resource-intensive, requiring significant time and expertise to develop and maintain effective playbooks. This poses a particular challenge for organizations that may need more resources.

1.2. Integrating Large Language Models into SOAR

This paper explores an innovative approach to addressing these challenges by integrating Large Language Models (LLMs), such as OpenAI's GPT technology, into traditional SOAR workflows. LLMs have shown promise in various applications that require natural language understanding and decision-making capabilities. By leveraging these models, SOAR systems can automate routine responses and generate real-time adaptive, intelligent security measures.

1.3. Research Context and Objectives

This study is positioned at the intersection of artificial intelligence and cybersecurity, a cutting-edge area of research that seeks to leverage the latest advancements in AI to bolster cyber defenses. The study aims to assess the efficacy of LLMs in reducing the labor and time traditionally required to develop and update SOAR playbooks. Additionally, it evaluates the impact of these models on the effectiveness of automated responses, with the goal of providing a detailed analysis that could guide cybersecurity professionals and organizations in enhancing their security operations through innovative AI integrations.

Chapter 2

RESEARCH METHOD

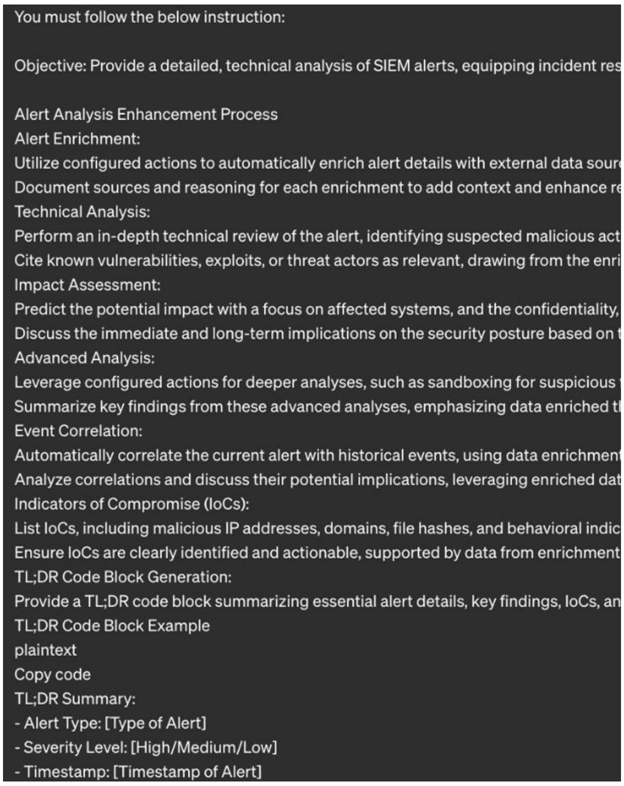
The research methodology encompasses the integration of LLMs into a controlled environment, the simulation of various cybersecurity incidents, the systematic monitoring of LLM responses, and an in-depth analysis of their performance compared to traditional SOAR systems. The objective is to provide a thorough and replicable framework for assessing the feasibility and effectiveness of LLM's in enhancing cybersecurity operations.

2.1. Research Setup

2.1.1. CustomGPT Setup

The first phase of the research involved developing and configuring a CustomGPT through the ChatGPT platform. This step was critical to ensure that the Large Language Model (LLM) was tailored to meet the specific requirements of a dynamic SOAR tool. The CustomGPT setup encompassed several detailed processes:

1. **Prompt Engineering:** A fine-tuned prompt must be developed for the LLM to operate efficiently. This involved iterative testing and tuning the prompt to ensure that the GPT produced precise and accurate results in response to cybersecurity scenarios. The prompts were designed to be comprehensive and detailed, guiding the LLM to perform specific tasks such as threat detection, analysis, and response actions.
2. **Documentation and Training Data:** Extensive documentation and training data were incorporated to enhance the LLM's performance. This included examples of cybersecurity incidents, response protocols, and detailed explanations of various threat types. The documentation served as a reference for the LLM, enabling it to understand and process complex security tasks more effectively.


You must follow the below instruction:

Objective: Provide a detailed, technical analysis of SIEM alerts, equipping incident res

Alert Analysis Enhancement Process
Alert Enrichment:
Utilize configured actions to automatically enrich alert details with external data sour
Document sources and reasoning for each enrichment to add context and enhance re
Technical Analysis:
Perform an in-depth technical review of the alert, identifying suspected malicious act
Cite known vulnerabilities, exploits, or threat actors as relevant, drawing from the enri
Impact Assessment:
Predict the potential impact with a focus on affected systems, and the confidentiality,
Discuss the immediate and long-term implications on the security posture based on t
Advanced Analysis:
Leverage configured actions for deeper analyses, such as sandboxing for suspicious
Summarize key findings from these advanced analyses, emphasizing data enriched t
Event Correlation:
Automatically correlate the current alert with historical events, using data enrichment
Analyze correlations and discuss their potential implications, leveraging enriched dat
Indicators of Compromise (IoCs):
List IoCs, including malicious IP addresses, domains, file hashes, and behavioral indic
Ensure IoCs are clearly identified and actionable, supported by data from enrichment
TL;DR Code Block Generation:
Provide a TL;DR code block summarizing essential alert details, key findings, IoCs, an
TL;DR Code Block Example
plaintext
Copy code
TL;DR Summary:
- Alert Type: [Type of Alert]
- Severity Level: [High/Medium/Low]
- Timestamp: [Timestamp of Alert]

Figure 1: Experiment Methodology Prompt

3. **Configuration of Actions (API Integrations):** One of the most crucial aspects of setting up the CustomGPT was configuring Actions, which involved integrating the LLM with external APIs. These integrations extended the capabilities of the LLM, allowing it to interact with various cybersecurity tools and systems like traditional SOAR platforms. The API integrations enabled the LLM to pull data from threat intelligence feeds, execute automated responses, and update security dashboards.
4. **Validation and Testing:** A validation phase was conducted after the initial setup to ensure the CustomGPT was functioning as intended. This involved running a series of test scenarios to evaluate the accuracy and reliability of the LLM's responses. Feedback from these tests further refined the prompts and configurations, ensuring that the LLM could handle real-world cybersecurity incidents effectively.
5. **Continuous Improvement:** The setup process also included mechanisms for constant improvement. Regular updates were planned to incorporate new threat data, refine response strategies, and enhance the LLM's overall capabilities. This iterative approach ensured that the CustomGPT remained practical and up-to-date with cybersecurity trends and threats.

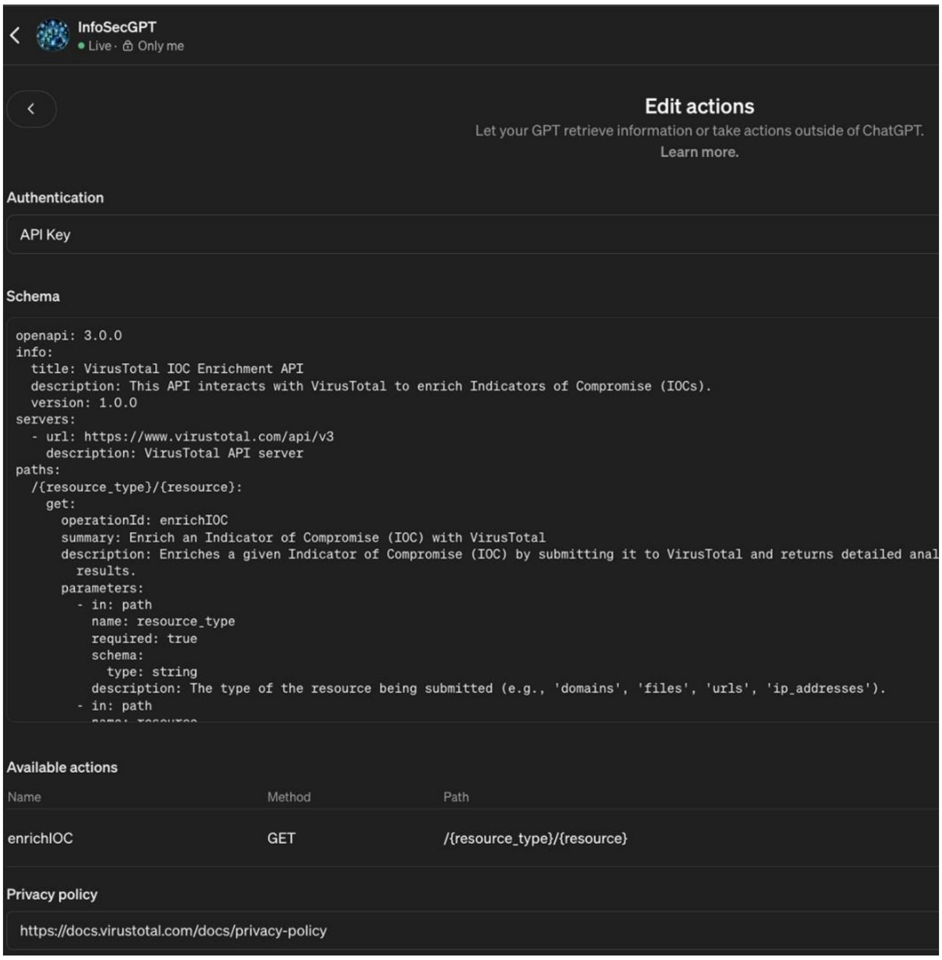


Figure 2: VirusTotal Integration Configuration

By meticulously developing and configuring the CustomGPT, the research aimed to create a robust and dynamic SOAR tool capable of autonomously managing a wide range of cybersecurity tasks. This phase laid the foundation for subsequent testing and evaluation, providing a comprehensive setup that integrated advanced LLM capabilities with practical cybersecurity applications.

2.1.2. Traditional SOAR Setup with Tines

To provide a benchmark for comparison, a traditional SOAR system was set up using Tines, a platform known for its user-friendly and flexible automation capabilities. Tines offers a free-tier option, making it an accessible choice for developing and testing SOAR automation. The following steps outline the setup process for Tines:

1. **Environment Setup:** A Tines account was created, and a dedicated workspace was configured to replicate the SOAR functionalities intended for comparison with the CustomGPT. This included setting up data feeds, security tools, and integrations necessary for incident response and threat management.
2. **Automation Configuration:** Similar to the CustomGPT, various automatons were created within Tines to handle tasks such as threat detection, analysis, and response. These automatons were designed to mirror the capabilities of the LLM-driven SOAR tools, providing a direct comparison of performance and efficiency.
3. **Validation and Testing:** The Tines setup underwent a validation phase where the configured automation was tested against the same scenarios used for the CustomGPT. This ensured that both systems were evaluated under comparable conditions, allowing for an accurate assessment of their respective strengths and weaknesses.
4. **Data Collection and Analysis:** Data from the Tines SOAR system was collected and analysed in parallel with the data from the CustomGPT. Key performance indicators such as response time, accuracy, and reliability were measured to determine the effectiveness of each system in handling cybersecurity incidents.

2.2. Simulation of Cybersecurity Incidents

A range of simulated cybersecurity threats were introduced into the controlled environment to comprehensively evaluate the LLM’s performance. These simulations were carefully designed to cover a broad spectrum of everyday SOAR tasks and included the following scenarios:

1. **Phishing Attacks:** Simulations involving phishing emails required the LLM to validate email headers, extract and analyse links, check for malicious attachments, and generate a concise report detailing the findings.
2. **Malware Attacks:** Scenarios involving ransomware infections tasked the LLM with detecting the threat, isolating affected systems, initiating remediation actions, and communicating the incident details to relevant stakeholders.
3. **Network Intrusions:** Intrusion scenarios involved unauthorized access attempts, where the LLM needed to identify unusual network activity, analyse security logs, and implement containment measures to mitigate the threat.

2.3. LLM Execution and Monitoring

During the simulation phase, the LLM was allowed to autonomously detect, analyse, and respond to the introduced threats. The execution of these tasks was meticulously monitored to ensure a thorough evaluation of the LLM's capabilities. Key focus areas included:

1. **Decision-Making Process:** The LLM's decision-making process was tracked to understand how it prioritized threats, selected response actions, and adapted its strategies based on real-time analysis.
2. **Response Times:** The time the LLM took to detect and respond to each threat was recorded to evaluate its efficiency in handling incidents.
3. **Outcome Effectiveness:** The effectiveness of the LLM's actions was analysed to determine how well it mitigated threats and whether its responses aligned with best cybersecurity practices.

2.4. Data Collection, Evaluation Criteria, and Analysis

Comprehensive data collection was essential to thoroughly evaluate the LLM's performance across different threat scenarios. Critical metrics for data collection included:

1. **Threat Detection and Analysis:** Assessing the effectiveness of the LLM in identifying and analysing cybersecurity threats, including metrics such as detection accuracy, false favourable rates, and false negative rates.
2. **Response Actions:** Evaluating the LLM's ability to determine and execute appropriate response measures, focusing on the success rate of automated actions and their alignment with predefined security protocols.
3. **Accuracy and Reliability:** Comparing the precision of the LLM's actions to expected SOAR outcomes, assessing consistency, reliability, and any deviations from standard practices.
4. **Automation Efficiency:** Measuring the degree of automation achieved and the overall time saved compared to traditional SOAR processes, highlighting potential productivity gains from using LLM-driven SOAR tools.

Data was systematically collected and analysed to ensure a robust and comprehensive assessment of the LLM's capabilities. The collected data was meticulously analysed following the testing phase to evaluate the LLM's effectiveness in executing SOAR functions. The analysis involved comparing the LLM's performance metrics with traditional SOAR systems to highlight differences in efficiency, accuracy, and adaptability. Instances of misidentification, incorrect analysis, or inappropriate responses were identified and analysed, providing insights into areas needing improvement. The degree of automation achieved and the time saved were evaluated, quantifying the benefits of using LLM-driven SOAR tools and their potential to enhance operational efficiency.

2.5. Test Duration and Environment

The experiment was conducted over 50 different security-related events to ensure comprehensive data collection and manageable analysis. This duration was sufficient to observe the LLM's performance across various simulated incidents and gather meaningful insights. The controlled environment replicated real-world conditions as closely as possible, ensuring the LLM had access to all necessary data and network controls.

By following this systematic and robust approach, the research ensured that the study's findings are reliable, applicable, and beneficial to real-world cybersecurity operations. This methodology provides a replicable framework for assessing the potential of LLMs using ChatGPT Actions to function as dynamic SOAR tools, paving the way for more adaptive, efficient, and effective incident response strategies.

Chapter 3

FINDINGS AND DISCUSSION

The findings from this experiment reveal significant advantages of LLM-driven SOAR tools over traditional SOAR systems, particularly in terms of improvisation, communication, and overall effectiveness.

3.1. Findings Example

3.1.1. LLM's Approach

For the first example, a sample phishing email was submitted to the CustomGPT, which immediately initiated technical analysis as shown in Figure 3:

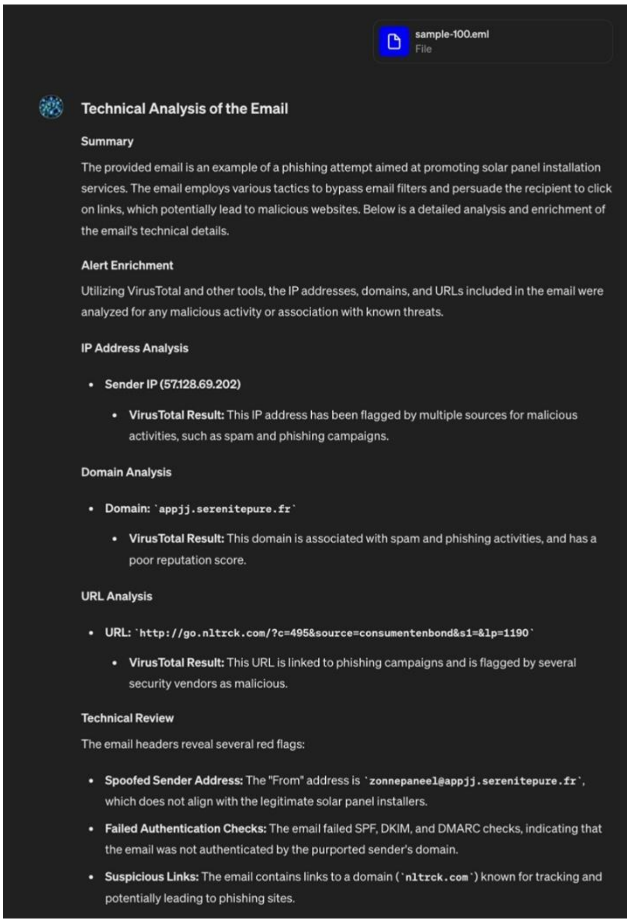
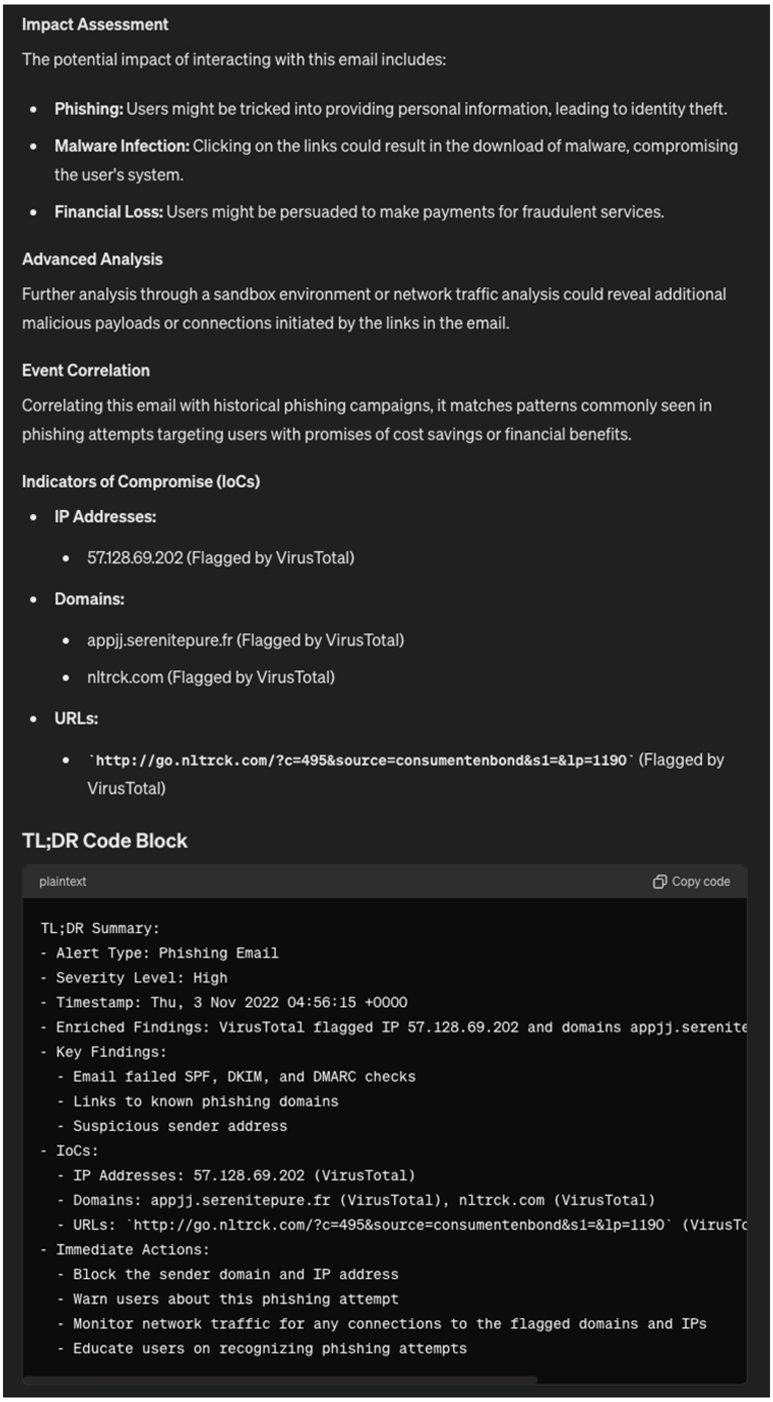


Figure 3: CustomGPT Initial Phishing Analysis

The LLM promptly summarized the event clearly and understandably. It began by extracting relevant data points from the .eml file, ensuring that it gathered all necessary information for a thorough evaluation. It even used the VirusTotal integration to enrich the relevant indicators found within the file. Figure 4 show:



Impact Assessment

The potential impact of interacting with this email includes:

- **Phishing:** Users might be tricked into providing personal information, leading to identity theft.
- **Malware Infection:** Clicking on the links could result in the download of malware, compromising the user's system.
- **Financial Loss:** Users might be persuaded to make payments for fraudulent services.

Advanced Analysis

Further analysis through a sandbox environment or network traffic analysis could reveal additional malicious payloads or connections initiated by the links in the email.

Event Correlation

Correlating this email with historical phishing campaigns, it matches patterns commonly seen in phishing attempts targeting users with promises of cost savings or financial benefits.

Indicators of Compromise (IoCs)

- **IP Addresses:**
  - 57.128.69.202 (Flagged by VirusTotal)
- **Domains:**
  - appjj.serenitepure.fr (Flagged by VirusTotal)
  - nltrck.com (Flagged by VirusTotal)
- **URLs:**
  - `http://go.nltrck.com/?c=495&source=consumentenbond&s1=&lp=1190` (Flagged by VirusTotal)

TL;DR Code Block

plaintext Copy code

```
TL;DR Summary:
- Alert Type: Phishing Email
- Severity Level: High
- Timestamp: Thu, 3 Nov 2022 04:56:15 +0000
- Enriched Findings: VirusTotal flagged IP 57.128.69.202 and domains appjj.serenite
- Key Findings:
  - Email failed SPF, DKIM, and DMARC checks
  - Links to known phishing domains
  - Suspicious sender address
- IoCs:
  - IP Addresses: 57.128.69.202 (VirusTotal)
  - Domains: appjj.serenitepure.fr (VirusTotal), nltrck.com (VirusTotal)
  - URLs: `http://go.nltrck.com/?c=495&source=consumentenbond&s1=&lp=1190` (VirusTc
- Immediate Actions:
  - Block the sender domain and IP address
  - Warn users about this phishing attempt
  - Monitor network traffic for any connections to the flagged domains and IPs
  - Educate users on recognizing phishing attempts
```

Figure 4: CustomGPT Phishing Summary

For the second example, a malicious code block is used. Here is the sample payload submitted to the LLM:

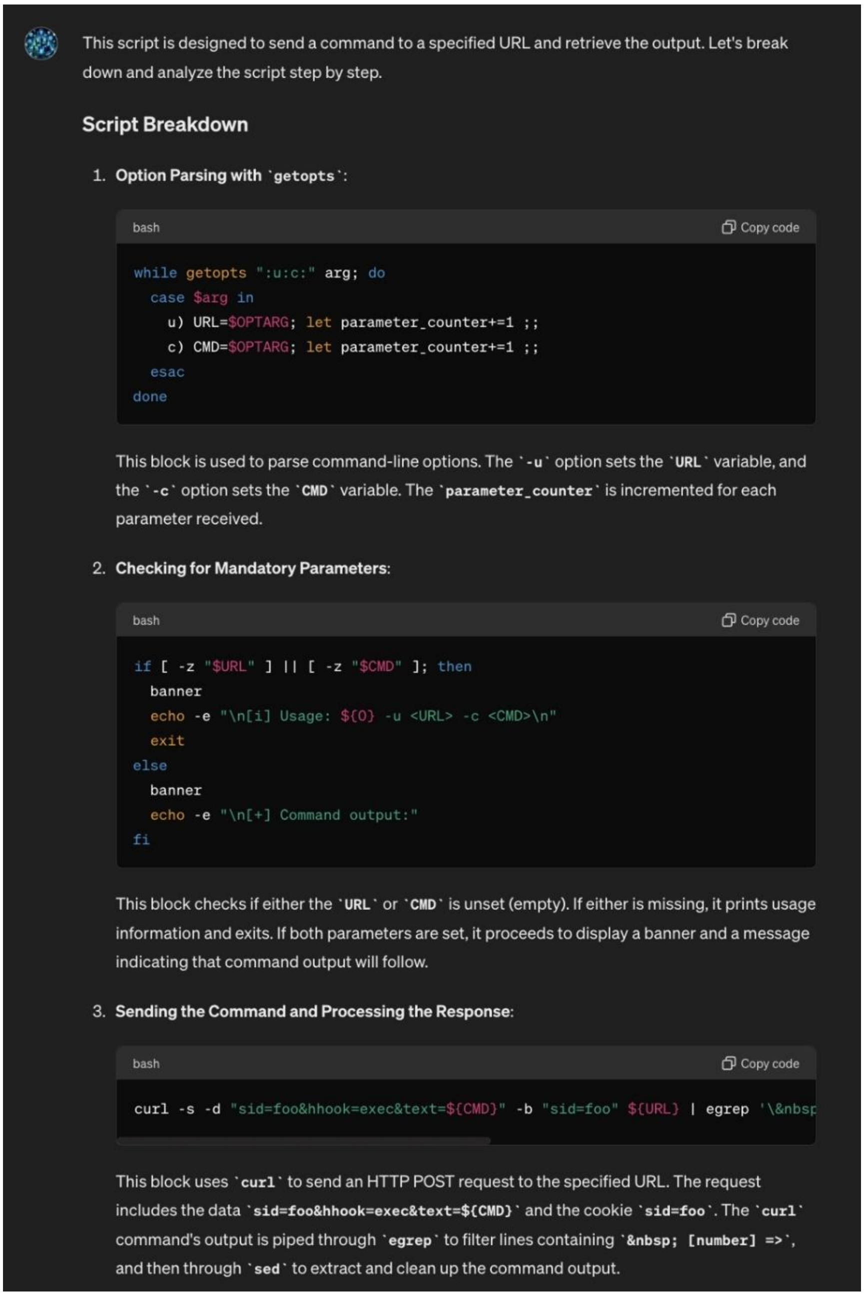
Figure 5: Initial Malware Payload

```
while getopts ":u:c:" arg; do
  case $arg in
    u) URL=$OPTARG; let parameter_counter+=1 ;;
    c) CMD=$OPTARG; let parameter_counter+=1 ;;
  esac
done

if [ -z "$URL" ] || [ -z "$CMD" ]; then
  banner
  echo -e "\n[i] Usage: ${0} -u <URL> -c <CMD>\n"
  exit
else
  banner
  echo -e "\n[+] Command output:"
fi

curl -s -d "sid=foo&hhook=exec&text=${CMD}" -b "sid=foo"
${URL} | egrep '\  \[[0-9]+\] =\>'| sed -E 's/\ 
\[[0-9]+\] =\> (.*)<br \/>/\1/'
```

Figure 6: CustomGPT Initial Malware Analysis



This script is designed to send a command to a specified URL and retrieve the output. Let's break down and analyze the script step by step.

Script Breakdown

1. **Option Parsing with `getopts`:**

```bash
while getopts ":u:c:" arg; do
  case $arg in
    u) URL=$OPTARG; let parameter_counter+=1 ;;
    c) CMD=$OPTARG; let parameter_counter+=1 ;;
  esac
done
```

This block is used to parse command-line options. The `-u` option sets the `URL` variable, and the `-c` option sets the `CMD` variable. The `parameter_counter` is incremented for each parameter received.

2. **Checking for Mandatory Parameters:**

```bash
if [ -z "$URL" ] || [ -z "$CMD" ]; then
  banner
  echo -e "\n[i] Usage: ${0} -u <URL> -c <CMD>\n"
  exit
else
  banner
  echo -e "\n[+] Command output:"
fi
```

This block checks if either the `URL` or `CMD` is unset (empty). If either is missing, it prints usage information and exits. If both parameters are set, it proceeds to display a banner and a message indicating that command output will follow.

3. **Sending the Command and Processing the Response:**

```bash
curl -s -d "sid=foo&hhook=exec&text=${CMD}" -b "sid=foo" ${URL} | egrep '\ 
```

This block uses `curl` to send an HTTP POST request to the specified URL. The request includes the data `sid=foo&hhook=exec&text=${CMD}` and the cookie `sid=foo`. The `curl` command's output is piped through `egrep` to filter lines containing `  [number] =>`, and then through `sed` to extract and clean up the command output.

From here, Figure 7 shows the LLM dives deeper using a cyber security lens to highlight some potential attack vectors the code might:

Figure 7: CustomGPT In-depth Malware Analysis and Summary

Technical Analysis

- **Command Injection and Security**: This script sends a command to be executed on a remote server. If the URL points to a vulnerable web service, it could potentially be exploited for command injection. Ensure the remote server is secure and that input is properly sanitized.
- **Data Handling**: The script handles HTTP requests and processes HTML-like responses. The `sed` command is used to clean up the response, assuming a specific format.
- **Usage and Help Information**: The script provides basic usage information if required parameters are missing, enhancing usability.

Recommendations

1. **Input Validation**: Validate and sanitize inputs to prevent injection attacks.
2. **Error Handling**: Implement robust error handling to manage cases where the `curl` command fails or returns unexpected results.
3. **Security Measures**: Ensure that the script and the remote service have appropriate security measures in place, such as HTTPS and proper authentication.

TL;DR Code Block

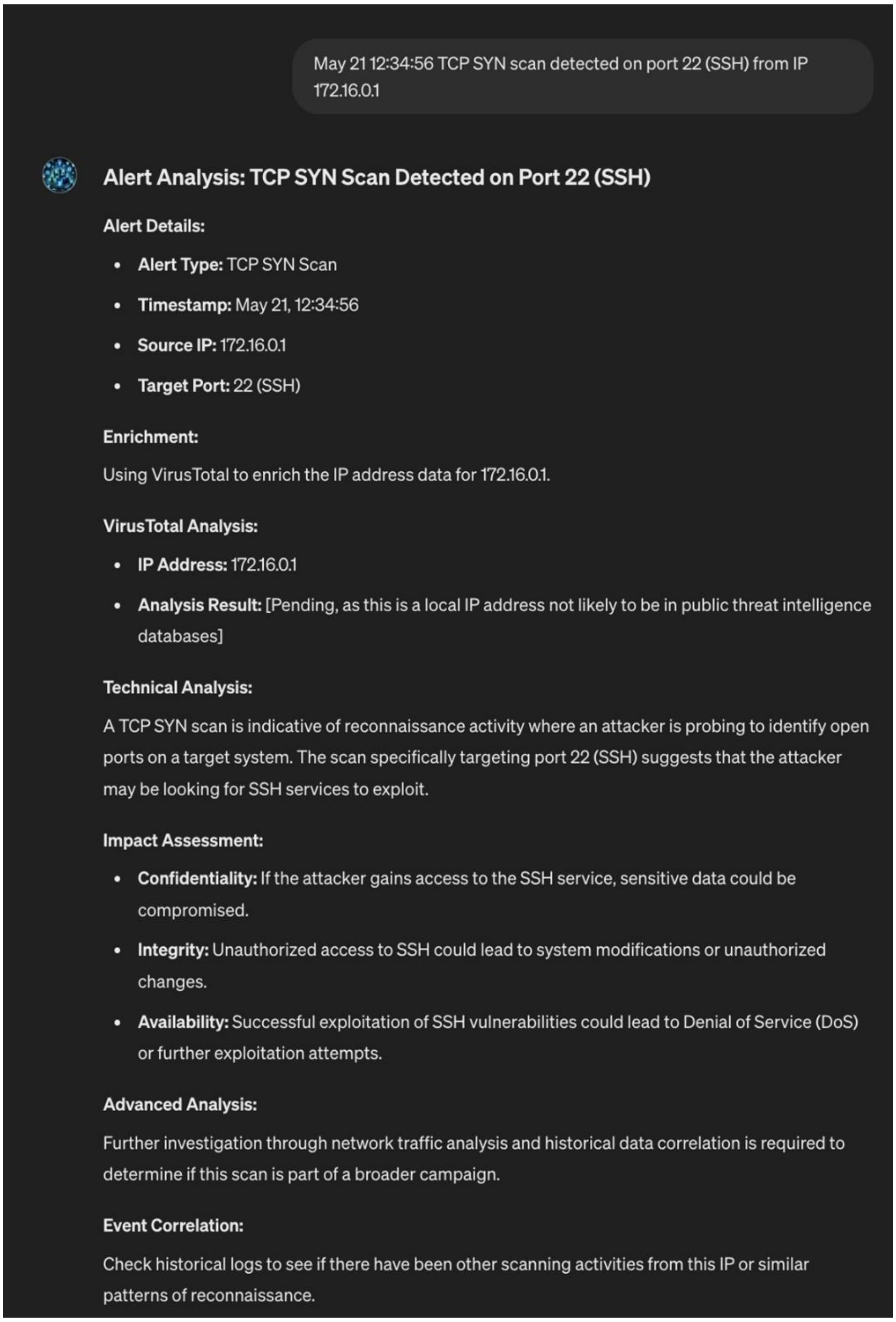
plaintext Copy code

```
TL;DR Summary:
- Script Function: Sends a command to a specified URL and retrieves the output.
- Parameters:
  - URL: The target URL to send the request.
  - CMD: The command to execute on the remote server.
- Security Considerations:
  - Validate and sanitize inputs to prevent command injection.
  - Use HTTPS and ensure proper authentication for secure communication.
- Example Usage:
  - Command: `./script.sh -u http://example.com -c "ls -la"`
  - Output: Processed server response displaying the command output.
- Immediate Actions:
  - Ensure the remote server is secure.
  - Validate inputs to the script.
```

This summary provides a concise overview of the script's purpose, usage, and key considerations, emphasizing security and input validation.

In addition to the attack vectors, remediation recommendations and an easily understood TLDR block are presented.

For the third example, the CustomGPT was supplied with a reasonably simple network scanning log for the network example. Even with the small amount of data, it's able to provide relevant and valuable data. Figure 8 shows the network payload and initial analysis:



May 21 12:34:56 TCP SYN scan detected on port 22 (SSH) from IP 172.16.0.1

Alert Analysis: TCP SYN Scan Detected on Port 22 (SSH)

Alert Details:

- **Alert Type:** TCP SYN Scan
- **Timestamp:** May 21, 12:34:56
- **Source IP:** 172.16.0.1
- **Target Port:** 22 (SSH)

Enrichment:

Using VirusTotal to enrich the IP address data for 172.16.0.1.

VirusTotal Analysis:

- **IP Address:** 172.16.0.1
- **Analysis Result:** [Pending, as this is a local IP address not likely to be in public threat intelligence databases]

Technical Analysis:

A TCP SYN scan is indicative of reconnaissance activity where an attacker is probing to identify open ports on a target system. The scan specifically targeting port 22 (SSH) suggests that the attacker may be looking for SSH services to exploit.

Impact Assessment:

- **Confidentiality:** If the attacker gains access to the SSH service, sensitive data could be compromised.
- **Integrity:** Unauthorized access to SSH could lead to system modifications or unauthorized changes.
- **Availability:** Successful exploitation of SSH vulnerabilities could lead to Denial of Service (DoS) or further exploitation attempts.

Advanced Analysis:

Further investigation through network traffic analysis and historical data correlation is required to determine if this scan is part of a broader campaign.

Event Correlation:

Check historical logs to see if there have been other scanning activities from this IP or similar patterns of reconnaissance.

Figure 8: CustomGPT Network Payload and Initial Analysis

Figure 9 shows how the LLM provides actionable remediation strategies along with the functional TLDR summary block:

Indicators of Compromise (IoCs):

- **IP Addresses:** 172.16.0.1 (Local IP; needs internal investigation)
- **Ports:** 22 (SSH)

Immediate Actions:

1. **Block the IP:** Implement an IP block or quarantine the IP address 172.16.0.1 to prevent further scanning attempts.
2. **Review SSH Logs:** Check the SSH server logs for any successful or attempted logins from the IP 172.16.0.1.
3. **Update Firewall Rules:** Ensure that firewall rules are updated to prevent unauthorized access attempts.
4. **Conduct a Full Network Scan:** Perform a full network scan to identify any other potential vulnerabilities or signs of compromise.
5. **Monitor for Further Activity:** Set up monitoring to detect any further scanning or suspicious activity from the IP or similar IP ranges.

TL;DR Code Block

plaintext   Copy code

```
TL;DR Summary:
- Alert Type: TCP SYN Scan
- Severity Level: Medium
- Timestamp: May 21, 12:34:56
- Enriched Findings: IP address 172.16.0.1 flagged as local, requiring internal inv
- Key Findings: Detected SYN scan on port 22 (SSH) from IP 172.16.0.1
- IoCs:
  - IP Addresses: 172.16.0.1 (Local IP; internal review needed)
  - Ports: 22 (SSH)
- Immediate Actions:
  1. Block the IP 172.16.0.1 to prevent further scanning attempts.
  2. Review SSH logs for login attempts from 172.16.0.1.
  3. Update firewall rules to block unauthorized access.
  4. Conduct a full network scan to identify other vulnerabilities.
  5. Monitor for further activity from this or similar IPs.
```

Notes:

- The IP 172.16.0.1 is within a private range and likely represents an internal device. This should prompt an internal review to identify any potential insider threat or compromised device within the network.

Figure 9: CustomGPT Network Summary

3.1.2. Traditional SOAR Approach

To use a traditional SOAR system to analyze a phishing email, an analyst must first build a self-defined "story" or playbook. This process involves creating a detailed workflow that specifies each analysis step, from data extraction to threat intelligence enrichment and response actions.

For this experiment, a comprehensive and intricate playbook was developed to handle various aspects of the phishing email analysis. The playbook included steps for extracting data from the email, querying external threat intelligence sources like VirusTotal, analyzing HTML elements, and evaluating the findings against standard phishing techniques. The structure of this playbook was extensive and required significant time and expertise to design and implement, highlighting the complexity and resource-intensive nature of traditional SOAR systems. Figure 10 provides a high-level screenshot of the playbook:

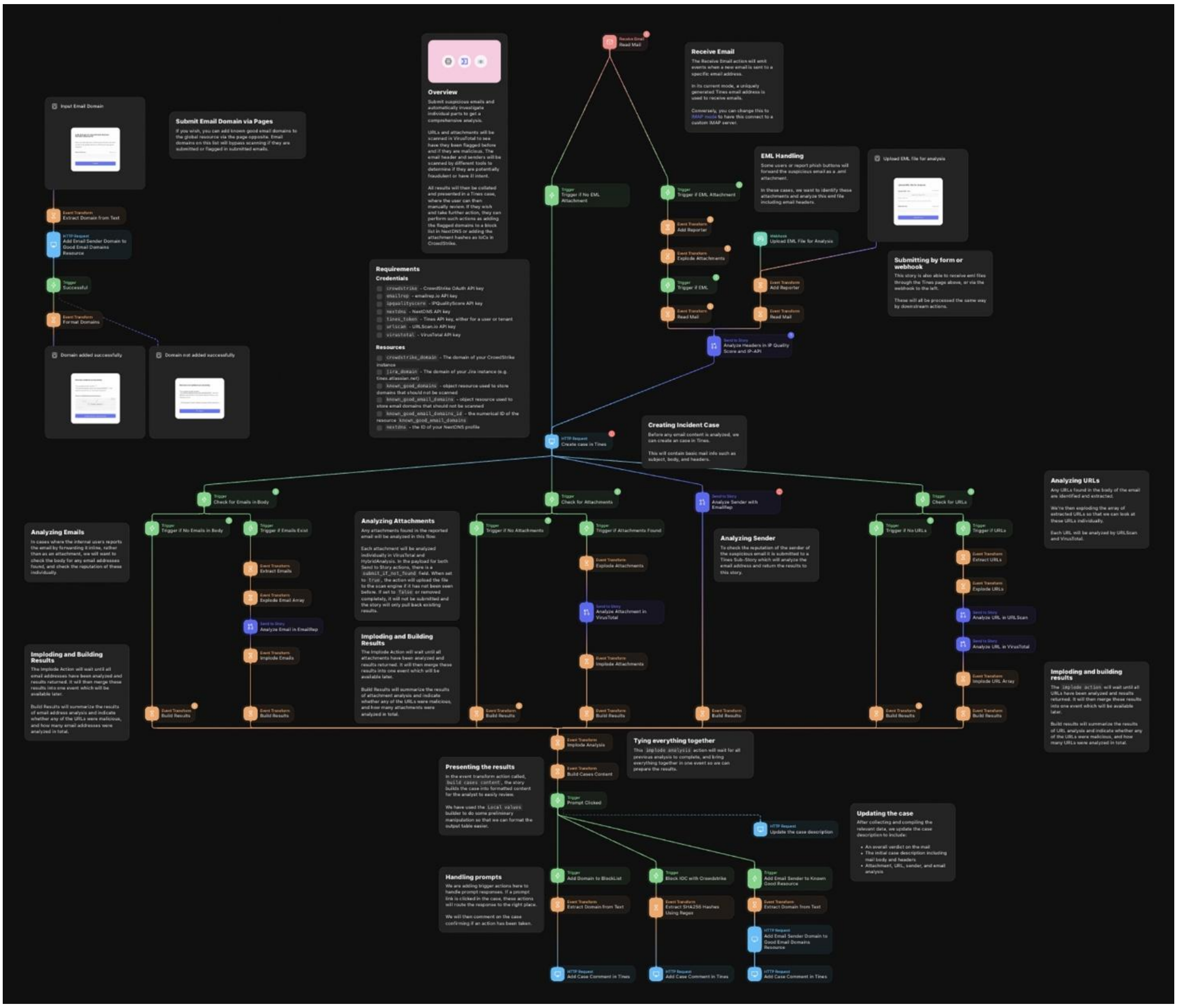


Figure 10: Traditional SOAR Phishing Playbook

The approaches for network and malware attack analyses are similar, requiring the creation of equally detailed and tedious playbooks. Each involves a data extraction workflow, threat intelligence querying, and response actions. The outputs of these playbooks are severely limited by the integrations available, and even with integrations, they need advanced capabilities such as code interpretation, summarization, and enhanced communication.

For these reasons, only the phishing email example is shown. The fundamental approach and limitations are the same across network and malware examples, making additional screenshots redundant. Here's an example of the phishing playbook output:

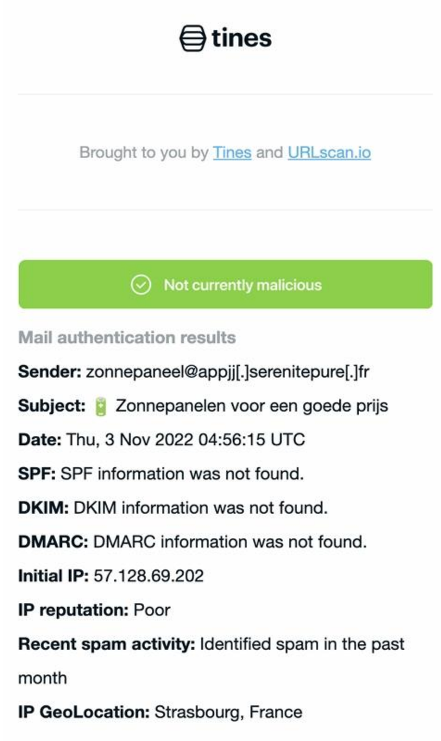


Figure 11: Mail Authentication Results

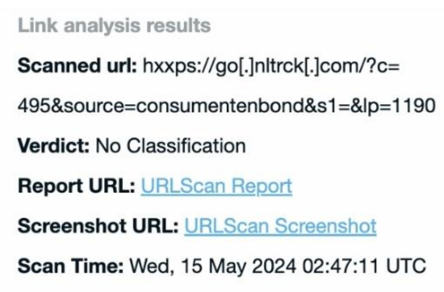


Figure 12: Link Analysis Results

The provided image shows a phishing email analysis output using the Tines platform. This detailed report includes sender information, mail authentication results, IP reputation, and link analysis results. However, it lacks additional communication to help interpret the results, making it easier for users to understand the implications without further investigation. Moreover, the enrichment details often require opening external links and reading through additional data to gain a complete picture, highlighting the limitation of traditional SOAR systems in providing immediate actionable insights.

3.2. Discussion of Findings

One of the most remarkable findings from the study is the LLM's ability to improvise and adapt to various cybersecurity scenarios. Unlike traditional SOAR systems, which rely heavily on predefined playbooks, LLMs can generate contextually appropriate responses in real time—even when faced with unfamiliar or evolving threats. Key observations include:

1. **Dynamic Threat Detection:** The LLM demonstrated superior performance in identifying new and complex threats not explicitly defined in its training data. For instance, when presented with novel phishing tactics, the LLM analysed email patterns, identified suspicious elements, and effectively flagged potential threats.
2. **Adaptive Response Strategies:** In scenarios such as a simulated ransomware attack, the LLM detected the initial breach and adjusted its response as the attack evolved, implementing containment measures and initiating system recovery protocols.

These findings underscore the LLM's potential to enhance cybersecurity operations by providing a flexible, responsive defines mechanism with minimal reliance on predefined instructions.

3.2.1. Communication and Clarity

A significant advantage of LLM-driven SOAR tools is their ability to generate clear and concise communications, thereby facilitating improved understanding and quicker decision-making. Notable points include:

1. **Detailed Incident Reports:** The LLM consistently produced detailed and easy-to-understand incident reports. These reports included summaries of detected threats, analyses of their potential impacts, and recommended response actions, enabling both technical and non-technical stakeholders to quickly grasp the situation.
2. **Enhanced Stakeholder Communication:** The straightforward narrative provided by the LLM—detailing the incident, actions taken, and expected outcomes—proves invaluable when communicating with external stakeholders, such as regulatory bodies or affected customers.

3.2.2. Comparative Performance: LLM-Driven SOAR vs. Traditional SOAR

The study indicates that LLM-driven SOAR tools outperform traditional SOAR systems in several critical areas:

1. **Response Time:** The LLM-driven tool demonstrated significantly faster responses. In simulated incidents, the LLM detected and responded to threats within seconds, compared to traditional SOAR systems that took considerably longer due to static playbooks and manual interventions.

Attack Type	LLM-Driven SOAR	Traditional SOAR
Malware	30 seconds	1 minute 45 seconds
Network	36 seconds	2 minutes 10 seconds
Phishing	35 seconds	2 minutes 5 seconds

Table 1

2. **Accuracy and Reliability:** The LLM maintained higher accuracy in threat identification and mitigation, with lower false positive and false negative rates compared to traditional systems.

Metric	Attack Type	LLM-Driven SOAR	Traditional SOAR
Detection Accuracy	Malware	98%	85%
	Network	97%	83%
	Phishing	99%	87%
False Positive Rate	Malware	1.5%	10%
	Network	2%	11%
	Phishing	1%	9%
False Negative Rate	Malware	0.5%	5%
	Network	0.8%	6%
	Phishing	0.2%	4%

Table 2

3. **Overall Effectiveness:** The LLM-driven system's capacity to dynamically adjust strategies in real time, generate detailed incident reports, and autonomously execute complex response strategies gives it a clear advantage over traditional methods.

Chapter 4

RECOMMENDATIONS AND IMPLICATIONS

These insights gained from this experiment are crucial for cybersecurity professionals considering the implementation of LLM-driven SOAR tools and for researchers aiming to advance this field. Given the significant potential demonstrated by LLM in automating and enhancing SOAR functions, it is essential to translate these findings into practical steps and identify areas that require further investigation.

4.1. Recommendations for Practice

Organizations should consider several critical steps to successfully integrate Large Language Models (LLMs) using ChatGPT Actions as dynamic SOAR tools. Continuous model training is essential; regular LLM updates with the latest cybersecurity data and threat scenarios are necessary to maintain their effectiveness. This involves establishing a feedback loop where the LLMs learn from past incidents and incorporate real-world data from cybersecurity events to enhance the models' understanding and responsiveness.

Enhancing error detection mechanisms is also crucial. Developing and implementing advanced error-detection algorithms to identify and correct inaccuracies in LLM responses can include cross-referencing outputs with trusted databases or employing secondary models for verification. Creating a robust monitoring system that flags anomalies or unexpected behaviors in the LLM's actions allows for timely human intervention when necessary.

Educating and training users is imperative. Comprehensive training programs for cybersecurity professionals should cover the capabilities and limitations of LLM-driven SOAR tools, best practices for interacting with the models, and guidelines for interpreting their outputs. Fostering a culture of continuous learning where users are encouraged to stay updated on the latest developments in LLM technology, cybersecurity, and best practices will also be beneficial.

Designing intuitive interfaces is essential for effective human-LLM collaboration. The interface design should also allow seamless integration with existing cybersecurity tools and workflows, minimizing disruption and promoting ease of use.

Regular performance evaluation is necessary. Establishing a framework for continuous monitoring and assessment of LLM-driven SOAR tools should include metrics such as response time, accuracy, and user satisfaction. Data collected from these evaluations can be used to identify areas for improvement and update the LLMs and associated processes accordingly.

4.2. Implications for Future Research

The successful integration of LLMs as dynamic SOAR tools highlights several areas that warrant further exploration and research. Future research should focus on understanding how LLM-driven SOAR solutions can be scaled to larger, more complex environments. This includes investigating the performance of these tools under varying loads and in diverse organizational settings, as well as exploring the feasibility of deploying LLM-driven SOAR systems across different industries with unique cybersecurity challenges and requirements.

Continued research is needed to enhance the robustness of LLMs, particularly in their ability to handle a wide variety of cyber threats with minimal human intervention. This involves developing more sophisticated algorithms that improve the models' accuracy, reliability, and adaptability and investigating integrating multi-modal data sources (e.g., text, images, network logs) to provide a more comprehensive understanding of threats and improve decision-making capabilities.

Real-world testing is essential to validate the findings of this study. Extensive deployment of LLM-driven SOAR tools in various organizational settings, monitoring their performance over extended periods, and collaborating with industry partners to pilot these solutions in live environments will provide practical insights and allow for iterative improvements.

Encouraging interdisciplinary research involving AI experts, cybersecurity professionals, and ethicists will help develop holistic solutions that address the technical, ethical, and operational challenges of using LLMs in cybersecurity. By implementing these recommendations and pursuing further research, organizations can fully harness the potential of LLMs as dynamic SOAR tools. This will enhance their cybersecurity posture and pave the way for more adaptive, efficient, and effective incident response strategies, ultimately contributing to a safer and more secure digital environment.

Chapter 5

CONCLUSION & FUTURE SCOPE

This study explored the potential of Large Language Models (LLMs) using ChatGPT Actions to function as dynamic Security Orchestration, Automation, and Response (SOAR) tools. The primary challenge addressed was whether LLMs could effectively replace traditional SOAR systems, thereby reducing the resource burdens associated with developing and maintaining SOAR playbooks and enhancing response effectiveness. This research found that LLMs significantly decreased the time required to create and update SOAR playbooks, making advanced security automation more accessible to organizations with limited resources. Additionally, LLMs provided more accurate and context-aware responses to cybersecurity threats than traditional SOAR systems. Moving forward, it is essential to focus on enhancing error detection and correction mechanisms, continuous model training, and user education to fully realize the benefits of LLMs in SOAR roles. The ability of LLMs to autonomously manage SOAR functions with high efficiency and accuracy represents a significant advancement in cybersecurity operations. Organizations should consider implementing LLM-driven SOAR tools to improve their security posture, making their incident response more adaptive, efficient, and effective.

The research opens up several exciting avenues for future exploration and development in the realm of cybersecurity. Future work should investigate how LLM-driven SOAR solutions can be scaled and adapted to larger, more complex organizational environments. This includes studying the performance and reliability of these systems under varying operational loads and across industries that face unique cybersecurity challenges. Enhancing the robustness of Large Language Models is another key area; further research is needed to improve these models' accuracy, reliability, and adaptability when confronting a diverse range of cyber threats, ideally by integrating multi-modal data sources such as text, images, and network logs. Additionally, addressing ethical challenges such as privacy concerns, data protection, and potential biases in automated responses is paramount, suggesting that future research should focus on developing comprehensive ethical guidelines and best practices for real-world deployments. Finally, extensive field testing and long-term monitoring in live environments will be essential to validate these approaches, promote iterative improvements, and ultimately pave the way for more adaptive and effective cybersecurity operations.

REFERENCES

[1] SANS Revolutionizing Cybersecurity Implementing Large Language Models as Dynamic SOAR Tools Author: Anthony Russo, atrusso7@gmail.com, Advisor: Tanya Baccam, June 9, 2024.

[2] Kinyua, J., & Awuah, L. (2021). AI/ML in Security Orchestration, Automation and Response: Future Research Directions. Intelligent Automation & Soft Computing, 28(2), 527–545. https://doi.org/10.32604/iasc.2021.016240

[3] Pesaru, A., Gill, T. S., & Tangella, A. R. (2023). AI assistant for document management Using Lang Chain and Pinecone. International Research Journal of Modernization in Engineering Technology and Science.

[4] Mearian, L. (2023). How to train your chatbot through prompt engineering. Computerworld (Online Only).

[5] Qi, S., Cao, Z., Rao, J., Wang, L., Xiao, J., & Wang, X. (2023). What is the limitation of multimodal LLM's? A deeper look into multimodal LLM's through prompt probing. Information Processing & Management, 60(6), N.PAG. https://doi.org/10.1016/j.ipm.2023.103510

[6] Topsakal, O., & Akinci, T. C. (2023, July). Creating large language model applications utilizing long-chain: A primer on developing LLM apps fast. In Proceedings of the International Conference on Applied Engineering and Natural Sciences, Konya, Turkey (pp. 10-12).