Enhancing 6G Network Resilience through HORSE's LLM Agent-Based Security Mechanisms

M. Danousis*, A. Piemonti†, F. Granelli‡, X. Masip-Bruin§, E. Rodriguez§, A. Carrega¶,
C. Skianis*, E. Kafetzakis*, I. Giannoulakis*
*Eight Bells Ltd, Nicosia, Cyprus
†Martel Innovate, Chiasso, Switzerland
‡CNIT / University of Trento, Trento, Italy
§CRAAX / Universitat Politècnica de Catalunya, Vilanova i la Geltrú, Barcelona, Spain
¶CNIT - National, Inter-University Consortium for Telecommunications, Genoa, Italy

***Abstract*—The HORSE (Holistic, Omnipresent, Resilient Services for future 6G wireless and computing Ecosystems) framework integrates AI to address the complexity and vulnerabilities of next-generation networks. This paper focuses on an LLM-agent-based subsystem of HORSE that automates the creation and enforcement of security mitigation actions. This subsystem utilizes a specialized Knowledge Base enriched with threat-mitigation pairs from trusted sources like MITRE ATT&CK, through a collaborative workflow between an LLM and human expertise. LLMs translate high-level mitigation actions into executable Ansible commands, while a feedback-driven workflow, powered by LLM agents, ensures accuracy and reliability. One LLM agent (the "translator") generates the commands, while another (the "executor") executes them and provides feedback to the translator in case of unsuccessful execution, allowing the translator to refine the commands for a retry. Experiments with open-source LLMs (like Llama 2 and Falcon) demonstrate rates of up to 73% in execution success, and high accuracy in translating mitigation intentions. This approach automates threat mitigation, reduces human intervention, and scales effectively for the dynamic security demands of 6G networks. Future work will focus on enhancing LLM capabilities for complex environments and improving adaptability in cybersecurity operations.**

***Index Terms*—LLMs for 6G security, LLM agents, AI/ML-based mitigation actions, 6G and AI**

I. Introduction

A. 5G and 6G Cybersecurity Challenges

The upcoming sixth-generation (6G) networks are expected to revolutionize connectivity by offering unprecedented levels of heterogeneity, scalability, reliability, security, and energy efficiency [1]. However, this increased complexity introduces new attack vectors and vulnerabilities, necessitating a robust end-to-end security framework. The interoperability of 6G networks further amplifies the challenge of securing these systems, making security and trust fundamental [2] yet difficult to achieve. In this complex landscape, artificial intelligence (AI) emerges as a powerful tool capable of addressing these challenges [3] by providing both reactive and proactive security measures, such as threat detection and attack prevention. Recent advancements in AI, particularly with the advent of Large Language Models (LLMs), have unlocked new possibilities for enhancing cybersecurity. LLMs exhibit exceptional adaptability across different domains, particularly when fine-tuned with domain-specific expertise [4]. As cybersecurity threats evolve rapidly, traditional approaches, such as signature-based detection and rule-based systems, struggle to keep pace [5]. On the contrary, despite being a relatively recent innovation, preliminary efforts have demonstrated the potential of LLMs in cybersecurity applications [6]. For instance, in threat intelligence, LLMs can process vast amounts of data to identify patterns and extract critical insights, a task that traditionally demands extensive human effort and expertise. Moreover, LLMs have shown good capabilities in secure code generation, ensuring that software adheres to the best security practices while maintaining functionality. A key advantage of LLMs lies in their ability to leverage pre-trained models, trained on massive datasets, enabling them to generalize knowledge across diverse cybersecurity scenarios. The availability of open-source (or preferably, open-weights) LLMs, such as Llama 2 or Mistral models, allows for further fine-tuning using domain-specific datasets, enhancing their effectiveness for targeted cybersecurity applications. Researchers have also developed specialized pre-trained models trained on security datasets and evaluated against security benchmarks. These models integrate expert knowledge to provide accurate and actionable insights, making them valuable assets in real-world cybersecurity operations. Furthermore, LLMs offer techniques to enhance their specialization without the need for extensive retraining. Model editing techniques allow for precise modifications, incorporating new cybersecurity knowledge while preserving existing capabilities. Similarly, prompt engineering strategies can guide LLMs to generate highly relevant and accurate responses, mitigating the challenges associated with limited training data and computational resources.

B. Motivation for the HORSE project

HORSE [7] is a pioneering architecture designed to establish a robust foundation for end-to-end security service management in future 6G networks. It embodies a human-centric, open-source, green, and sustainable approach, offering coordinated provisioning and protection within the envisioned 6G landscape. While 5G networks have introduced significant improvements in communication performance over previous generations, it presents limitations regarding scalability, latency, energy efficiency and security [8]. 6G networks are anticipated to surpass these limitations, though their standardization remains still an ongoing process. HORSE aims to bridge this gap by developing a versatile and advanced platform capable of anticipating challenges and implementing technological solutions to deliver omnipresent, intelligent, and secure network services. Within the HORSE framework, several security-driven services have been designed and implemented to address the unique challenges of 6G networks. One key feature is end-to-end network slicing isolation, enabling multiple devices to connect securely while ensuring service differentiation and preventing data leakage. AI is integrated as a native component within HORSE, extending beyond traditional optimization tasks to support advanced functionalities such as threat detection, predictive security, and secure coding. The project leverages cutting-edge technologies, including generative AI and large language models (LLMs), to enhance cybersecurity capabilities and adaptability in highly dynamic environments. Another core element of HORSE is the adoption of zero-trust architecture, which strengthens security by continuously verifying and validating all entities before granting access. HORSE also incorporates intent-based and programmable networking, which translates high-level user intents into device-specific rules and automated actions. This increases overall network performance and enables proactive security measures. By integrating such a diverse range of technologies, HORSE provides a comprehensive framework to analyze and address potential vulnerabilities and attack vectors across different scenarios. It offers valuable insights for the security implications of adopting 6G networks, enabling the development of security strategies and best practices for securing next-generation infrastructures by design.

In this work, we present several key contributions aimed at enhancing cybersecurity automation through the integration of LLMs within the HORSE framework:

1) **Utilization of LLMs to populate an attack-mitigation database:** LLMs are leveraged to generate data and populate a comprehensive database with attack-mitigation pairs, ensuring effective countermeasure deployment. A human-in-the-loop approach is used to validate the generated results.
2) **Automated translation of high-level countermeasures into Ansible commands:** LLMs are utilized to convert high-level cybersecurity countermeasures from natural language format into executable Ansible commands, enabling the automation of defensive actions within complex infrastructures. We implemented a workflow with LLM agents to facilitate a feedback loop between the LLMs and the system's state.
3) **Integration into the HORSE framework:** The successful applications of LLMs in attack mitigation and automated command generation are integrated into the HORSE framework, creating a comprehensive and adaptive cybersecurity system capable of dynamically responding to evolving threats.

II. Overview of the HORSE approach

A. Architecture Overview

As described above, HORSE is proposed as a pioneering architecture aimed at guaranteeing end-to-end security provisioning in future 6G networks. To the best of our knowledge, there is no solution that addresses security provisioning in a holistic manner within the 6G ecosystem, making HORSE an outstanding contribution in this area [9]. To achieve this, novel technologies are considered, setting the set of functional blocks and components that constitute the HORSE architecture.

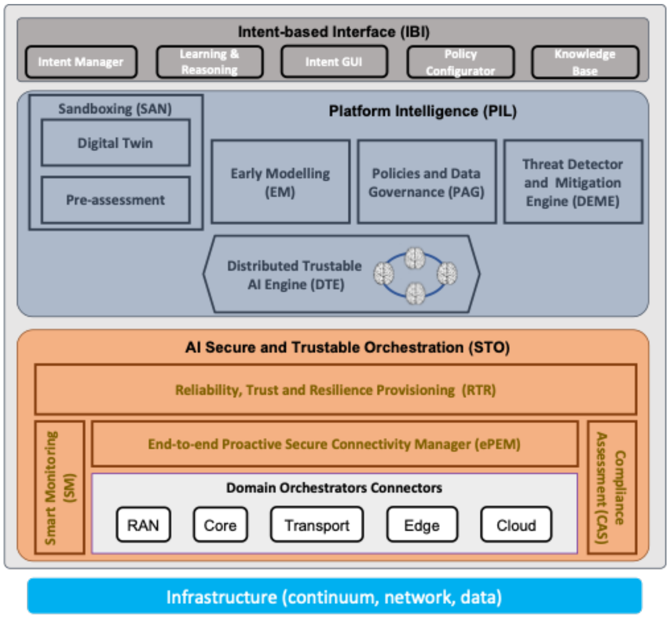

Intent-based Interface (IBI)
Intent Manager
Learning & Reasoning
Intent GUI
Policy Configurator
Knowledge Base
Sandboxing (SAN)
Digital Twin
Pre-assessment
Platform Intelligence (PIL)
Early Modelling (EM)
Policies and Data Governance (PAG)
Threat Detector and Mitigation Engine (DEME)
Distributed Trustable AI Engine (DTE)
AI Secure and Trustable Orchestration (STO)
Reliability, Trust and Resilience Provisioning (RTR)
Smart Monitoring (SM)
End-to-end Proactive Secure Connectivity Manager (ePEM)
Domain Orchestrators Connectors
RAN
Core
Transport
Edge
Cloud
Compliance Assessment (CAS)
Infrastructure (continuum, network, data)

Fig. 1. HORSE Functional Architecture

Fig. 1 presents a high-level overview of the main HORSE functional components, highlighting the key elements: **Platform Intelligence (PIL)**, which integrates the whole set of smart components, and **AI Secure and Trustable Orchestration (STO)**, responsible for delivering the proper set of actions to either mitigate or prevent an attack. Both components are assisted by a northbound interface known as **Intent-Based Interface (IBI)**, designed to facilitate an easy and explainable user interaction. Indeed, PIL becomes a central component of the architecture, routed on three key contributions. First, PIL includes an emulated environment, called **Sandboxing (SAN)** where all proactive actions can be analyzed and validated with no need to disrupt the real operational network. The analyses and studies carried out in the SAN will be extremely useful to predict or early detect an attack, as well as to check for the potential impact of an attack and the effectiveness of mitigation or preventive action on the real infrastructure. Second, three elements, the **Early Modeling (EM)**, including a strategy for modeling attacks, both known and zero-day attacks, the **Policies and Data Governance (PAG)**, dealing with aspects related to policies definition to be granted within the envisioned compliance test, and the threat **Detector and Mitigation Engine (DEME)**, responsible for detecting attacks and proposing mitigation measures, play, coordinated with the SAN, and assisted by the third key contribution, namely the **Distributed Trustable AI Engine (DTE)**, responsible for endowing all components with the required AI support, the strategic components to develop the proactive security provisioning approach proposed in HORSE. While PIL guarantees the proactive approach, the STO guarantees the required mitigation or preventive actions are properly deployed in the infrastructure, considering the distinct domains a 6G ecosystem is built upon, as shown in Fig. 1. The STO is responsible for properly orchestrating and deploying the set of actions to guarantee a secure performance but also policy compliance tests. Finally, the IBI provides an intent-based interface to facilitate users interaction, replacing traditional complex instructions by a set of intents easing the deployment of innovative technologies, such as for example LLMs. This whole set of functional blocks must properly interact, turning into different workflows, illustrating the distinct security functions to be provided by HORSE. For the sake of illustration, Fig. 2 presents a general workflow showing expected blocks interaction.

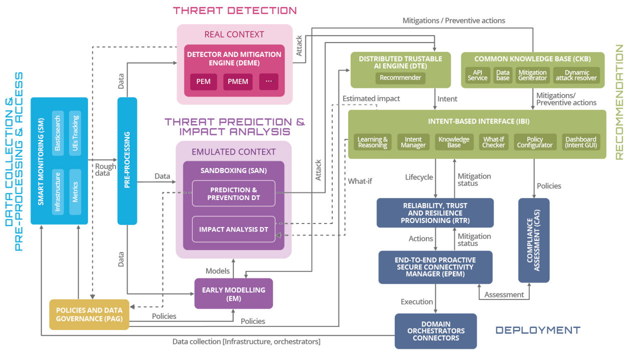


Fig. 2. HORSE General Workflow

B. Key Components

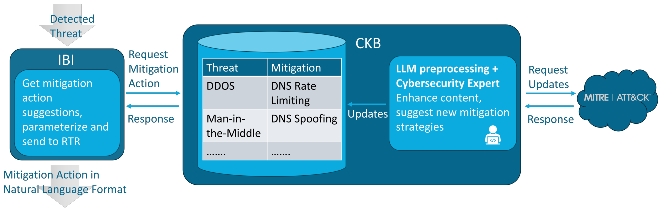
The HORSE architecture is designed to dynamically adapt to new and unforeseen threats. This is achieved through the development of specific modules and the automation of the mitigation procedures through LLMs.

In this scenario, the major functionalities offered by HORSE are the following:

- *Real Time Threat Detection*: The DEME continuously monitors the data collected by the infrastructure in order to quickly identify potential threats. The module is based on a Machine Learning approach trained on normal traffic patterns and known attacks, and it aims to identify and classify a potential attack in order to inform the DTE.
- *Threat Prediction and Impact Analysis*: A Sandbox module is provided, based on the concept of Network Digital Twin. The Network Digital Twin provides an isolated replica of the actual network infrastructure and it provides two functionalities: (i) prediction and prevention, early identification of attacks and anomalies, driven by the EM module, and (ii) impact analysis, to analyze potential mitigation actions.
- *AI-Based Threat Identification and Intent Definition*: The actual threat is analyzed by the DTE, which can operate in a central way or via federated learning, with the objective of understanding the details of the attack and define potential intents of mitigation of such attack.
- *Intent-Based Mitigation*: IBI receives the suggestions from the DTE and the knowledge stored in the **Common Knowledge Base (CKB)**. The CKB is an intelligent database that gathers data from reliable cybersecurity databases and uses generative AI to produce attack-mitigation pairs. IBI collects these data and, after checking the infrastructure policies, it defines a list of potential mitigation actions. Those actions will be tested through interaction with the SAN, and the best strategy will be selected for deployment on the network infrastructure.
- *Reliability, Trust, and Resilience*: The **Reliability, Trust and Resilience provisioning (RTR)** module provides a suite of tools and technologies to ensure secure system performance. It specifies the mitigation and preventive actions to be carried out by the **end-to-end Proactive Secure Connectivity Manager (ePEM)** component using Ansible security playbooks, based on the mitigation intent defined by the IBI.
- *The end-to-end secure connectivity manager*: ePEM functions as an Operations Support System (OSS), implementing the mitigation and preventive actions defined by RTR across the available infrastructure. It also manages and maintains records of deployed applications, network services, and available resources.

III. AI-ASSISTED THREAT MITIGATION WORKFLOW

A. Data Collection and Knowledge Base Population


Detected Threat
IBI
Get mitigation action suggestions, parameterize and send to RTR
Request Mitigation Action
Response
Mitigation Action in Natural Language Format
CKB
Updates
LLM preprocessing + Cybersecurity Expert
Enhance content, suggest new mitigation strategies
Request Updates
Response
MITRE | ATT&CK

Threat	Mitigation
DDOS	DNS Rate Limiting
Man-in-the-Middle	DNS Spoofing
....	....

Fig. 3. Architecture Overview of CKB-IBI Workflow

The data collection process is a critical phase of the HORSE architecture, aimed at aggregating cyber threat-mitigation action pairs from trusted sources to populate the CKB (Figure 3). The primary source for these pairs is the MITRE ATT&CK framework, renowned for its comprehensive cataloging of cyber threats and corresponding mitigations [10]–[12]. In addition to MITRE, other reliable cybersecurity databases are utilized to ensure a diverse and comprehensive collection of data points, capturing a wide range of potential threats and their corresponding countermeasures.

To further improve the knowledge and relevance of the CKB, we have implemented advanced techniques leveraging LLMs. These models analyze the aggregated data to enhance its content by suggesting additional mitigation strategies and generating new ones where none are provided. This approach enriches the dataset, ensuring comprehensive and actionable intelligence. As a result, our method enhances the quality of the knowledge base using the power of generative AI, while enabling more effective threat mitigation.

A human-in-the-loop strategy is incorporated into the process, allowing cybersecurity experts to review the generated attack-mitigation pairs and verify their accuracy and relevance. There literature that explores the *human-in-the-loop* strategy for tackling cybersecurity issues is substantial [13], [14]. This expert validation step is crucial for maintaining the integrity of the CKB, as it reduces potential inaccuracies that may arise from fully automated processes. The feedback provided by experts plays a critical role in evaluating the performance of the LLMs, ensuring that the model outputs align with the real-world cybersecurity requirements.

This data collection and validation process is conducted at regular intervals to ensure that the CKB stays up-to-date with the latest threat intelligence. Regular updates is of great importance in the dynamic field of cybersecurity, where new threats and vulnerabilities emerge rapidly. The integration of automated LLM-driven processes with expert oversight creates a robust framework for continuously enhancing the HORSE platform's security capabilities.

B. Mitigation Actions Using LLM and Ansible

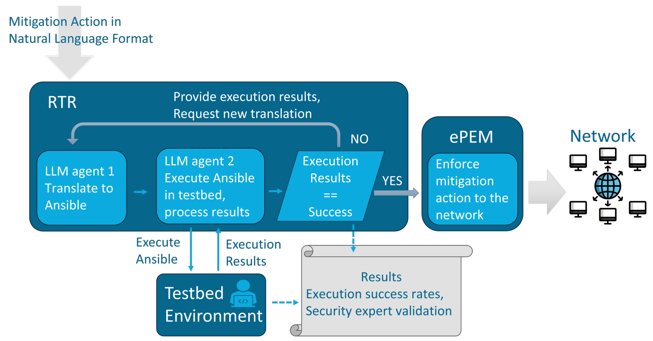


Fig. 4. Architecture Overview of RTR-ePEM Workflow

The integration of LLMs into the cybersecurity domain, particularly for automating mitigation actions, represents a significant advancement in reactive security strategies [15]. In our approach, we utilize LLMs to translate mitigation actions, which are often described in natural language, into executable Ansible commands (Figure 4). This translation process is critical for bridging the gap between high-level threat descriptions and the precise, actionable commands needed to implement effective countermeasures within a networked environment.

The workflow begins with the collection of mitigation actions in natural language format using the intelligence gathered by the CKB. Each collected action is then processed by a dual-agent system within the LLM framework. The **first agent**, referred to as the Command Generator, constructs an initial Ansible command based on the given mitigation action. This command is subsequently passed to the **second agent**, the Command Executor, which tests the command within a controlled testbed environment.

A **feedback loop mechanism** is employed to refine the generated commands. If the initial command, generated by Command Generator, fails to execute correctly, the Command Executor provides detailed feedback to the Command Generator, which then modifies the command accordingly. This iterative process is repeated up to three times per command|number three was determined by the cybersecurity experts based on the overall latency tolerance for the complete workflow|with the objective of achieving successful execution. Commands that are successfully executed within these three iterations are marked as *successful*, while those that fail after three attempts are recorded as *failures*. This iterative feedback approach is similar to methods used in other domains where LLMs are employed for code generation and problem-solving tasks [16].

The effectiveness of this automated mitigation strategy is evaluated across two primary metrics: executability and correct intention translation. Executability refers to the percentage of commands that were successfully executed in the testbed environment, providing a measure of the system's practical reliability. Recent studies like [17] have also been exploring the utilization of LLMs for modifying Ansible scripts. Correct intention translation, on the other hand, is assessed by a cybersecurity expert who evaluates whether the executed commands accurately reflect the intended mitigation actions. This dual evaluation framework ensures both functional and semantic accuracy in the automated responses. Similar evaluation methodologies have been discussed in previous studies comparing LLM performance in programming and automation tasks [18].

The collected data, including the number of iterations required for successful execution and expert assessments of intention accuracy, offer valuable insights into the capabilities and limitations of LLMs in this context. By analyzing these outcomes, we can refine our models and improve the overall system's robustness and effectiveness.

IV. Experiment Setup and Results

A. Experiment Setup

1) Dataset Definition: To create a challenging, scalable, and diverse dataset, we combined input from human security experts, trusted organizations like MITRE, and LLMs. Initially, experts defined four categories of mitigation actions (e.g., **DNS service disabling, rate limiting, firewall spoofing detection, server handover, and traffic absorption**) based on consultations with MITRE. They crafted initial sentences for each category, which were expanded using an LLM. Duplicate or overly similar sentences were filtered out, and the process was repeated with slightly more ambiguous inputs, allowing the

LLM to generate further sentences. After iterative refinement, the final dataset consisted of **230 sentences** with varying levels of ambiguity, designed to test how well different LLMs could handle translating diverse inputs into Ansible commands.

2) LLM Agents Workflow: We utilized **LangChain** framework [19] to set up two-agent workflows using, for each experiment, one of these three open-source LLMs: **Llama 2 (70B), Llama (7B), Falcon (40B)**. The number that comes after the name of each model refers to the size of each model. The reason behind the selection of the three models is that we wanted to compare the most popular open source LLM models, for replicability and usefulness purposes. The workflow, as described earlier, was applied to every sentence in the dataset, with results stored after testbed execution.

Execution outcomes were evaluated on a scale from 1 to 4, where 1 indicates successful execution on the first attempt and 4 signifies failure after three attempts. Following the execution phase, cybersecurity experts evaluated the successfully executed Ansible commands by their effectiveness in translating the original intention of each sentence into actionable commands. Using a 1 to 5 scale, three experts graded each command's intention accuracy, and the scores were averaged to produce a final grade for each LLM.

B. Execution Results

1) Success Rate: The success rate of each LLM|Llama 2 (70B), Llama (7B), Falcon (40B)| was determined by categorizing the results into successful (1, 2, or 3) or failed (4) outcomes. Llama 2 (70B) achieved the highest success rate of 73%, followed by Falcon (40B) at 65%, and Llama (7B) at 56%. These results indicate the overall effectiveness of each model in generating executable commands within three attempts.

2) Average Iterations Till Success: The average number of iterations required for successful command execution was evaluated for each LLM. Llama 2 (70B) required an average of 1.7 iterations, followed by Falcon (40B) with 1.5 iterations and Llama (7B) with 1.2 iterations. These results show the models' efficiency in producing accurate commands and the models' ability to adapt and correct it's mistakes when feedback are provided during the workflow. Llama 2 (70B) shows the lowest success rate on the first attempt, which can be translated as the highest in adaptability. The results are summarized in Table I.

LLM Model	Success (%)	Failure (%)	Avg. Iterations
Llama 2 (70B)	73%	27%	1.7
Llama (7B)	56%	44%	1.2
Falcon (40B)	65%	35%	1.5

TABLE I
SUCCESS RATE AND AVERAGE ITERATIONS FOR EACH LLM

3) Execution Time Comparison: All models were executed on the same hardware setup to ensure a fair comparison, utilizing an **Nvidia RTX 3090 GPU** for inference. The execution time for each model was calculated by combining its average inference time with the average number of iterations required for success. This provides an overall latency metric for each LLM.

The calculated average inference times, for both of the agents, are:

- **Llama 2 (70B):** 3.6 seconds per inference (1.8 sec. per agent).
- **Llama (7B):** 2.0 second per inference (1.0 sec. per agent).
- **Falcon (40B):** 2.8 seconds per inference (1.4 sec. per agent).

To calculate the overall latency, the average number of iterations until success was multiplied by the average inference time for each model. Llama 2 (70B), despite its slower inference time of 1.8 seconds, required the fewest iterations on average, resulting in a good total execution time. Llama (7B), with a faster inference time of 1.0 second, required significantly more iterations, increasing its overall latency. Falcon (40B) achieved a balance between inference speed and iteration efficiency, making it the best model. A notable fact is that the average prompt and response lengths were small enough (in terms of tokens) for the models to process them in a single pass. The overall results for the execution time for each model are shown in Fig. 5.

Notice that the overall execution time did not take into account latency caused by the testbed; this is due to the fact that a testbed's latency can depend on many parameters, adding a non-tolerable level of uncertainty to our experiments.

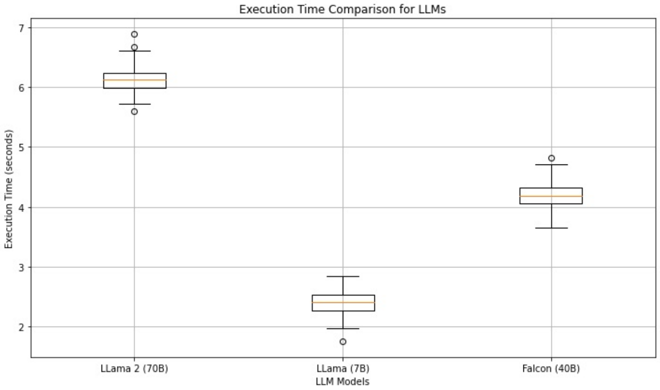


Fig. 5. Execution Time for each LLM

C. Correct Intention Translation Results

The accuracy of translating the original intent of the input sentences into actionable Ansible commands was evaluated by three cybersecurity experts. They graded each successfully executed command on a 1-to-5 scale, where 1 represented a completely incorrect translation, and 5 signified a perfect translation, then we computed the average score of the evaluations. The following average scores were observed:

- **Llama 2 (70B):** 4.5/5
- **Llama (7B):** 3.8/5
- **Falcon (40B):** 4.2/5

These scores reflect the models' ability to understand and correctly interpret the intentions behind the mitigation actions described in the input sentences.

D. Overall Performance Evaluation

The LLM models were evaluated on success rate, iterations till success, execution time, and correct intention translation. Llama 2 (70B) had the highest success rate and the highest number of average iterations, indicating that it can adapt to feedback and overcome previous failures, while Llama (7B) was the most time-efficient, probably due to its smaller size. Llama (7B) showed the lowest performance but still provided a reasonable balance between speed and success.

In terms of correct intention translation, Llama 2 (70B) led, followed closely by Falcon (40B). Overall, Llama 2 (70B) and Falcon (40B) were the most efficient models, while Llama (7B) offered a trade-off between speed and accuracy.

V. Conclusion and Future Work

This study highlights the potential of leveraging AI, specifically LLMs, to enhance the security of next-generation networks. Our findings suggest that LLMs are not only applicable to traditional use cases but also hold promises in cybersecurity applications, particularly for automating tasks like Ansible command generation. The use of LLM agents has demonstrated the ability to streamline workflows and significantly reduce the need for human supervision. Moving forward, the proposed workflow can be tested in more complex network environments to assess its scalability and effectiveness. Additionally, incorporating a larger number of LLM agents could provide more detailed feedback and improve collaboration in generating commands more efficiently. Future research could focus on fine-tuning LLMs to improve both the success rates and accuracy of generated Ansible commands, or it could shift toward enhancing feedback mechanisms and broadening the range of supported mitigation actions, thereby strengthening our automated cybersecurity response framework.

VI. Acknowledgment

The authors acknowledge the financial support of the Smart Networks and Services Joint Undertaking (SNS JU) under the European Union's Horizon Europe research and innovation programme under Grant Agreement No 101096342. This work has received funding from the Swiss State Secretariat for Education, Research and Innovation (SERI).

References

[1] E. Rodriguez, X. Masip-Bruin, J. Martrat, R. Diaz, A. Jukan, F. Granelli, P. Trakadas, and G. Xilouris, "A security services management architecture toward resilient 6g wireless and computing ecosystems," *IEEE access*, 2024.

[2] A. I. Salameh and M. El Tarhuni, "From 5g to 6g—challenges, technologies, and applications," *Future Internet*, vol. 14, no. 4, p. 117, 2022.

[3] V. Ziegler, P. Schneider, H. Viswanathan, M. Montag, S. Kanugovi, and A. Rezaki, "Security and trust in the 6g era," *Ieee Access*, vol. 9, pp. 142 314–142 327, 2021.

[4] J. Zhang, H. Bu, H. Wen, Y. Chen, L. Li, and H. Zhu, "When llms meet cybersecurity: A systematic literature review," *arXiv preprint arXiv:2405.03644*, 2024.

[5] Y. Yao, J. Duan, K. Xu, Y. Cai, Z. Sun, and Y. Zhang, "A survey on large language model (llm) security and privacy: The good, the bad, and the ugly," *High-Confidence Computing*, p. 100211, 2024.

[6] M. A. Ferrag, F. Alwahedi, A. Battah, B. Cherif, A. Mechri, and N. Tihanyi, "Generative ai and large language models for cyber security: All insights you need," *arXiv preprint arXiv:2405.12750*, 2024.

[7] H. project, "Horse project." [Online]. Available: https://horse-6g.eu/

[8] M. Pons, E. Valenzuela, B. Rodríguez, J. A. Nolazco-Flores, and C. Del-Valle-Soto, "Utilization of 5g technologies in iot applications: Current limitations by interference and network optimization difficulties—a review," *Sensors*, vol. 23, no. 8, p. 3876, 2023.

[9] F. Granelli, M. Qaisi, P. Kapsalis, P. Gkonis, N. Nomikos, I. Zacarias, A. Jukan, and P. Trakadas, "Ai/ml-assisted threat detection and mitigation in 6g networks with digital twins: The horse approach," in *2024 IEEE 29th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. IEEE, 2024, pp. 1–6.

[10] B. Al-Sada, A. Sadighian, and G. Oligeri, "MITRE ATT&CK: State of the Art and Way Forward," *ACM Computing Surveys*, vol. 57, no. 1, pp. 1–37, Jan. 2025. [Online]. Available: https://dl.acm.org/doi/10.1145/3687300

[11] A. Georgiadou, S. Mouzakitis, and D. Askounis, "Assessing MITRE ATT&CK Risk Using a Cyber-Security Culture Framework," *Sensors*, vol. 21, no. 9, p. 3267, May 2021. [Online]. Available: https://www.mdpi.com/1424-8220/21/9/3267

[12] P. Rajesh, M. Alam, M. Tahernezhadi, A. Monika, and G. Chanakya, "Analysis Of Cyber Threat Detection And Emulation Using MITRE Attack Framework," in *2022 International Conference on Intelligent Data Science Technologies and Applications (IDSTA)*. San Antonio, TX, USA: IEEE, Sep. 2022, pp. 4–12. [Online]. Available: https://ieeexplore.ieee.org/document/9923170/

[13] R. Rohan, S. Funilkul, D. Pal, and H. Thapliyal, "Humans in the Loop: Cybersecurity Aspects in the Consumer IoT Context," *IEEE Consumer Electronics Magazine*, vol. 11, no. 4, pp. 78–84, Jul. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9495235/

[14] A. Karunamurthy, R. Kiruthivasan, and S. Gauthamkrishna, "Human-in-the-Loop Intelligence: Advancing AI-Centric Cybersecurity for the Future," *Quing: International Journal of Multidisciplinary Scientific Research and Development*, vol. 2, no. 3, pp. 20–43, Sep. 2023.

[15] M. Danousis, K. Kaltakis, A. Dimos, C. Skianis, E. Kafetzakis, and I. Giannoulakis, "Optimizing network cybersecurity: Ai-powered nlp for natural language command interpretation," in *2024 IEEE 29th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2024, pp. 1–7.

[16] Y. Ishibashi and Y. Nishimura, "Self-Organized Agents: A LLM Multi-Agent Framework toward Ultra Large-Scale Code Generation and Optimization," Apr. 2024, arXiv:2404.02183 [cs]. [Online]. Available: http://arxiv.org/abs/2404.02183

[17] S. Kwon, S. Lee, T. Kim, D. Ryu, and J. Baik, "Exploring LLM-based Automated Repairing of Ansible Script in Edge-Cloud Infrastructures," *Journal of Web Engineering*, Dec. 2023.

[18] J. Wang and Y. Chen, "A Review on Code Generation with LLMs: Application and Evaluation," in *2023 IEEE International Conference on Medical Artificial Intelligence (MedAI)*. Beijing, China: IEEE, Nov. 2023, pp. 284–289. [Online]. Available: https://ieeexplore.ieee.org/document/10403378/

[19] O. Topsakal and T. C. Akinci, "Creating Large Language Model Applications Utilizing LangChain: A Primer on Developing LLM Apps Fast," *International Conference on Applied Engineering and Natural Sciences*, vol. 1, no. 1, pp. 1050–1056, Jul. 2023. [Online]. Available: https://as-proceeding.com/index.php/icaens/article/view/1127