INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Assessing the Effectiveness of Large Language Models in Automated Threat Modeling

ANA ISABEL MOURA BATISTA
julho de 2025

POLITÉCNICO
DO PORTO

ISEP INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO

Assessing the Effectiveness of Large Language Models in Automated Threat Modeling

Ana Batista

MSc in Computer Engineering,
Specialisation Area of Cybersecurity And Systems Administration

Advisors: Prof. Nuno Pereira and Prof. Pedro Pinto, ISEP
Supervisor: Marco Rodrigues, INEGI

Porto, July 4, 2025

Statement of Integrity

I hereby declare having conducted this academic work with integrity.

I have not plagiarised or applied any form of undue use of information or falsification of results along the process leading to its elaboration.

Therefore the work presented in this document is original and authored by me, having not previously been used for any other end.

I further declare that I have fully acknowledged the Code of Ethical Conduct of P.PORTO.

ISEP, Porto, July 4, 2025

Abstract

As cyberattacks become more frequent, Threat Modeling has emerged as an essential component of software security practices. Traditionally, Threat Modeling is an intensive process, relying on experts to identify and evaluate risks within a system, which limits its adoption. The advent of Large Language Models (LLMs) presents an opportunity to automate this process. However, the successful application of these models in Threat Modeling requires careful prompt engineering and a rigorous strategy to assess the generated threat scenarios.

The project investigates this applicability, centering on a case study involving the Institute of Science and Innovation in Mechanical and Industrial Engineering (INEGI) SUNDIAL application. Using STRIDE GPT as the tool for threat models generation, four prompting techniques were studied and applied: STRIDE GPT's Initial Prompt, Chain of Thought (CoT), Negative-Only Few-Shot, and the combined NO-Few-Shot-CoT, across three LLMs.

A Threat Model Evaluation Tool, TMEval, is proposed to enable focused comparison of identified STRIDE threats by LLMs against those in the ground truth for a specific application, employing four metrics: BLEU, ROUGE, BERTScore, and LLM-as-a-Judge. The emphasis is on the LLM-as-a-Judge approach across five dimensions: consistency, plausibility, and coverage of targets, weaknesses, and attack vectors.

The results show that any LLM with a specific prompting strategy does not produce scenarios consistent with the ground truth across all threat categories, suggesting that performance depends on the category and the application context provided. For the case study, the NO-Few-Shot-CoT prompting approach demonstrated the highest effectiveness across most categories.

Keywords: Threat Modeling, Prompt Engineering, STRIDE, Artificial Intelligence, Large Language Models, LLM-as-a-Judge

Resumo

À medida que os ciberataques se tornam mais frequentes, a Modelação de Ameaças emergiu como um componente essencial das práticas de segurança de software. Tradicionalmente, a modelação de ameaças é um processo intensivo, dependente de especialistas para identificar e avaliar os riscos de um sistema, o que limita a sua adoção. O surgimento dos Modelos de Linguagem de Grande Escala (LLMs) apresenta uma oportunidade para automatizar este processo. No entanto, a aplicação bem-sucedida destes modelos na modelação de ameaças exige uma engenharia de prompts cuidadosa e uma estratégia rigorosa para avaliar os cenários de ameaça gerados.

Este projeto investiga essa aplicabilidade, centrando-se num estudo de caso que envolve a aplicação SUNDIAL do Instituto de Ciência e Inovação em Engenharia Mecânica e Engenharia Industrial (INEGI). Utilizando o STRIDE GPT como ferramenta para a geração dos modelos de ameaça, foram estudadas e aplicadas quatro técnicas de prompting: o Prompt Inicial do STRIDE GPT, Chain-of-Thought (CoT), Negative-Only Few-Shot e a combinação NO-Few-Shot-CoT, em três LLMs distintos.

É proposta uma ferramenta de avaliação de modelos de ameaça, a TMEval, que permite a comparação focada entre as ameaças STRIDE identificadas pelos LLMs e aquelas presentes no ground truth de uma aplicação específica, recorrendo a quatro métricas: BLEU, ROUGE, BERTScore e LLM-as-a-Judge. A ênfase recai sobre a abordagem LLM-as-a-Judge, avaliada em cinco dimensões: consistência, plausibilidade e cobertura de alvos, vulnerabilidades e vetores de ataque.

Os resultados demonstram que nenhum LLM com uma estratégia de prompting específica produz cenários consistentes com o ground truth em todas as categorias de ameaça, sugerindo que o desempenho depende da categoria e do contexto da aplicação fornecido. No estudo de caso, a abordagem NO-Few-Shot-CoT revelou a maior eficácia na maioria das categorias.

Contents

List of Figures **xi**

List of Tables **xiii**

List of Acronyms **xvii**

1 Introduction **1**
1.1 Objectives and Research Questions . . . 2
1.2 Contributions . . . 2
1.3 Ethical Considerations . . . 3
1.4 Structure . . . 3

2 Background and State of the Art **5**
2.1 Threat Identification and STRIDE Categorization . . . 5
2.2 The Potential and Limitations of LLMs in Identifying Threats . . . 6
2.3 Literature Review . . . 7
2.4 RQ1: The Role and Importance of Threat Modeling as an Ongoing Development Process . . . 8
2.4.1 Keywords, Selection Criteria, and PRISMA Workflow . . . 8
2.4.2 RQ1: When and How Should Threat Modeling Be Performed? . . . 9
2.5 RQ2: The Adoption of LLMs for Generating Threat Models . . . 10
2.5.1 Keywords, Selection Criteria, and PRISMA Workflow . . . 10
2.5.2 RQ2.1: What Motivates the Use of LLMs for This Task? . . . 12
2.5.3 RQ2.2: Were Any Techniques Used to Boost the Performance of LLMs on This Task? . . . 13
2.6 RQ3: Replacing Traditional Metrics With the LLM-as-a-Judge Evaluation Strategy . . . 15
2.6.1 Keywords, Selection Criteria, and PRISMA Workflow . . . 15
2.6.2 RQ3.1: What Are the Limitations of Traditional Evaluation Metrics? 16
2.6.3 RQ3.2: What Are the Advantages of the Innovative LLM-as-a-Judge Approach? . . . 17

3 Case Study: Sundial Threat Model **19**
3.1 External Dependencies . . . 19
3.2 Entry Points and Trust Levels . . . 20
3.3 Assets . . . 22
3.4 Threat Modeling . . . 22

4 Methodology and Experiment Design **25**
4.1 Reference Data Creation . . . 25
4.2 LLM-based Threat Modeling Tool . . . 26

4.3 Prompt Optimization 27
4.4 Evaluation Process 29
4.5 Evaluation Metrics 29

5 Implementation and Evaluation **33**
5.1 TMEval and Metrics Configuration 33
5.2 Evaluation 34
5.2.1 Initial Prompt 34
5.2.2 Chain-of-Thought Prompting 39
5.2.3 Negative-Only Few-Shot Prompting 41
5.2.4 NO-Few-Shot-CoT Prompting 43
5.3 Discussion 45
5.3.1 Models Performance 45
5.3.2 Prompting Strategies Effectiveness 45
5.3.3 Performance Variation Across STRIDE Categories 46
5.3.4 Overall Synthesis 46

6 Conclusions **49**
6.1 Objectives Accomplished 49
6.2 Contributions 50
6.3 Future Work 50

Bibliography **51**

Appendix A Common Dimensions Prompts **53**

Appendix B Spoofing Dimensions Prompts **55**

Appendix C Tampering Dimensions Prompts **59**

Appendix D Repudiation Dimensions Prompts **63**

Appendix E Information Disclosure Dimensions Prompts **67**

Appendix F Denial of Service Dimensions Prompts **71**

Appendix G Elevation of Privilege Dimensions Prompts **75**

Appendix H Ground Truth Threat Model **79**

List of Figures

1 Study Selection Process According to PRISMA for RQ1 9
2 Study Selection Process According to PRISMA for RQ2 12
3 Study Selection Process According to PRISMA for RQ3 16

4 SUNDIAL Data Flow Diagram 23

5 Methodology 25
6 Prompt Engineering Process 28
7 TMEval Framework 31

8 LLM-as-a-Judge Evaluation Example 36
9 LLM-as-a-Judge Results for the Initial Prompt 37
10 LLM-as-a-Judge Results for the Chain-of-Thought Prompt 39
11 LLM-as-a-Judge Results for the Few-Shot Prompt 41
12 LLM-as-a-Judge Results for the Few-Shot Prompt 43
13 Average Score in Each Category Over the Four Prompts 45

List of Tables

1 Mapping of STRIDE Framework Elements to the CIA Triad 6
2 Databases Included 7
3 Query and Selection Criteria (RQ1) 8
4 Query and Selection Criteria (RQ2) 11
5 Techniques for Enhancing Models' Performance 14
6 Query and Selection Criteria (RQ3) 15

7 External Dependencies of SUNDIAL 20
8 Entry Points and Trust Levels of SUNDIAL 21
9 Assets of SUNDIAL 22

10 BLEU, ROUGE, and BERTScore Results for the Initial Prompt 35
11 Interpretation of LLM-as-a-Judge Results for the Initial Prompt 38
12 Interpretation of LLM-as-a-Judge Results for the Chain-of-Thought Prompt 40
13 Interpretation of LLM-as-a-Judge Results for the Few-Shot Prompt 42
14 Interpretation of LLM-as-a-Judge Results for the NO-Few-Shot-CoT Prompt 44
15 Mapping of STRIDE Categories to the Best LLM and Prompting Strategy 47

16 SUNDIAL Threat Model 79

Listings

1 STRIDE GPT Context . . . 26
2 STRIDE GPT Initial Prompt . . . 27
3 Chain-of-Thought Prompt . . . 28
4 Negative-Only Few-Shot Prompt . . . 29
5 Dimensions Prompt Structure . . . 30
6 Consistency Prompt . . . 53
7 Plausibility Prompt . . . 54
8 Entity Coverage Prompt . . . 55
9 Authentication Gaps Coverage Prompt . . . 56
10 Attack Vectors Coverage Prompt . . . 57
11 Asset Coverage Prompt . . . 59
12 Integrity Gaps Coverage Prompt . . . 60
13 Tampering Methods Coverage Prompt . . . 61
14 Action Coverage Prompt . . . 63
15 Logging Gaps Coverage Prompt . . . 64
16 Attack Vectors Coverage Prompt . . . 65
17 Data Coverage Prompt . . . 67
18 Protection Gaps Coverage Prompt . . . 68
19 Attack Methods Coverage Prompt . . . 69
20 Resource Coverage Prompt . . . 71
21 Protection Gaps Coverage Prompt . . . 72
22 Attack Types Coverage Prompt . . . 73
23 Vulnerability Point Coverage Prompt . . . 75
24 Control Gaps Coverage Prompt . . . 76
25 Exploit Methods Coverage Prompt . . . 77

List of Acronyms

2FA	Two-Factor Authentication.
AI	Artificial Intelligence.
API	Application Programming Interface.
CAPEC	Common Attack Pattern Enumeration and Classification.
CI/CD	Continuous Integration and Continuous Delivery/Deployment.
CIA	Confidentiality, Integrity, and Availability.
CoT	Chain of Thought.
CPU	Central Processing Unit.
CVEs	Common Vulnerabilities and Exposures.
DAST	Dynamic Application Security Testing.
DFD	Data Flow Diagram.
GPU	Graphics Processing Unit.
IEEE	Institute of Electrical and Electronics Engineers.
INEGI	Institute of Science and Innovation in Mechanical and Industrial Engineering.
IPP	Instituto Politécnico do Porto.
IPs	Internet Protocols.
JWT	JSON Web Token.
LCA	Life Cycle Assessment.
LLMs	Large Language Models.
MACM	Model for the Architecture of Cloud-based Microservices.
NIST	National Institute of Standards and Technology.
NLP	Natural Language Processing.
OPRO	Optimization by Prompting.
OWASP	Open Source Foundation for Application Security.

PRISMA	Preferred Reporting Items for Systematic Reviews and Meta-Analyses.
RAG	Retrieval Augmented Generation.
SAST	Static Application Security Testing.
SIEM	Security Information and Event Management.
SSDE	Secure Software Development Environment.
TMEval	Threat Model Evaluation Tool.
TMT	Threat Modeling Tool.
VPN	Virtual Private Network.

Chapter 1

Introduction

With the sophistication of cyberattacks threatening the fundamental principles of Confidentiality, Integrity, and Availability (CIA), it is critical to prioritize the identification and mitigation of vulnerabilities before exploitation occurs. This objective should be pursued not only by integrating security measures during the design phase (security-by-design) but also by maintaining security considerations throughout the entire software development lifecycle (Kreitz, 2020). The Open Source Foundation for Application Security (OWASP) underscores this necessity by advocating Threat Modeling as a continuous process to be conducted whenever new features are introduced, infrastructure changes are implemented, or security incidents arise (Drake, 2020). Through detailed analysis of an application's infrastructure and functionalities, potential threats that could compromise the system can be identified. These threats can then be systematically categorized using established frameworks such as STRIDE, enabling organizations to assess risk levels and implement effective countermeasures to prevent future security breaches (Sion et al., 2018). However, traditional Threat Modeling remains a time-consuming and expertise-intensive process.

The advent of Artificial Intelligence (AI), particularly Large Language Models (LLMs) exemplified by conversational agents such as OpenAI's ChatGPT and Google AI's Gemini, has introduced helpful tools across multiple domains. This progress has generated growing interest in exploring the potential of LLMs to assist with cybersecurity tasks. A survey by Tian et al., 2025 cites over 20 studies up to April 2025 demonstrating the practical application, benchmarking, and challenges of LLMs across virtually all phases of the cyberattack lifecycle and various cybersecurity tasks (Tian et al., 2025).

This work aims to expand the body of work by investigating the feasibility of leveraging LLMs for the generation of threat models.The main objective is to determine whether different LLMs vary in performance and whether refining the prompt can unlock greater potential through an LLM-as-a-Judge evaluation, thereby assessing if AI can make Threat Modeling more accessible to non-experts. The study focused on a specific case study: a Docker container-based application called SUNDIAL, developed by Institute of Science and Innovation in Mechanical and Industrial Engineering (INEGI). This application was analyzed and deconstructed to create a reference threat model, referred to as the ground truth.

The evaluation was supported by the development of Threat Model Evaluation Tool (TMEval), specifically designed for threats identification task, enabling the comparison of threat models generated by LLMs with a ground truth for a focused application. Implements four distinct metrics, including traditional ones like BLEU and ROUGE, as well as more recent and advanced methods such as BERTScore and LLM-as-a-Judge. The latter received the most emphasis, as it enables a more human-like assessment by allowing an LLM to score a threat model generated by another LLM on a scale from 1 to 5 based on predefined criteria, closely

mimicking human judgment. These criteria include plausibility, where the technical terms in the threat models are evaluated based on how plausible they are when compared to those in the ground truth; consistency, which assesses the overall similarity of the threats listed; and coverage of targets, weaknesses, and attacks, which provides a granular evaluation by verifying whether the threatened assets, exploited weaknesses, and types of attacks align with those in the ground truth, enabling a decomposed comparison.

1.1 Objectives and Research Questions

Aiming to support INEGI in understanding the security posture of its platform, and with the study focused on how LLMs can assist in this task, the objectives pursued within the practical scope are as follows:

1. **Define the ground truth for the case study**: Collect detailed information about the SUNDIAL platform and its underlying infrastructure. Conduct a thorough analysis of the application following established steps to identify potential security threats and formulate the ground truth STRIDE threat model.
2. **Use of LLMs to generate threat models**: Employ several LLM models and prompting techniques to generate threat models for the case study.
3. **Implement a systematic evaluation framework**: Adopt metrics to perform alternative types of evaluation of LLM-generated threat modeling, culminating in the development of a systematic evaluation framework for the task.
4. **Proceed to the evaluation**: Assess the use of LLMs in providing assistance with threat modeling. Compare the applicability of the metrics, models, and techniques.

Due to time constraints, mitigation strategies were beyond the scope of this study. Consequently, the focus is exclusively on threat identification and categorization, while automated mitigation measures and risk-based threat prioritization are planned for future research.

1.2 Contributions

The contributions of this work can be summarizes as follows:

- The creation of TMEval (Batista & Pereira, 2025) provides a tool that facilitates a more focused evaluation of threat models generated by LLMs, specifically for each STRIDE category, using well-researched, category-specific criteria that closely resemble those applied by human experts. The tool enables the definition of both the ground truth to be compared against and the generated threat models. Users can run one or multiple evaluation metrics, each with an interactive Plotly dashboard.
- The exploration of four different prompting strategies to investigate whether the phrasing and structure of prompts influence the quality of LLM-generated threat models. The prompting techniques used were: the default prompt from the STRIDE GPT tool, which follows standard steps to guide the LLM in producing a threat model; Chain-of-Thought (CoT), where guided steps inspired by OWASP's Threat Modeling recommendations (Conklin, 2025) direct the LLM through a structured reasoning process; Negative-Only Few-Shot, which provides generic threat examples to encourage the model to avoid them and tailor scenarios specifically to the application details; and NO-Few-Shot-CoT, which combines both approaches.

- The analyses demonstrated that earlier models, such as Gemini 1.5 Pro, struggle with understanding broader context and produce less detailed threat models compared to more recent models like Gemini 2.5 Pro and GPT-4.5, although no LLM or single prompting strategy consistently comes close to the manually created ground truth across all STRIDE categories; performance depends on the categories as well as the specific application under analysis. For the SUNDIAL case study, the NO-Few-Shot-CoT prompting strategy achieves the best performance in most categories.

1.3 Ethical Considerations

As a student of Instituto Politécnico do Porto (IPP), the following duties from the Institute's Code of Conduct have been adhered to throughout this document (IPP, 2020):

- As mentioned in Article 4, point o), the best practices of scientific research and ethical principles have been respected, ensuring compliance with copyright laws and guaranteeing proper citation and referencing of bibliographic sources.
- As stated in Article 6, point n) 2.8, when using ideas from third parties, these have been correctly cited and referenced.
- Article 6, point n) 2.11, ensures that only truthful results are presented.
- Article 8, point 1, is reflected in the Statement of Integrity.
- Article 10 emphasizes good practices in research activities, particularly with regard to the use of AI, as noted in point 2, which was used for addressing sudden uncertainties, providing spelling suggestions, and to support the objectives of the thesis.

As an additional consideration regarding the last point, it is important to reflect on the use of AI as the central focus of this thesis. Confidentiality and integrity concerns related to chatbots have led to a thorough review of all details regarding the INEGI application before sharing them with such models, in order to preserve the organization's privacy. Therefore, despite the advantages offered by these tools, they may pose a risk of violating ethical and cybersecurity principles and consequently undermine trust in research in this field (Chowdhury et al., 2023).

1.4 Structure

This document is organized into six chapters.

Chapter 1 introduces the project, providing its context and outlining the research problem, as well as relevant ethical considerations. It also defines the objectives and formulates the research questions that guide the study.

Chapter 2 presents the theoretical background, offering an overview of the concepts and motivations required to understand the foundations of the state of the art, which addresses three research questions through a systematic literature review, conducted in accordance with the PRISMA methodology, in order to identify and synthesize relevant findings.

Chapter 3 describes the application that serves as the case study for this project.

Chapters 4 and 5 cover the methodology and the implementation/evaluation phases, respectively, including a discussion of the results obtained.

Finally, Chapter 6 presents the conclusions of the work. Reflects on the results, highlights the main contributions, and outlines potential future work and research directions within this domain.

Chapter 2

Background and State of the Art

In this chapter, the aim is to lay the foundation of key concepts. By presenting Threat Modeling alongside the essential aspects of LLMs, the objective is to explore how both can be applied together and to underline how this combination supports the motivation behind the proposed theme, while also addressing the limitations. Furthermore, the chapter also undertakes a state-of-the-art literature review, structured according to the PRISMA methodology, to systematically address each research question.

2.1 Threat Identification and STRIDE Categorization

According to OWASP (Drake, 2020), a threat modeling process typically includes the following steps: 1) the description of the subject to be modeled, 2) assumptions that can be verified or challenged in the future as the threat landscape evolves, 3) the identification of potential threats to the system, 4) the definition of mitigation actions for each threat, and 5) a method for validating the model and threats, as well as verifying the success of the implemented actions.

Therefore, by understanding the application to be analyzed (1), it becomes possible to identify potential threats (3), which must be continuously assessed and adapted (2). A **weakness** in a **target** allows a threat to be realized through an **attack**, leading to risks/impacts such as the loss of confidentiality, where unauthorized individuals gain access to sensitive information; the loss of integrity, where data may be altered, rendering it unreliable or incorrect; and the loss of availability, when the system fails to ensure that information is accessible when needed (Honkaranta et al., 2021).

Table 1 provides an explanation and mapping of each STRIDE threat category to its corresponding impact on the CIA triad.

Table 1: Mapping of STRIDE Framework Elements to the CIA Triad. Adapted by the author from (Honkaranta et al., 2021)

Threat Name	**Explanation/Example**	**Relation to CIA: what is risked**
Spoofing	A malicious user uses an **attack method** to pretend to be someone else, targeting **spoofable entities** like a user's account by exploiting **authentication weaknesses**.	Confidentiality, Integrity
Tampering	The content within the system, a **critical asset vulnerable to unauthorized changes**, is altered by a malicious party's **attack method**, made possible by **weak or missing data integrity protections**.	Integrity
Repudiation	A user denies performing a **critical action prone to denial**; this **attack method** is successful due to **auditing or logging weaknesses**, such as the absence of a non-repudiable audit trail.	Integrity
Information Disclosure	**Sensitive data types** are exposed to unauthorized parties via an **attack method**. This exploits **weak or missing confidentiality controls**.	Confidentiality
Denial of Service	**Critical resources vulnerable to exhaustion** are made unavailable to a legitimate user by an **attack method**. This exploits **weak or missing DoS mitigations**.	Availability
Elevation of Privilege	A user applies an **attack method** to get more privileges than entitled to on **privileged resources or functionalities**, exploiting **missing or weak controls preventing escalation**.	Integrity, Confidentiality

2.2 The Potential and Limitations of LLMs in Identifying Threats

LLMs, such as those from the GPT and BERT families, represent a significant leap in Natural Language Processing (NLP), driven by the Transformer architecture. The core innovation of this architecture is its self-attention mechanism, which enables the model to weigh the importance of different words in a sequence. This allows it to capture long-range and complex contextual relationships, a capability directly relevant to analyzing detailed system descriptions in threat modeling (Ghosh et al., 2024).

LLMs are pre-trained on vast, internet-scale datasets. Own experiments have shown that their training data includes security-related knowledge, such as Common Vulnerabilities and Exposures (CVEs) (MITRE Corporation, 2025), and informations from cybersecurity organizations. As a result, they have shown the potential to synthesize this latent knowledge to suggest common threats, significantly accelerating the threat modeling process. However, not only are many of the CVEs provided by these models inaccurate, but they are also not

updated in accordance with the daily updates of CVEs. This means they are unable to keep up with new vulnerabilities and the rapidly evolving cybersecurity landscape.

With over 50,000 human evaluations, Vu et al., 2023 demonstrate that these models really present limitations when dealing with recent facts. In some cases, complementing what was mentioned about self-attention, their analysis also shows that both the amount of contextual information in the prompt and its ordering directly impact the quality of the model's responses (Vu et al., 2023). Concise and direct answers reduce the likelihood of hallucinations compared to longer, more verbose responses, an aspect that also influences the level of detail in STRIDE threat modeling.

2.3 Literature Review

The Preferred Reporting Items for Systematic Reviews and Meta-Analyses (PRISMA) methodology ensures rigor and transparency by providing a structured approach to the identification, screening, and analysis of relevant studies (Page et al., 2021). The review will explore: the significance of Threat Modeling (RQ1), the use of LLMs for its generation (RQ2), and the use of the LLM-as-a-judge strategy for LLM evaluation (RQ3) across papers published over the past five years.

- RQ1: What is the importance of adopting Threat Modeling as an ongoing development process? (Section 2.4).
- RQ2: What drives the use and advancement of LLMs for generating threat models instead of relying on manual approaches? (Section 2.5).
  - RQ2.1: What motivates the use of LLMs for this task?
  - RQ2.2: Were any techniques used to boost the performance of LLMs on this task?
- RQ3: Why has the LLM-as-a-Judge strategy been preferred for evaluating LLM cybersecurity outputs over other metrics? (Section 2.6).
  - RQ3.1: What are the limitations of traditional evaluation metrics?
  - RQ3.2: What are the advantages of the innovative LLM-as-a-judge approach?

To address each of the research questions, PRISMA includes key features aimed at identifying articles efficiently, such as predefined keywords and established inclusion and exclusion criteria.

Few prestigious digital libraries including Institute of Electrical and Electronics Engineers (IEEE), ScienceDirect, and arXiv are chosen (Table 2). They have large collections of scholarly publications and with most relevant ones to the current research domain.

Table 2: Databases Included

Database	**URL**
IEEE	https://ieeexplore.ieee.org/
ScienceDirect	https://www.sciencedirect.com/
arXiv	https://arxiv.org/

2.4 RQ1: The Role and Importance of Threat Modeling as an Ongoing Development Process

The answer to RQ1 (Subsection 2.4.2) is supported by the Subsection 2.4.1, which build upon the prior definition of keywords, inclusion and exclusion criteria, and the structured article search process outlined through the PRISMA workflow.

2.4.1 Keywords, Selection Criteria, and PRISMA Workflow

To better align the review with the proposed thesis case study, the criteria consider studies focused on applications hosted in containers. Additionally, only studies published between 2020 and 2025, written in English, peer-reviewed, and properly formatted are included. The keywords used are "DevSecOps" and "threat modeling", which are combined using search operators into a query and adapted for each database. Both the query and the selection criteria are presented in Table 3.

Table 3: Query and Selection Criteria (RQ1)

Search Query	**Searching in Databases (until April, 2025)**	**Inclusion Criteria**	**Exclusion Criteria**
DevSecOps AND "threat modeling"	**ScienceDirect** uses the "Title, abstract, or author-specified keywords" field. **IEEE** lacks this field and uses Command Search instead: *(("Abstract":DevSecOps OR "Document Title":DevSecOps OR "Author Keywords":DevSecOps) AND ("Abstract":"threat modeling" OR "Document Title":"threat modeling" OR "Author Keywords":"threat modeling"))* **ArXiv** does not offer any of these filtering options, so the search was conducted across all fields.	1) Studies from 2020–2025 2) Studies in English 3) Research focusing on the security of applications hosted in Docker containers and/or implemented as microservices	1) Studies not in English 2) Articles that are not peer-reviewed or lack proper formatting 3) Studies that do not address threat modeling for the security of container-based applications

Figure 1 illustrates the stages and number of papers processed for RQ1. The process began with the identification stage, where 13 papers were retrieved from academic databases using the tailored search query for each database and filtered by publication year (2020–2025). During full-text screening, 9 studies were excluded based on exclusion criterion 3, and 2 of the 13 were inaccessible. Consequently, a total of 11 articles were excluded, resulting in a final inclusion of 2 studies.

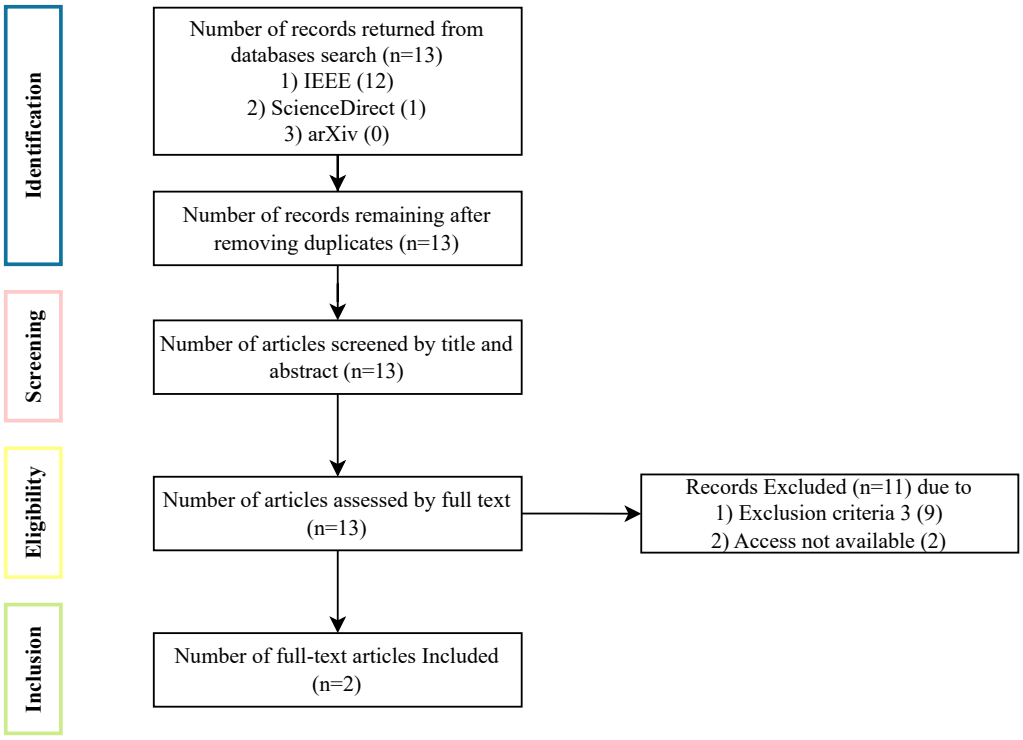


Figure 1: Study Selection Process According to PRISMA for RQ1

2.4.2 RQ1: When and How Should Threat Modeling Be Performed?

The study conducted by Kushwaha et al., 2024 considered the implementation of DevSecOps as a continuous monitoring approach. With a primary focus on the security of financial applications, security measures must be adopted at every stage of software development to prevent breaches. Using the Microsoft Threat Modeling Tool (TMT) during the planning phase, the potential vulnerabilities of a containerized banking application were analyzed. The tool identified 26 threats based on the STRIDE model — Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege. Along with the integration of Static Application Security Testing (SAST) and Dynamic Application Security Testing (DAST) within an automated Azure DevSecOps pipeline, the findings from the threat modeling served as essential initial guidelines for the secure development of the application. This ensures that security is prioritized from the outset and that only secure container images are deployed (Kushwaha et al., 2024).

The Secure Software Development Environment (SSDE) is a methodology that highlights the importance of integrating threat modeling phases with security testing, creating a continuous and aligned workflow between identified threats and executed tests from the early stages of the software development lifecycle (Casola et al., 2024). It begins with an initial identification of different asset types and data flows within the application, using tools such as Model for the Architecture of Cloud-based Microservices (MACM) instead of Data Flow Diagram (DFD), offering a more tailored approach for modern microservices, and container-based systems. This provides a clear view of critical points that need protection, leading

to the Threat Modeling process — categorizing threats and defining the necessary security controls.

Once this phase is completed, the next step is conducting security testing. The idea is: once the application is modeled and security controls are defined, SSDE enables security tests based on the identified threats' respective attack patterns (Common Attack Pattern Enumeration and Classification (CAPEC)). The methodology automates test execution whenever possible, leveraging tools such as MITRE CALDERA and Atomic Red Team, which emulate cyberattacks using ATT&CK framework techniques.

SSDE was tested in the context of the OWASP Juice Shop project, a vulnerable web application developed by OWASP for security testing purposes. The methodology involved threat modeling and security testing, identifying 44 threats across three main application components. This evaluation allowed for an assessment of the approach's effectiveness and ensured that vulnerabilities were properly addressed using security techniques, such as those from the ATT&CK framework.

Although, as studied, it is entirely indispensable at the beginning of application development, Threat Modeling is not a one-time process. Its updates are recommended after applications undergo the following events, according to OWASP: when a new feature is released, when a security incident occurs, and when changes in architecture or infrastructure arise (Drake, 2020).

2.5 RQ2: The Adoption of LLMs for Generating Threat Models

The following subsections contribute to the answer to RQ2, preceded by the identification of keywords and inclusion/exclusion criteria, as well as the article search conducted and documented using the PRISMA workflow.

2.5.1 Keywords, Selection Criteria, and PRISMA Workflow

Table 4 presents the query used, employing Boolean search operators such as "AND" and "OR" to combine the keywords "LLM" and "threat modeling", and refine the search within the selected databases. The table also includes the inclusion and exclusion criteria; only studies from the past five years are considered, given the recent emergence of LLM-based chatbots. Furthermore, studies will be excluded if they lack structure and proper referencing, are not written in English, or do not focus on supporting threat identification through AI.

Table 4: Query and Selection Criteria (RQ2)

Search Query	**Searching in Databases (until April, 2025)**	**Inclusion Criteria**	**Exclusion Criteria**
LLM AND "threat modeling"	**ScienceDirect** uses the "Title, abstract, or author-specified keywords" field. **IEEE** lacks this field and uses Command Search instead: *(("Abstract":docker OR "Document Title":LLM OR "Author Keywords":LLM) AND ("Abstract":"threat modeling" OR "Document Title":"threat modeling" OR "Author Keywords":"threat modeling"))* **ArXiv** does not offer any of these filtering options, so the search was conducted across all fields.	1) Studies from 2020–2025, since it is still an emerging area 2) Studies in English 3) Studies on the use of Artificial Intelligence to support and/or automate threat modeling	1) Studies not in English 2) Articles that are not peer-reviewed or lack proper formatting 3) Studies without a focus on threats identification and their generation using LLMs

Figure 2 presents the stages and number of papers processed in this systematic review. The process began with the identification stage, where 70 papers were retrieved from academic databases using the search query defined, along with filters such as publication year (2020–2025) and keyword presence in the title, abstract, or author-defined fields. A full-text screening of these 70 articles led to the exclusion of studies based on exclusion criterion 3: studies addressing cybersecurity in general without a specific focus on threat models or their generation through LLMs. As a result, 68 articles were excluded. During the inclusion stage, a final selection of 2 studies was made, all of which demonstrated high relevance to the research objectives. As part of the analysis of these articles, a backward citation search was conducted, resulting in four additional relevant sources.

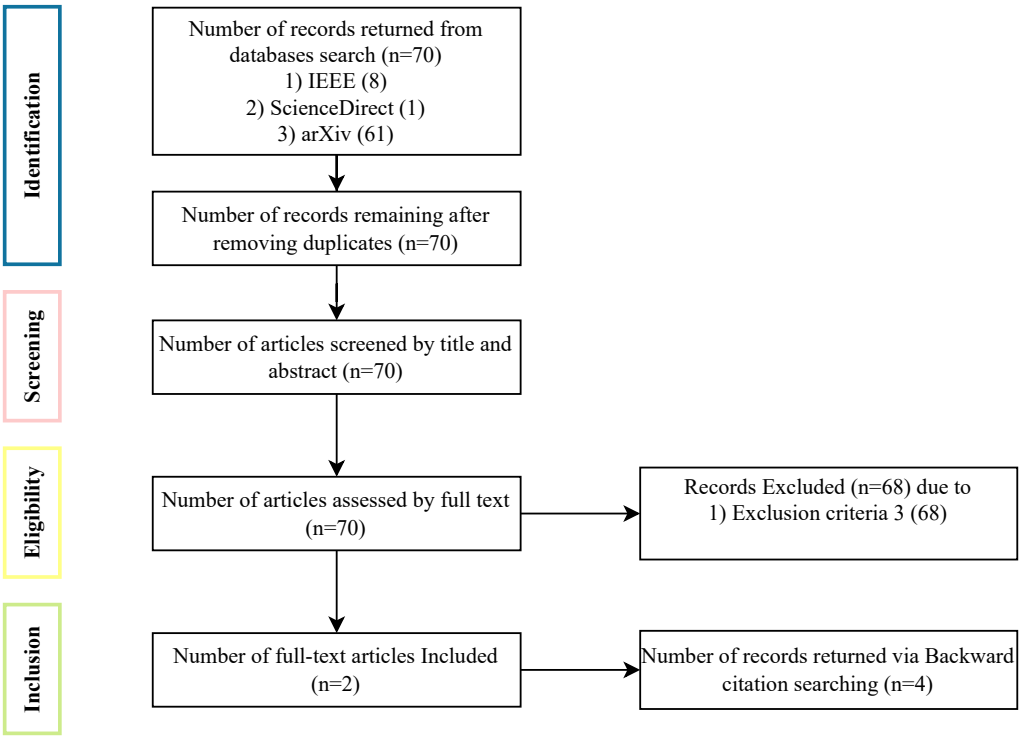


Figure 2: Study Selection Process According to PRISMA for RQ2

2.5.2 RQ2.1: What Motivates the Use of LLMs for This Task?

As LLMs continue to advance, more industries are leveraging this technology to tackle domain-specific challenges. However, there has been limited research exploring the effectiveness of LLMs in addressing the challenges of Threat Modeling (Kwarteng et al., 2024).

STRIDE GPT (Adams & Shibata, 2024) is a developed project designed by Matt Adams designed to harness the capabilities of a LLM for cybersecurity applications. It is used to generate an initial threat model, with Microsoft's STRIDE. Incorporating this tool involves analyzing a web application, breaking it down into its individual components and services, and leveraging AI to automatically assess the potential impact of each category on the overall application, based on the available information. STRIDE GPT can make use of existing AI models, including the latest GPT models for improved output quality, by communicating with the OpenAI API in JSON format to increase reliability, or even through integration with the Azure OpenAI Service. One issue with this project, as analyzed by Wu et al., 2025, is that it may make errors in organizing threats into the correct categories (Wu et al., 2025).

Cyber Sentinel (Kaheh et al., 2023) emerges as a conversational agent designed to optimize cybersecurity. It is not only aimed at assisting in Threat Modeling but also supports various other cybersecurity operations, with the primary goal of contributing to a more adaptable security posture. By utilizing its cyber threat intelligence module, it can analyze real-time security alerts from a Security Information and Event Management (SIEM), suggest practices, and manage security policies. Transforms conversations into an interactive and collaborative

method for discussing and sharing knowledge with security analysts, while also raising awareness among employees. This agent, although highly adaptable to new threats, struggles to provide effective mitigation strategies (Wu et al., 2025).

ThreatModeling-LLM is a model created with the same objective of automating threat identification and mitigation suggestions, specifically for banking systems. The automation of this process became crucial to mitigate human errors resulting from traditional manual methods. ThreatModeling-LLM was developed to address this issue, overcoming the limitations of STRIDE GPT and Cyber Sentinel and the gaps in LLMs in threat modeling for banking systems. To significantly improve the performance of this LLM in threat modeling tasks for financial systems, the following steps were undertaken: prompt engineering to enhance prediction accuracy and fine-tuning of existing models through a benchmark for training, specifically adapted to the threats in the banking sector, while meeting the confidentiality, integrity, and privacy requirements by directly mapping threats to security standards such as National Institute of Standards and Technology (NIST) and, consequently, efficiently suggesting appropriate security controls from the standard (Wu et al., 2025).

To ensure the validity of the data, it was created using reliable and industry-recognized tools by the author, such as the TMT. After creating a DFD in real-world scenarios, the TMT was used to identify potential threats, followed by the suggestion of mitigations for each identified threat and mapping them to security control codes (NIST 800-53). The threats generated by the TMT were manually verified by experts to ensure accuracy and applicability to the banking system before being used for training the threat modeling model. Although initially designed for the banking sector, ThreatModeling-LLM is adaptable to various industries through dataset customization and model retraining, making it a versatile cybersecurity tool.

Similarly, the study conducted by Kwarteng et al. aimed to fine-tune and optimize the Llama2 model for Threat Modeling in the domain of modern medical devices – the CyberLlama2 model. This study was among the first to integrate training datasets covering cybersecurity, privacy, and safety (Kwarteng et al., 2024).

Many steps remain to be taken before these AI systems are ready to assume the role of security analysts independently.

2.5.3 RQ2.2: Were Any Techniques Used to Boost the Performance of LLMs on This Task?

As demonstrated by the authors of ThreatModeling-LLM, the integration of fine-tuning and prompt engineering strategies significantly improves the performance of LLMs. Fine-tuning is essential, as the models are not initially equipped to answer cybersecurity-related questions (Kwarteng et al., 2024), while prompt engineering addresses how the questions are formulated. Whether complex or ambiguous, the formulation of questions can influence the model's ability to interpret them effectively (Kaheh et al., 2023). Table 5 summarizes the techniques applied for Cyber Sentinel, ThreatModeling-LLM, and CyberLlama2 discussed in the previous subsection.

The authors employed several prompt engineering techniques, particularly in Cyber Sentinel and ThreatModeling-LLM, to enhance model performance. Both adopted the approach of combining Chain of Thought (CoT) prompting with few-shot learning:

- **Chain-of-Thought Prompting**: What are the reasoning steps the model can follow to reach the output? (Wei et al., 2023)
- **Few-Shot Learning**: Providing the model with examples as additional context beyond its existing knowledge on the subject (Brown et al., 2020).

In addition, ThreatModeling-LLM combined prompt engineering with fine-tuning to better tailor the model to the specific demands of banking systems, while CyberLlama2 relied primarily on fine-tuning to adapt the model to the medical domain. ThreatModeling-LLM demonstrated that the combination of prompt engineering and fine-tuning yielded the most effective results. STRIDE GPT is expected to represent a subject of study, as the specific strategies it employs remain unclear and require further investigation.

Table 5: Techniques for Enhancing Models' Performance

	Cyber Sentinel (GPT-4 models)	**ThreatModeling-LLM (Llama-3.1-8B and GPT-3.5-turbo models)**	**CyberLlama2 (Llama2 model optimization)**
Objective	Serve as an intelligent conversational assistant to support and validate threat models, among other cybersecurity tasks	Automate the threat modeling process in banking systems based on the STRIDE framework and NIST 800-53 control codes	Automate the threat modeling process in the Medical Field (MMD) based on the MEDICALHARM methodology
No techniques applied	An initial prompt is given to the model to identify the type of request made by the user	Values of metrics: Accuracy (0.17), Precision (0.35), and Recall (0.27)	Llama2 baseline model was evaluated using metrics such as ROUGE, MAUVE, and METEOR
Fine-tuning	Not applicable	Through fine-tuning, the models achieved over 60% accuracy, and the other metrics (precision, recall, and text similarity with BERT) also improved	Fine-tuning the Llama2 model with data from various cybersecurity sources. 2,000 entries were tested, achieving better results in most metrics compared to the original Llama2 model
Zero-shot prompting	Not applicable	Not applicable	Enabling the model to respond to new questions based on previously acquired knowledge
Chain-of-Thought (CoT) + Few-Shot Learning	Through these techniques, the model provides additional context in steps	Accuracy and Recall increased, showing that articulated reasoning processes with examples help improve the identification of threats	Not applicable
Optimization by Prompting (OPRO)	Not applicable	Precision achieves moderate improvements, but does not reach CoT's overall performance	Not applicable
CoT + OPRO	Not applicable	Achieved the highest scores in all metrics. Precision and Recall almost reached 0.6, and Text Similarity, measured with BERT, exceeded 0.95	Not applicable
Self-Consistency	GPT-4 generates approaches and compares them to choose the most appropriate one	Not applicable	Not applicable

2.6 RQ3: Replacing Traditional Metrics With the LLM-as-a-Judge Evaluation Strategy

Similarly, for RQ3, the definition of keywords and the documentation of the identification, screening, eligibility, and inclusion processes are also conducted.

2.6.1 Keywords, Selection Criteria, and PRISMA Workflow

For RQ3, two queries were used: "LLM judge AND security" and "LLM metrics AND cybersecurity". Based on the retrieved articles, the following inclusion criteria were established: studies published between 2020 and 2025, written in English, and applying the LLM-as-a-Judge strategy to cybersecurity tasks involving LLMs. Papers that are not peer-reviewed or lack proper formatting are excluded (Table 6).

Table 6: Query and Selection Criteria (RQ3)

Search Queries	**Searching in Databases (until April, 2025)**	**Inclusion Criteria**	**Exclusion Criteria**
LLM judge AND security LLM metrics AND cyber-security	**ScienceDirect** uses the "Title, abstract, or author-specified keywords" field. **IEEE** lacks this field and uses Command Search instead: e.g.:*(("Abstract":LLM metrics OR "Document Title":LLM metrics OR "Author Keywords":LLM metrics) AND ("Abstract":cybersecurity OR "Document Title":cybersecurity OR "Author Keywords":cybersecurity))* **ArXiv** does not offer any of these filtering options, so the search was conducted across all fields.	1) Studies from 2020–2025, since it is still an emerging area 2) Studies in English 3) Research on AI-Driven cybersecurity tasks with evaluation via LLM-as-a-judge	1) Studies not in English 2) Articles that are not peer-reviewed or lack proper formatting 3) Studies not focused on LLM-based cybersecurity and LLM-as-a-judge evaluation

Figure 3 summarizes the systematic review process. Of 81 initially retrieved papers, filtered by year (2020–2025) and relevant fields, 79 were excluded after full-text screening based on criterion 3. In the end, 2 highly relevant studies were included. Moreover, through citation searching of these articles, three more were added to provide a better background.

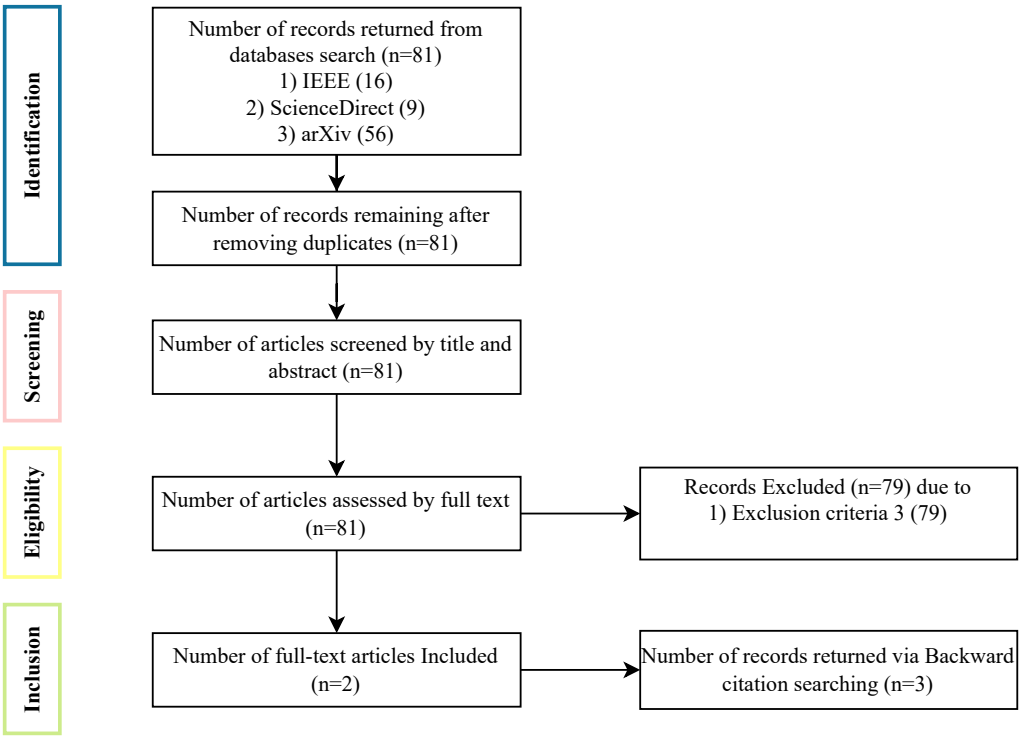


Figure 3: Study Selection Process According to PRISMA for RQ3

2.6.2 RQ3.1: What Are the Limitations of Traditional Evaluation Metrics?

As presented in Table 5 in Subsection 2.5.3, the LLMs evaluation metrics adopted across studies range from traditional approaches, such as ROUGE (Lin, 2004) and BLEU (Papineni et al., 2002), to LLM-enhanced methods, including semantic similarity metrics like BERTScore, complemented by expert evaluations.

A closer examination of these metrics reveals that many may fall short in effectively capturing the specific requirements of the threat model generation task. Traditional metrics primarily assess lexical similarity, failing to account for contextual meaning, an essential factor in tasks of this nature. BERTScore approach represents a significant improvement, as noted by Liu et al., yet still presents certain limitations. Therefore, careful selection and comparison of evaluation metrics becomes a matter of critical importance (Liu et al., 2023).

G-Eval, introduced by Liu et al., 2023, acts as an LLM-as-a-Judge that harnesses the capabilities of LLMs to evaluate outputs generated by other AI systems. When provided with a clear explanation of the task and its evaluation criteria (dimensions) through a well-crafted user-defined prompt, the LLM generates a detailed evaluation process, an automatic Chain-of-Thought, and completes a structured scoring form for each criterion, ultimately producing a final score based on probabilistic reasoning.

2.6.3 RQ3.2: What Are the Advantages of the Innovative LLM-as-a-Judge Approach?

With the goal of developing a LLM-question-answering system focused on cybersecurity, specifically cyberattacks, based on entries from the MITRE ATT&CK framework using a Retrieval Augmented Generation (RAG) approach, the authors employed a LLM-as-a-judge methodology to evaluate the system's performance (Krishna, 2024). They argued that this evaluation method is more appropriate than traditional metrics such as BLEU or ROUGE, which fail to assess semantic correctness. The judging model was responsible for scoring the quality of each generated response against a reference answer (ground truth) on a scale from 0 to 1, accompanied by a textual justification. For this evaluation, the authors used the DeepEval tool, which relies on the G-Eval framework, as previously discussed.

To capture different levels of answer quality, the evaluation incorporated two complementary metrics: hard correctness and soft correctness. Hard correctness required that the generated response includes only accurate and relevant information, penalizing any hallucinations, contradictions, omissions, or irrelevant content. In contrast, soft correctness was more permissive; it acknowledged responses as correct when the model explicitly recognized that it lacked sufficient context to provide an answer, thus rewarding appropriate uncertainty and avoidance of speculation.

Using this dual-criteria approach, the authors found that fine-tuned open-source models consistently outperformed others, producing more accurate and reliable responses, especially under strict correctness conditions.

Another study focused on enhancing the security of digital railway infrastructures led to the development of a system also based on the use of LLMs, designed to assist non-experts in following railway safety regulations. In addition to manual evaluation by human experts, the LLM-as-a-Judge approach was also employed. The Llama-3.1-405B-Instruct model was used as the judge to assess the following criteria: correctness, reasoning ability, and hallucination detection. The evaluations were compared across different models using a structured prompt (Bolton et al., 2025).

Chapter 3

Case Study: Sundial Threat Model

Based on the initial steps of the OWASP Threat Modeling methodology (Conklin, 2025), the application will be presented through a description and decomposition, facilitating its understanding and contributing to the creation of the ground truth threat model (Appendix H). This description will also serve as input for the LLMs, as further analyzed in Section 4.2.

3.1 External Dependencies

The SUNDIAL platform was created to provide a unified solution for managing sustainability data across multiple projects within the organization, functioning as a separate application for each one. Its main objective is to ensure that data collection, storage, and analysis follow a standardized format, facilitating the comparison and integration of information across different projects. Instead of each initiative developing its system, SUNDIAL offers a common and configurable environment, enabling the efficient organization of large volumes of structured data, tracking activity history, and supporting the environmental, economic, and social assessment of initiatives. RELiEF is one of the projects where the platform is used to organize and analyze data related to lithium recovery.

The application is running on a Linux server with Apache. This application is divided into Docker containers, which are set up in a separate network within the machine, created exclusively for this purpose. Containers for the frontend, backend, and database have different Internet Protocols (IPs) and ports, and they are built and distributed by Jenkins. The frontend, React 17.0.2, a JavaScript library, and the backend, Node.js 14.18, vary for each project, while the database server, MySQL 8.0.32, is the same for all projects hosted by SUNDIAL. The entire web server is protected by a firewall and a proxy, which then forwards traffic to the web containers. These external dependencies are also outlined in the Table below.

Table 7: External Dependencies of SUNDIAL

Id	Description
1	The application runs on a remote Ubuntu server with Apache. The application's functionality depends on the operating system and the updates and patches provided by its distribution.
2	The application is divided into Docker containers within the server. Jenkins is used for building Docker images and automating the Continuous Integration and Continuous Delivery/Deployment (CI/CD) pipeline.
3	React for the frontend, and Node.js for the backend, are third-party technologies on which the application depends to function correctly.
4	The application uses MySQL to store and manage data.
5	The firewall controls network traffic between the application and external networks, allowing access only from the INEGI network.
6	The proxy acts as an intermediary for managing network traffic.

3.2 Entry Points and Trust Levels

Access to the SUNDIAL platform is granted through a Virtual Private Network (VPN) server configured at INEGI.

Then, access to the application requires a login before any use case can be executed. Users must authenticate using their email and password, which are verified against the database. If the credentials match, a Two-Factor Authentication (2FA) process is triggered, sending a link to the user's associated email to access the dashboard and proceed with using the application. Upon successful authentication, a JSON Web Token (JWT) is generated and stored in localStorage. This token is used in subsequent requests to verify whether the user is authenticated and has the necessary permissions for specific use cases. The token remains valid for 15 days unless the Logout Application Programming Interface (API) is called.

There are two types of users: Data Supplier and Administrator. Administrators can create Users, Entities, Work Packages, Tasks, Life Cycle Assessment (LCA) Models and versions, and manage the information entered within the Scopes. Data Suppliers, on the other hand, can view the Work Packages, Tasks, and LCA Models associated. Admins create different LCA models in agreement with partner meetings and information provided regarding the main steps and connections between flows. Models have a visual tool (SUNDIAL diagram) that allows creating diagrams related to the processes under analysis and assigning users who will have access to them. Admins and users associated with the model can then manipulate the diagram and create entry flows in the inventory table for each process. Only the user who created an entry flow is allowed to edit its attributes. At the Table 8 the entry points and their corresponding trust levels are represented.

Table 8: Entry Points and Trust Levels of SUNDIAL

Id	Name	Description	Trust Levels
1	VPN Access	The VPN is used to securely connect to the INEGI network, granting access to the application.	*
1.1	Login Page	Login is the page that all users see when accessing the application for the first time.	(1) User with valid INEGI credentials (2) User (Data supplier) (3) Administrator (4) User with invalid Login credentials
1.1.1	Login Process	Users provide their login credentials, which are compared with the data stored in the database. If matched, (2) and (3) users need to pass the two-factor authentication process.	(2) User (Data supplier) (3) Administrator (4) User with invalid Login credentials
1.2	Project Management	The creation and editing of Work Packages, Tasks, and LCA Models requires the filling of forms.	(3) Administrator
1.3	LCA Model Data Entry	Through a set of nodes and connectors, it is possible to schematize the life cycle model from the raw materials to the final product using a toolset. Inventory tables are included within each process step of the diagram, to characterize what is occurring within. These tables contain a set of common attributes/inputs to characterize each entry flow.	(5) User assigned to the model
1.4	Inventory Table Update	No one can change the information provided, except for the User that created the entry flow.	(6) Creator of entry flow
1.5	Users Management	The creation and editing of Users and Entities requires the filling of forms.	(3) Administrator

3.3 Assets

By gaining a deeper understanding of SUNDIAL, we can identify its assets that require protection. Table 9 presents these assets along with brief descriptions.

Table 9: Assets of SUNDIAL

Asset Name	**Description**
User and Authentication Data	User credentials (email, password), roles (Admin, Data Supplier), and permissions.
Project Data	The platform's core intellectual property. Includes all information related to Work Packages, Tasks, and LCA Models.
Application Host Environment	The underlying infrastructure supporting the application (Ubuntu machine, Apache web server, and the Docker containerization environment).
Database	The shared MySQL server that stores and manages all data for every project hosted on SUNDIAL.
Network Security Perimeter	Components protecting the application's boundary. Includes the VPN, the firewall restricting traffic to the INEGI network and the reverse proxy managing traffic to the web containers.
Session Management	JWT tokens used to authorize requests after login.

3.4 Threat Modeling

Based on the analysis of the application and the information gathered, a DFD was created. This diagram, shown in Figure 4, was developed using the TMT tool and provides an overall understanding of the application, including its main components, functionalities, and interactions.

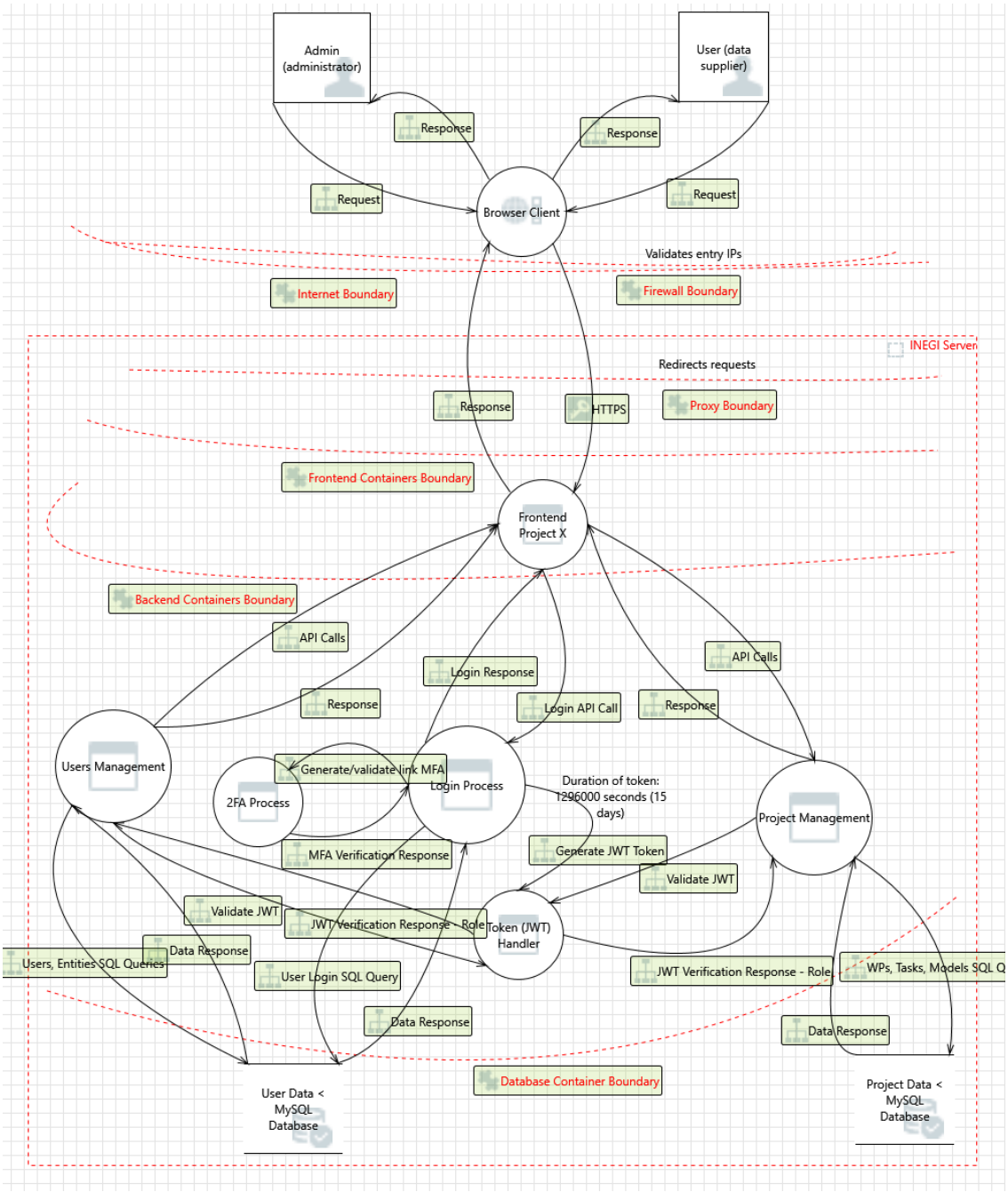


Figure 4: SUNDIAL Data Flow Diagram

A threat model was carefully constructed based on the analysis of the gathered information. Following the STRIDE framework and supported by known threats identified through CVEs, the ground truth includes 17 threats; all scenarios are listed in Table 16 in Appendix H, along with their respective impacts. In summary:

- **Spoofing**: Two documented threats focus on impersonation. The protection of Session Management and User and Authentication Data is undermined by allowing attackers to steal session tokens or intercept 2FA links, enabling them to act as legitimate users.

- **Tampering**: The integrity of the system is at risk from two identified threats. These scenarios threaten the Application Host Environment and Project Data by exploiting outdated software and insecure container configurations, which could permit unauthorized system modifications.
- **Repudiation**: Two threats were found that make it possible for users to deny their actions. Poor logging and session validation directly weaken the auditability of Project Data and undermine the reliability of Session Management.
- **Information Disclosure**: A significant risk of data leakage was identified across four scenarios. These threats primarily expose the Database and the Application Host Environment due to a lack of internal network controls and misconfigured services, creating multiple pathways for sensitive information to be revealed.
- **Denial of Service**: The platform's availability is threatened by three distinct DoS vectors. Attacks can exploit the absence of rate-limiting and resource caps, threatening the Network Security Perimeter and the stability of the Application Host Environment, and potentially causing a complete service outage.
- **Elevation of Privilege**: Four weaknesses create opportunities for privilege escalation. The issues, centered on flawed Session Management policies and Database vulnerabilities, could allow an attacker to gain broader access than authorized by exploiting overly long-lived tokens or known software flaws.

Chapter 4

Methodology and Experiment Design

This chapter details the methodology developed to examine the effectiveness of LLMs in generating threat models, which is briefly illustrated in Figure 5. The process encompasses the manual creation of a ground truth STRIDE threat model for the case study (Section 4.1), the presentation of the LLM-based Threat Modeling Tool and the initial prompt (Section 4.2), and the design of prompts for optimized LLMs to guide the generation of competing models for the same system (Section 4.3). Upon completion of these stages, the final phase involves a systematic evaluation framework in which the LLM-generated threat models are assessed against the ground truth (Section 4.4), using four evaluation metrics (Section 4.5).

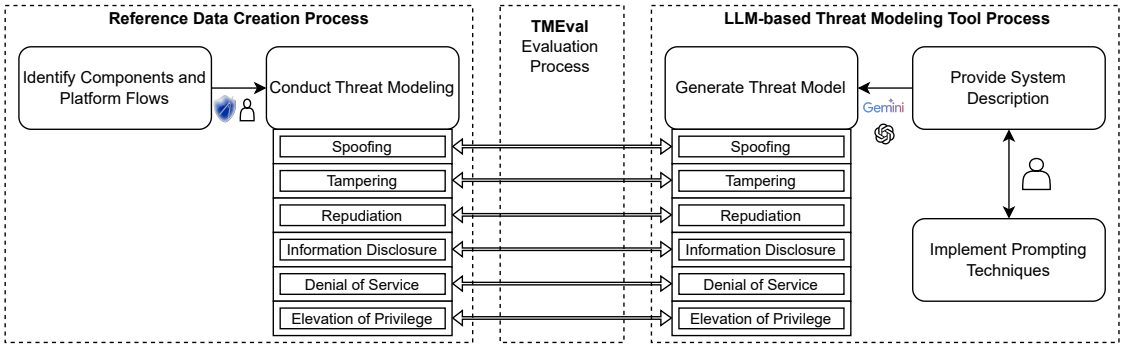


Figure 5: Methodology

4.1 Reference Data Creation

The definition of a reference model referred to as the ground truth, is essential to enabling a comparison-based evaluation of the threat models generated by different LLMs. The core idea behind this approach is to assess whether the threats identified by the LLMs closely align with those described in the reference model, based on a set of metrics and well-defined evaluation criteria.

The ground truth threat model was implemented for the SUNDIAL in the previous chapter. Its construction follows the OWASP Threat Modeling Process (Conklin, 2025), which consists of the following steps:

- **Application Decomposition**: Identification of potential attack targets and paths within the SUNDIAL platform; includes components such as external dependencies, entry points, assets, and trust boundaries.

- **Visual Representation of the Application**: The application is visually represented using a DFD created with the TMT tool, enabling a clearer understanding of the components' interactions as well as some simple threats already generated by the tool itself.
- **Threat Identification**: For each STRIDE category, threats are systematically identified to build a representative threat model to serve as the ground truth (Appendix H).

4.2 LLM-based Threat Modeling Tool

For the LLM-based threat modeling process, STRIDE GPT is the selected tool. Developed in Python, it is freely available as open-source software. The tool enables users to generate a threat model for a given application by providing a textual description of the system (Listing 1, lines 1-5), additional information required by the tool (lines 7-10) or an architecture diagram and optionally a link to the application's GitHub repository (Adams & Shibata, 2024).

```
1  Application description: The SUNDIAL platform was created by INEGI to provide a unified solution for
       managing sustainability data across multiple projects within the organization, functioning as a
       separate application for each one. Its main objective is to ensure that data collection, storage,
       and analysis follow a standardized format, facilitating the comparison and integration of
       information across different projects. Instead of each initiative developing its own system,
       SUNDIAL offers a common and configurable environment, enabling the efficient organization of large
       volumes of structured data, tracking activity history, and supporting the environmental, economic,
       and social assessment of initiatives. RELiEF is one of the projects, where the platform is used to
       organize and analyze data related to lithium recovery.
2  The application is running on a Linux server with Apache 2.4.52. Within this machine, the application is
        divided into Docker containers, which are set up in a separate network within the machine, created
        exclusively for this purpose. Containers for the frontend, backend, and database have different
       IPs and ports, and they are built and distributed by Jenkins. The frontend, React 17.0.2, a
       JavaScript library, and the backend, Node.js 14.18, vary for each project, while the database
       server, MySQL 8.0.32 is the same for all projects hosted by SUNDIAL. The entire web server is
       protected by a firewall and a proxy, which then forwards traffic to the web containers.
3  Access to the SUNDIAL platform is granted through a VPN server configured at INEGI.
4  Then, access to the application requires login before any use case can be executed. Users must
       authenticate using their email and password, which are verified against the database. If the
       credentials match, a two-factor authentication (2FA) process is triggered, sending a time-limited
       link to the user's associated email to access the dashboard and proceed with using the application.
        Upon successful authentication, a JWT token is generated and stored in localStorage. This token is
        used in subsequent requests to verify whether the user is authenticated and has the necessary
       permissions for specific use cases. The token remains valid for 15 days unless the Logout API is
       called.
5  There are two types of users: Data Supplier and Administrator. Administrators can create Users, Entities
       , Work Packages, Tasks, LCA (Life Cycle Assessment) Models and versions, and manage the information
        entered within the Scopes. Data Suppliers, on the other hand, can view the Work Packages, Tasks,
       and LCA Models associated. Admins create different LCA models in agreement with partner meetings
       and information provided regarding main steps and connections between flows. Models have a visual
       tool (SUNDIAL diagram) that allows creating diagrams related to the processes under analysis and
       assigning users who can have access to them. Admins and users associated with the model can then
       manipulate the diagram and create entry flows in the inventory table for each process. Only the
       user who created an entry flow is allowed to edit its attributes.
6
7  Application type: Web Application
8  Highest sensitivity level of the data processed: Confidential
9  Is the application internet-facing?: No
10 What authentication methods are supported by the application: MFA
```

Listing 1: STRIDE GPT Context

The choice of STRIDE GPT is justified by its specialization in threat modeling and its adherence to the STRIDE framework, which aligns with the focus of this study. Moreover, STRIDE GPT is already designed to optimize LLM performance by employing a structured, prompt-based approach that effectively guides the model toward generating valuable outputs. As implemented in Listing 2, the prompt is structured sequentially around four steps to guide the LLM: 1) defining the persona the model should emulate (line 1), 2) specifying the task to be performed (line 1), 3) clarifying the intended goal (lines 3-5), and 4) outlining the expected output format (lines 7-25).

```
Act as a cyber security expert with more than 20 years experience of using the
     STRIDE threat modelling methodology to produce comprehensive threat
    models for a wide range of applications. Your task is to analyze the
    provided code summary, README content, and application description to
    produce a list of specific threats for the application.

Pay special attention to the README content as it often provides valuable
    context about the project's purpose, architecture, and potential security
    considerations.

For each of the STRIDE categories (Spoofing, Tampering, Repudiation,
    Information Disclosure, Denial of Service, and Elevation of Privilege),
    list multiple (3 or 4) credible threats if applicable. Each threat
    scenario should provide a credible scenario in which the threat could
    occur in the context of the application. It is very important that your
    responses are tailored to reflect the details you are given.

When providing the threat model, use a JSON formatted response with the keys "
    threat_model" and "improvement_suggestions". Under  "threat_model",
    include an array of objects with the keys "Threat Type", "Scenario", and "
    Potential Impact".

...Other considerations provided to the user to improve the application
    description using a few-shot strategy.

Example of expected JSON response format:

"threat_model": [
    {{
        "Threat Type": "Spoofing",
        "Scenario": "Example Scenario 1",
        "Potential Impact": "Example Potential Impact 1"
    }},
    {{
        "Threat Type": "Spoofing",
        "Scenario": "Example Scenario 2",
        "Potential Impact": "Example Potential Impact 2"
    }},
    // ... more threats
]
```

Listing 2: STRIDE GPT Initial Prompt

4.3 Prompt Optimization

Initial tests will be conducted using the tool's default prompt; however, additional prompting strategies will also be explored (Figure 6).

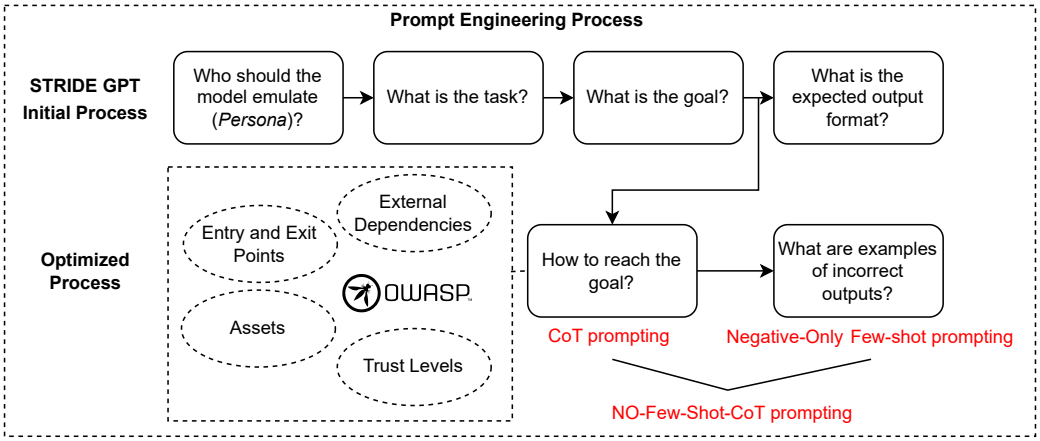


Figure 6: Prompt Engineering Process

Following the first step outlined in Section 5.1 for creating the ground truth, the model will be guided to proceed similarly using **Chain-of-Thought Prompting**. The prompt designed below instructs the LLM to follow the OWASP application decomposition steps in order to identify additional threats, with an effort to improve their scenarios based on the application context. To accomplish this, self-reflection within the prompt is also required, as the chain-of-thought approach asks the model not only to introduce new threats but also to reevaluate and improve upon those generated in the initial prompt.

```
After listing the threats, save your work. To expand your threat list, proceed
     by following the next steps in sequence:

1. ‘application_decomposition‘: Before generating any threats, you must first
    mentally decompose the application. Analyze all provided information to
    build a clear internal picture of the following components:
- ‘external_dependencies‘: Third-party services, APIs, or key libraries
- ‘entry_points‘: All ways data or commands enter the system
- ‘exit_points‘: All ways data leaves the system
- ‘assets‘: The most valuable data and components that need protection
- ‘trust_boundaries‘: Key areas where trust levels change

2. Based on the ‘application_decomposition‘, for each of the STRIDE categories
     (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of
    Service, and Elevation of Privilege), update or add more (3 or 4) credible
     threats to the list if applicable. Each threat scenario should provide a
    credible scenario in which the threat could occur in the context of the
    application. It is very important that your responses are tailored to
    reflect the details you are given.
```

Listing 3: Chain-of-Thought Prompt

Negative-Only Few-Shot Prompting emerges as a way to address the issue of generic threats, where LLMs tend to produce threats that fail to exclusively reflect the application and information provided by the user.

As written in Listing 4, the LLM is instructed to exclude and replace those generic threats through reflection on the threat model generated from the initial prompt. One example of an incorrect threat scenario will be provided for each STRIDE category, resulting in a 6-shot prompt. Although not commonly used, this adapted in-context learning strategy appears particularly well suited to the task.

```
Review the entire threat list carefully. Replace any generic threats by making
     them more specific and relevant to the application's context. Generic
    threats are threats that could beapplied to any system.
Examples of overly generic threats include:
- Spoofing:"An attacker compromises an account through social engineering"
- Tampering:"An attacker who has compromised a user account may manipulate
    sensitive data"
- Repudiation:"Due to insufficient logging, it might be difficult to determine
     who modified the data"
- Information Disclosure:"If the database is not properly secured,an attacker
    could gain access to it and extract all the information"
- Denial of Service:"A targeted DDoS attack against the web server could
    overwhelm the application, making it unavailable to users"
- Elevation of Privilege:"A vulnerability in the application could allow a
    user to escalate their privileges"
```

Listing 4: Negative-Only Few-Shot Prompt

4.4 Evaluation Process

At the evaluation stage, it is crucial to define a systematic approach for comparing LLM-generated threat models against the ground truth. To this end, a dedicated LLM evaluation tool, called Threat Model Evaluation Tool (TMEval), is introduced. TMEval is a simple Python tool tailored to assess the quality of STRIDE threat models produced by LLMs comparing them with a ground truth, thus offering a specialized focus that differentiates it from conventional LLM evaluation tools. Alongside traditional metrics (BLEU and ROUGE) and LLM-based metrics involving embedding comparison (BERTScore), an LLM-as-a-judge approach with customized evaluation criteria for each STRIDE category is thoroughly explored.

With the groundwork laid out, the tool will evaluate the outputs generated by the following models: Gemini 1.5 Pro, Gemini 2.5 Pro Preview, and GPT-4.5 Preview. These models were selected primarily due to cost considerations. The recency of their release was also a factor, with Gemini 1.5 Pro being the oldest, and the more recent Gemini 2.5 Pro and GPT-4.5 representing the current state of the art, both released around the same period.

Using the three models, four test scenarios will be executed exclusively within the context of the case study, with three iterations per each: 1) using STRIDE GPT's default prompt; 2) applying a Chain-of-Thought enhanced prompt; 3) employing the Negative-Only Few-Shot prompting strategy; and 4) combining both techniques.

Through these different strategies, it becomes possible to assess whether there are measurable improvements in the ability of the three LLMs to emulate application-specific threat modeling tasks, and to reinforce developers' confidence in leveraging LLMs to support system security.

4.5 Evaluation Metrics

To support a rich, multi-angled assessment, four metrics were carefully selected and integrated into TMEval.

As presented in Figure 7, the evaluation framework incorporates traditional and advanced techniques. The common metrics BLEU and ROUGE measure lexical similarity through word matching. In contrast, BERTScore compares the embeddings computed by the BERT

model for the ground truth with those for the threats generated by an LLM, enabling an evaluation that is aware of contextually similar threat scenarios.

Moreover, the framework employs LLM-as-a-Judge, an approach that leverages the reasoning capabilities of LLMs to evaluate the outputs generated by other AI systems. This innovative method relies on explicitly defined evaluation criteria, also referred to as dimensions, delivered through carefully crafted prompts (Appendices A-G) that follow a predefined Chain-of-Thought structure (Listing 5). The model receives these prompts and assigns scores from 1 to 5 for each dimension. To enhance reliability, each dimension score is measured eight times, and the average of these scores is computed. The final score represents the overall average across all dimensions, each assigned a percentage weight representing its relevance.

```
You will be given an input text, with a list of threats identified for an
    application. The table is given in a Markdown format with the following
    columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please
    keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Dimension Name (1-5): Dimension Description

Evaluation Steps:

A Chain-of-Thought consisting of 3 to 5 steps

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 5: Dimensions Prompt Structure

The evaluation criteria were specifically designed to assess the consistency between the reference and generated threats, the plausibility of a threat given its technical components, and the coverage reflected in the scenario descriptions. Accordingly, the following dimensions were defined:

- **Consistency**: All scenarios must represent consistent variations of those in the reference, avoiding hallucinations. A weight of 30% is assigned to this dimension, indicating that a high score corresponds to threats being nearly identical.
- **Plausibility**: Each scenario must make technical sense within the context of the inferred system described in the reference. A weight of 10% is allocated, as technical concepts, although important, may not always express valid threats within the context of the application under analysis.
- **Coverage**: The set of threats defined by the LLM must contain the same attack-related details as the threats presented in the reference. A weight of 20% is assigned to each of the sub-criteria:
  - **Targets**: Are the same targets identified as affected?

- **Weaknesses**: Are similar vulnerabilities or exposures highlighted?
- **Attacks**: Are comparable attack methods or strategies described?

For each reference-generated pair within a STRIDE category, the TMEval framework systematically incorporates these four evaluation strategies (Figure 7). This process generates multiple insights into the reliability of LLMs in such tasks and the applicability of the chosen metrics.

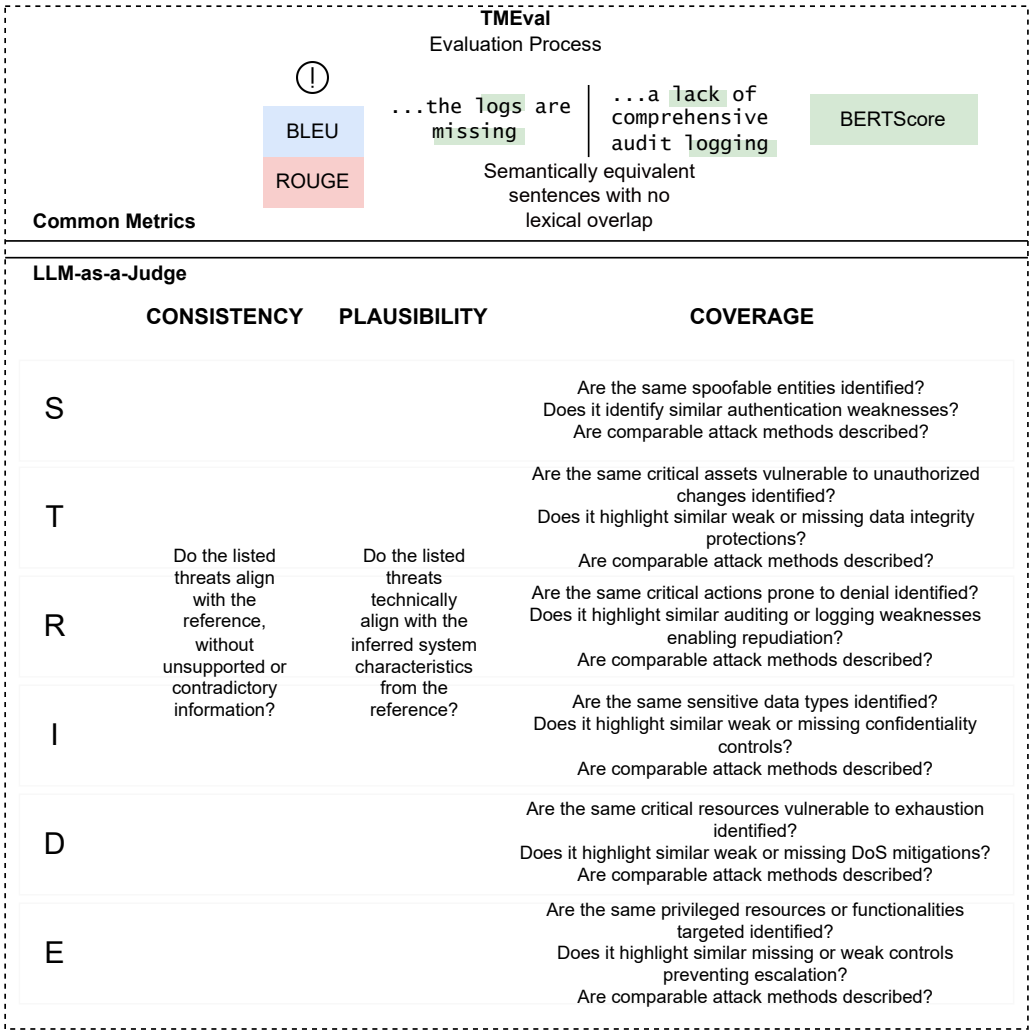


Figure 7: TMEval Framework

Chapter 5

Implementation and Evaluation

The TMEval tool is introduced in the first section, along with the configuration steps taken and the necessary adjustments made to ensure its effective operation in evaluating LLM-generated threat models. Section 5.2 will present the tool's results, followed by their analysis and discussion in Section 5.3.

5.1 TMEval and Metrics Configuration

TMEval was developed in Python. Its codebase and documentation are hosted on GitHub (Batista & Pereira, 2025). The tool allows for defining both the ground truth and the threat model generated by STRIDE GPT that is to be evaluated. Each of the four metrics can be run individually or multiple can be executed simultaneously, with a Plotly dashboard (Plotly, 2021) available for each metric where results can be analyzed. Although simple, the tool serves as a foundation for further research in this domain, accommodating future improvements.

Below are the principal configurations applied to each metric:

BERTScore:

- **Implementation**: "local-model". The evaluation runs locally, without the need for external API connections, utilizing available local Central Processing Unit (CPU) or Graphics Processing Unit (GPU) resources.
- **Model**: "roberta-large". A more capable variant of BERT, RoBERTa is used to compare threats semantically (HuggingFace, 2018).
- **IDF weighting**: true. Common English words such as "the", "a", and "an" are given less weight due to their low relevance in the comparison.
- **Scoring**: Reports precision (the proportion of generated tokens that match the reference), recall (the proportion of reference tokens covered by the generated text), and the F1 score, which is the balanced mean of precision and recall.

BLEU:

- **Max n-gram**: 4. The system compares sequences of consecutive words ranging from unigrams to 4-grams between the generated threats and the ground truth.
- **Weights and score**: [0.25, 0.25, 0.25, 0.25]. Equal weight is applied to all four n-gram matches. Includes a brevity penalty that penalizes generated scenarios that are too short compared to the reference.

ROUGE:

- **Use stemmer**: true. Allows the metric to consider different forms of the same word as matches (e.g., "log" and "logging"), helping to better capture similarity.
- **ROUGE types**: ["rouge1", "rouge2", "rougeL"]. Calculates unigram (ROUGE-1), bigram (ROUGE-2), and longest common subsequence (ROUGE-L) scores.
- **Scoring**: Provides precision, recall, and F1 score, aligned with BERTScore.

LLM-as-a-Judge:

- **Model**: "gemini-1.5-pro" (Google, 2025)
- **Number of completions**: 8. Outputs 8 scores per dimension across each category.
- **Throttling**:
  - **Requests per minute**: 40. Limits the API call rate to stay within usage quotas.
  - **Retry attempts**: 3. Automatically retries failed requests up to three times for improved reliability.
  - **Backoff factor**: 2.0. Uses exponential backoff to reduce congestion risk during retries.

5.2 Evaluation

Four tests were conducted, each with three iterations, across the three selected models. Accordingly, the subsections are organized so that each corresponds to one of the tests, enabling a structured analysis and discussion of the LLMs' performance. Subsection 5.2.1 uses and evaluates STRIDE GPT's default prompt; Subsection 5.2.2 employs and analyses the Chain-of-Thought prompting technique; Subsection 5.2.3 employs and assesses the Negative-Only Few-Shot prompting technique; and Subsection 5.2.4 adopts both techniques and examines their combined effectiveness.

5.2.1 Initial Prompt

The BLEU, ROUGE, and BERTScore metrics yield results that may not be appropriate for evaluating this task, as some scores are very low and the standard deviations across the three iterations are quite high (Table 10), given that these metrics primarily focus on lexical and semantic overlap. Therefore, the insights derived from these metrics do not help us meaningfully understand the LLMs' performance.

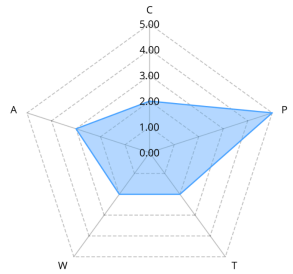
Table 10: BLEU, ROUGE, and BERTScore Results for the Initial Prompt (Three Iterations Average)

	Metrics \ Models	Gemini 1.5 Pro	Gemini 2.5 Pro Preview	GPT-4.5 Preview
S	BLEU	0.022 ± 0.009	0.019 ± 0.005	0.033 ± 0.013
	ROUGE-1	0.363 ± 0.037	0.325 ± 0.066	0.405 ± 0.006
	ROUGE-2	0.067 ± 0.018	0.069 ± 0.016	0.100 ± 0.010
	ROUGE-L	0.192 ± 0.014	0.130 ± 0.020	0.177 ± 0.013
	BERTScore	0.251 ± 0.052	0.211 ± 0.036	0.231 ± 0.020
T	BLEU	0.006 ± 0.003	0.008 ± 0.003	0.008 ± 0.002
	ROUGE-1	0.218 ± 0.013	0.194 ± 0.011	0.218 ± 0.012
	ROUGE-2	0.014 ± 0.001	0.022 ± 0.045	0.018 ± 0.016
	ROUGE-L	0.121 ± 0.012	0.088 ± 0.014	0.116 ± 0.013
	BERTScore	0.144 ± 0.028	0.103 ± 0.014	0.124 ± 0.031
R	BLEU	0.014 ± 0.009	0.010 ± 0.003	0.007 ± 0.003
	ROUGE-1	0.284 ± 0.045	0.282 ± 0.039	0.293 ± 0.076
	ROUGE-2	0.042 ± 0.017	0.034 ± 0.007	0.023 ± 0.011
	ROUGE-L	0.165 ± 0.051	0.160 ± 0.030	0.143 ± 0.030
	BERTScore	0.278 ± 0.052	0.207 ± 0.006	0.213 ± 0.014
I	BLEU	0.008 ± 0.004	0.010 ± 0.003	0.007 ± 0.004
	ROUGE-1	0.263 ± 0.033	0.304 ± 0.015	0.290 ± 0.035
	ROUGE-2	0.040 ± 0.024	0.039 ± 0.004	0.041 ± 0.032
	ROUGE-L	0.146 ± 0.016	0.123 ± 0.023	0.128 ± 0.016
	BERTScore	0.224 ± 0.044	0.192 ± 0.012	0.227 ± 0.032
D	BLEU	0.003 ± 0.001	0.010 ± 0.003	0.003 ± 0.002
	ROUGE-1	0.189 ± 0.026	0.267 ± 0.013	0.207 ± 0.034
	ROUGE-2	0.020 ± 0.014	0.039 ± 0.003	0.003 ± 0.006
	ROUGE-L	0.100 ± 0.011	0.116 ± 0.004	0.098 ± 0.014
	BERTScore	0.235 ± 0.041	0.188 ± 0.007	0.162 ± 0.014
E	BLEU	0.010 ± 0.007	0.008 ± 0.004	0.009 ± 0.003
	ROUGE-1	0.330 ± 0.063	0.311 ± 0.034	0.303 ± 0.057
	ROUGE-2	0.053 ± 0.009	0.030 ± 0.013	0.033 ± 0.006
	ROUGE-L	0.171 ± 0.030	0.118 ± 0.018	0.147 ± 0.011
	BERTScore	0.201 ± 0.037	0.139 ± 0.017	0.146 ± 0.006

On the other hand, the LLM-as-a-judge approach yields relatively better and more consistent results, as shown in Table 11, since its evaluation closely resembles human judgment.

Gemini 1.5 Pro underperforms in categories that require greater application-specific detail, most notably in Spoofing.

As noted by the comparison from Figure 8, which presents the Spoofing threats generated by Gemini 1.5 in the first iteration, the first two threats are generic and, therefore, deviate significantly from the ground truth. A phishing attack or an exploit targeting a VPN server vulnerability, without specifying the actual server used based on the provided system information, are considered generic and thus receive lower scores in the Attacks (A), Weaknesses (W), and Targets (T) coverage dimensions. In contrast, the third threat is more aligned with threat 2 from the ground truth, contributing to a slight improvement in these dimensions and in the Consistency (C) as well. Plausibility (P) receives a high score, as all technical concepts are aligned with the application.



Generated Threats

1. An attacker could attempt a phishing attack against INEGI employees to gain their VPN credentials, which would allow them to access the internal network where SUNDIAL resides. Unauthorized access to the SUNDIAL platform, potentially leading to data breaches or manipulation of sustainability data.

2. An attacker could exploit a vulnerability in the VPN server to impersonate a legitimate user and gain access to the SUNDIAL platform. Unauthorized access to the SUNDIAL platform, potentially leading to data breaches or manipulation of sustainability data.

3. An attacker intercepts the 2FA email and uses the link to gain unauthorized access. Unauthorized access to the SUNDIAL application, bypassing the MFA security measure

Ground Truth Threats

1. The platform uses JWT tokens for authentication and authorization, but these are stored in the browser's localStorage, which makes them vulnerable to XSS attacks, where an attacker can inject malicious code into the frontend to steal the token. The attacker acts as a legitimate user, thereby gaining unauthorized access to the platform's data and functionalities.

2. The access link generated by the 2FA process and sent via email can allow anyone with access to the email account and the INEGI network to gain access to the platform on behalf of the authenticated user. A critical risk also arises if multiple individuals are able to access the same session simultaneously. Anyone with access to the user's email receives the 2FA link and can impersonate them on the platform from within the internal network.

Figure 8: LLM-as-a-Judge Evaluation Example

Repudiation and Denial of Service allow for less variation in their descriptions; therefore, the generic nature of this LLM had limited impact. Consequently, for the Spoofing, Information Disclosure, and Elevation of Privilege categories, the results were better for the Gemini 2.5 Pro and GPT-4.5 models, which tend to provide more detailed responses, though occasionally produce hallucinated content and lack full alignment with the application's context.

Gemini 2.5 Pro, despite providing detailed threat scenarios, produced results that differed from those in the reference. As a result, its C criterion across most categories did not come close to the maximum score, illustrated clearly in Figure 9. The remaining dimensions (T, W, and A) showed average scores, given the high level of detail provided by the model.

GPT-4.5 was able to generate several threats that closely matched those in the reference. Its best-performing categories were Spoofing and Repudiation, as they achieved the highest scores in the C dimension.

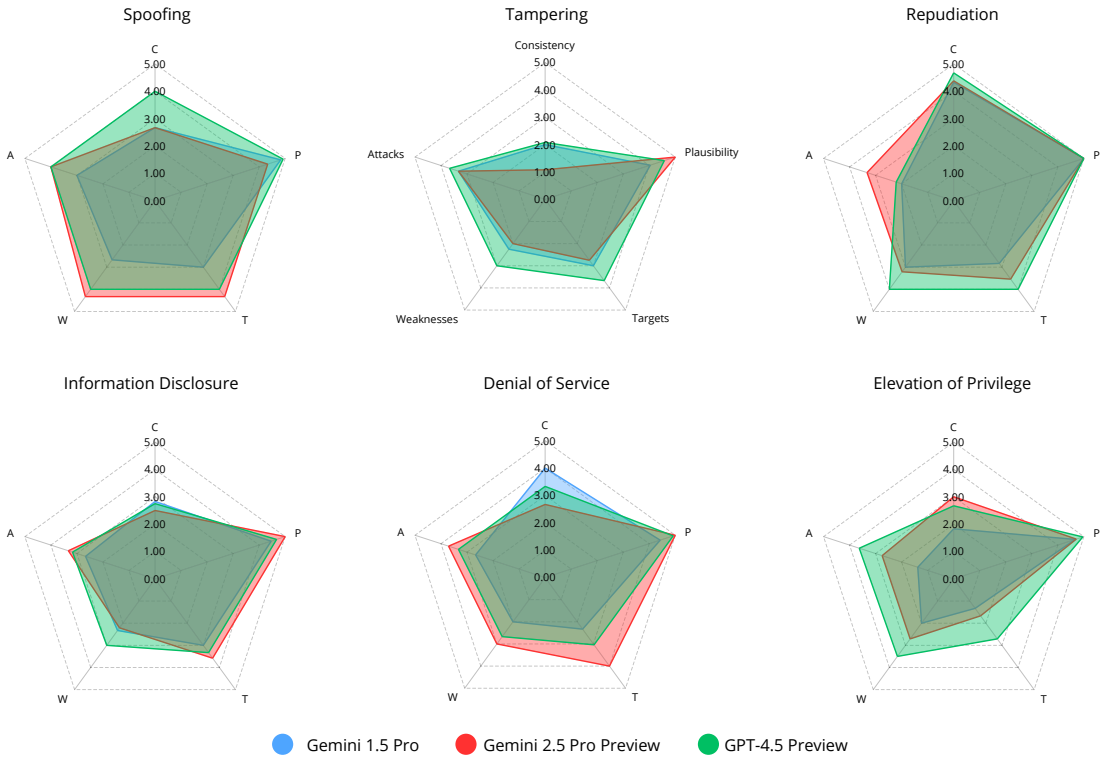


Figure 9: LLM-as-a-Judge Results for the Initial Prompt (Three Iterations Average)

Table 11: Interpretation of LLM-as-a-Judge Results for the Initial Prompt (Three Iterations Average)

	Gemini 1.5 Pro	**Gemini 2.5 Pro Preview**	**GPT-4.5 Preview**
S	3.017 ± 0.769	3.767 ± 0.814	4.092 ± 0.014
	The technical information is well-presented (high P), and the threat scenarios are mostly generic (medium T, W, and A). This variation in threats leads to an average C value	The technical information is well-presented (high P), and the threat scenarios are mostly diverse yet detailed (high T, W, and A). This variation in threats leads to an average C value	The technical information is well-presented (high P), and the threat scenarios are similar to those in the reference (high T, W, and A), which leads to a high C value
T	2.721 ± 0.198	2.442 ± 0.178	3.150 ± 0.307
	The technical information is well-presented (high P), and the threat scenarios are diverse, covering some key assets but varying in method (medium T, W, and A). The high variability in threats results in a low C value	The technical information is highly detailed and specific (high P), and the threat scenarios are diverse and advanced. While plausible, they diverge from the reference's specific examples (medium-to-low T, W, and A), resulting in an low C value	The technical information is well-presented (high P), and the threat scenarios are similar to those in the reference (medium-high T, W, and A), although with considerable variations, which leads to a low C value
R	3.367 ± 0.238	3.829 ± 0.355	3.942 ± 0.218
	The technical information is well-presented (high P), and the threat scenarios are highly repetitive and generic (medium-to-low T, W, and A), but consistently align with the reference's core problem, which leads to a high C value	The technical information is well-presented (high P), and the threat scenarios are diverse and detailed (medium T, W, and A). The introduction of new, advanced threats does not differ much from what the reference describes, leading to a high C value	The technical information is well-presented (high P), and the threat scenarios are mostly repetitive and generic, yet detailed (medium-high T, W, and A), but consistently align with the reference's core problem, which leads to a high C value
I	2.896 ± 0.430	3.075 ± 0.336	3.192 ± 0.530
	The technical information is well-presented (high P), but the threat scenarios are generic and focus on application-level attacks not present in the reference (medium-to-low T, W, and A), leading to a medium C value	The technical information is well-presented (high P), and the threat scenarios are diverse yet detailed (medium-high T, W, and A). They share some similarities with the reference by focusing on both infrastructure and application-level vulnerabilities, which leads to an average C value	The technical information is well-presented (high P), and the threat scenarios are closely similar to those in the reference (medium T, W, and A), which leads to a medium C value
D	3.042 ± 0.240	3.442 ± 0.246	3.300 ± 0.534
	The technical information is well-presented (high P), but the threat scenarios are generic (medium-to-low T, W, and A). The constrained nature of DoS scenarios results in a high C value	The technical information is highly detailed (high P), and the threat scenarios are diverse, aligning mostly with the DoS vectors from the reference (high T, W, and A), although there are a few hallucinations (medium C)	The technical information is well-presented (high P), and the threat scenarios show a mix of alignment, from generic coverage to specific examples that match the reference (medium T, W, and A). This variation leads to an average C value
E	1.958 ± 0.218	2.796 ± 0.406	3.263 ± 0.557
	The technical information is well-presented (high P), but the threat scenarios are highly repetitive and generic, failing to capture the specific and diverse vulnerabilities from the reference (low T, W, and A), which leads to a low C value	The technical information is highly detailed (high P), and the threat scenarios are alternative (medium-to-low T, W, and A). This variation and introduction of new threats leads to an average C value	The technical information is well-presented (high P), and the threat scenarios align well with the reference by focusing on both application and infrastructure vulnerabilities (medium-high T, W, and A), although there are a few hallucinations (medium C)

5.2.2 Chain-of-Thought Prompting

The Chain-of-Thought approach tends to generate more detailed responses, and for this task was no exception. Although the threats remained relatively generic, the added detail contributed to improved scores in certain categories.

The best-performing model was **Gemini 2.5 Pro**, which consistently produced more detailed outputs that closely aligned with the ground truth. This is exemplified by the Information Disclosure category, achieving a high score with this model (Figure 10), despite being an area with considerable variability in possible threat scenarios.

However, the primary challenge observed across all three LLMs lies in the misalignment of threat categorization with the ground truth, implying that STRIDE GPT encounters challenges in threat classification.

Furthermore, generic threats continue to persist and remain a significant limitation, particularly for **Gemini 1.5 Pro**, where this issue is most pronounced. This helps explain the results shown in Table 12. In the case of **GPT-4.5**, the threats showed little fluctuation compared to those from the initial prompt, suggesting that the Chain-of-Thought technique had minimal impact on this model.

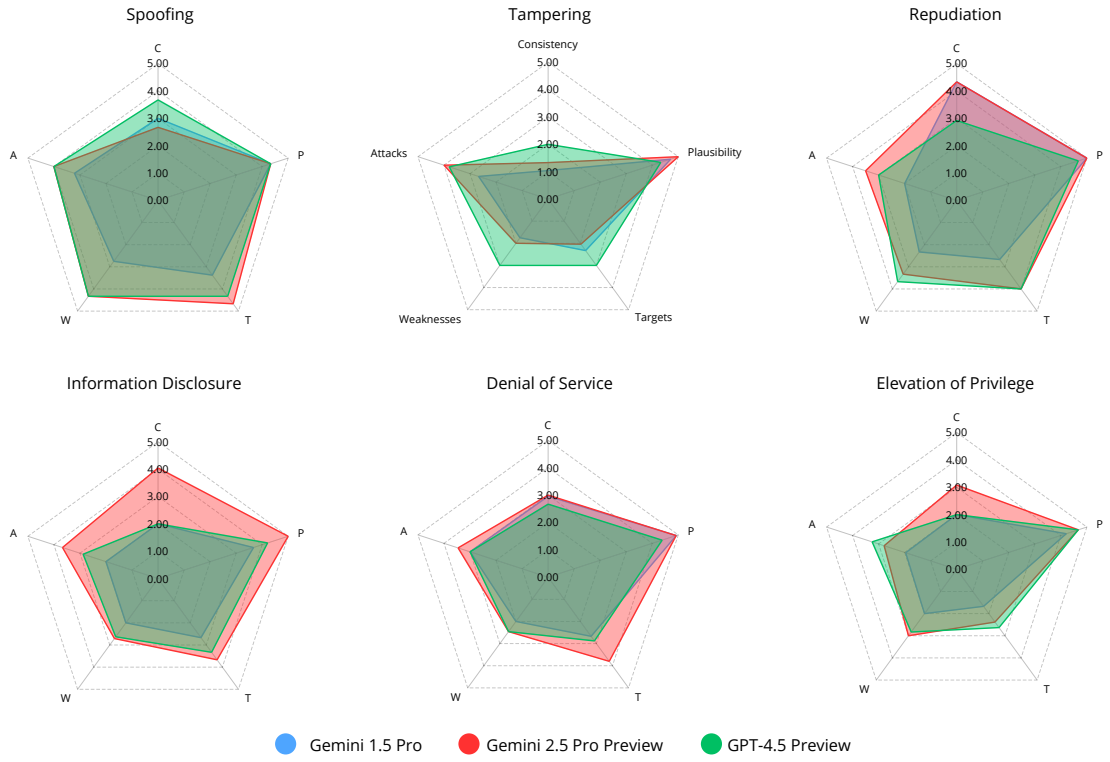


Figure 10: LLM-as-a-Judge Results for the Chain-of-Thought Prompt (Three Iterations Average)

Table 12: Interpretation of LLM-as-a-Judge Results for the Chain-of-Thought Prompt (Three Iterations Average)

	Gemini 1.5 Pro	**Gemini 2.5 Pro Preview**	**GPT-4.5 Preview**
S	3.200 ± 0.393	3.833 ± 0.603	4.058 ± 0.189
	The technical information is well-presented (high P), and the threat scenarios show a mix of generic phishing attacks and specific application vulnerabilities from the reference (medium T, W, and A). This variation in alignment leads to an average C value	The technical information is highly detailed (high P), and the threat scenarios are diverse and advanced (high T, W, and A). The introduction of new, advanced threats alongside those from the reference leads to an average C value	The technical information is well-presented (high P), and the threat scenarios are detailed (high T, W, and A). The similarity with those from the reference leads to a high C value
T	2.134 ± 0.191	2.508 ± 0.188	2.992 ± 0.126
	The technical information is well-presented (high P), but the threat scenarios are generic, failing to capture the vulnerabilities from the reference (medium-to-low T, W, and A), which leads to a low C value	The technical information is highly detailed and specific (high P), and the threat scenarios are diverse and advanced (medium-high T, W, and A). While plausible, they diverge from the reference's specific examples, resulting in a low C value	The technical information is well-presented (high P), and the threat scenarios are similar to those in the reference (medium-high T, W, and A), although with considerable variations, which leads to a low C value
R	3.200 ± 0.173	3.967 ± 0.513	3.475 ± 0.452
	The technical information is well-presented (high P), and the threat scenarios are highly repetitive and generic (medium-to-low T, W, and A), but consistently align with the reference's core problem, which leads to a high C value	The technical information is highly detailed (high P), and the threat scenarios are diverse and specific, aligning perfectly with the reference's threats (high T, W, and A), which leads to a high C value	The technical information is well-presented (high P), and the threat scenarios have techniques and gaps very similar to the reference (medium-high T, W, and A). The mix of relevant and also some unrelated threats leads to a medium C value
I	2.300 ± 0.100	3.721 ± 0.315	2.788 ± 0.238
	The technical information is well-presented (high P), but the threat scenarios are generic and focus on attacks not present in the reference (low T, W, and A), which leads to a low C value	The technical information is well-presented (high P), and the threat scenarios are diverse yet detailed (medium-high T, W, and A). They share some similarities with the reference by focusing on both infrastructure and application-level vulnerabilities, which leads to an average C value	The technical information is well-presented (high P), and the threat scenarios consistently blend infrastructure vulnerabilities from the reference with some unrelated threats (varied T, high W, and D). This mix of relevant and unrelated threats leads to a medium C value
D	2.913 ± 0.192	3.333 ± 0.469	2.904 ± 0.322
	The technical information is well-presented (high P), but the threat scenarios are generic (medium-to-low T, W, and A), which leads to a medium C value	The technical information is highly detailed (high P), and the threat scenarios are diverse and advanced, aligning perfectly with the reference's DoS vectors while adding plausible detail (medium-high T, W, and A), which leads to a medium C value	The technical information is well-presented (high P), and some threat scenarios resemble the reference but contain hallucinations (medium T, W, and A), which leads to a medium C value.
E	2.146 ± 0.139	3.025 ± 0.488	2.808 ± 0.305
	The technical information is well-presented (high P), but the threat scenarios are highly repetitive and generic, failing to capture the specific and diverse vulnerabilities from the reference (low T, W, and A), which leads to a low C value	The technical information is highly detailed and specific (high P), and the threat scenarios represent alternative and detailed cases (medium T, W, and A). They align well with the reference, which leads to a medium C value	The technical information is well-presented (high P), and the threat scenarios align well with the reference (medium T, W, and A), although there are a few hallucinations (low C)

5.2.3 Negative-Only Few-Shot Prompting

Providing examples effectively addressed the issue of generic threats. Spoofing, which previously exhibited a high number of generic threats, achieved strong results across all three LLMs.

Tampering, on the other hand, achieved poorer results, mainly due to a lack of detail. While the Chain-of-Thought technique contributed positively in this regard, the Few-Shot approach led the models to avoid generic threats but also caused them to overlook the application relevance.

As shown in Figure 11, the performance gap between the two categories is noticeable. Likewise, Table 13 presents the scores from the LLM-as-a-judge evaluation, where **Gemini 2.5 Pro** achieved the best results in Spoofing, Repudiation, and Elevation of Privilege, demonstrating strong alignment with the ground truth, particularly in the first two categories. In contrast, for the remaining three categories, **GPT-4.5** performed slightly better, especially in Denial of Service, where the threats displayed high consistency with the reference.

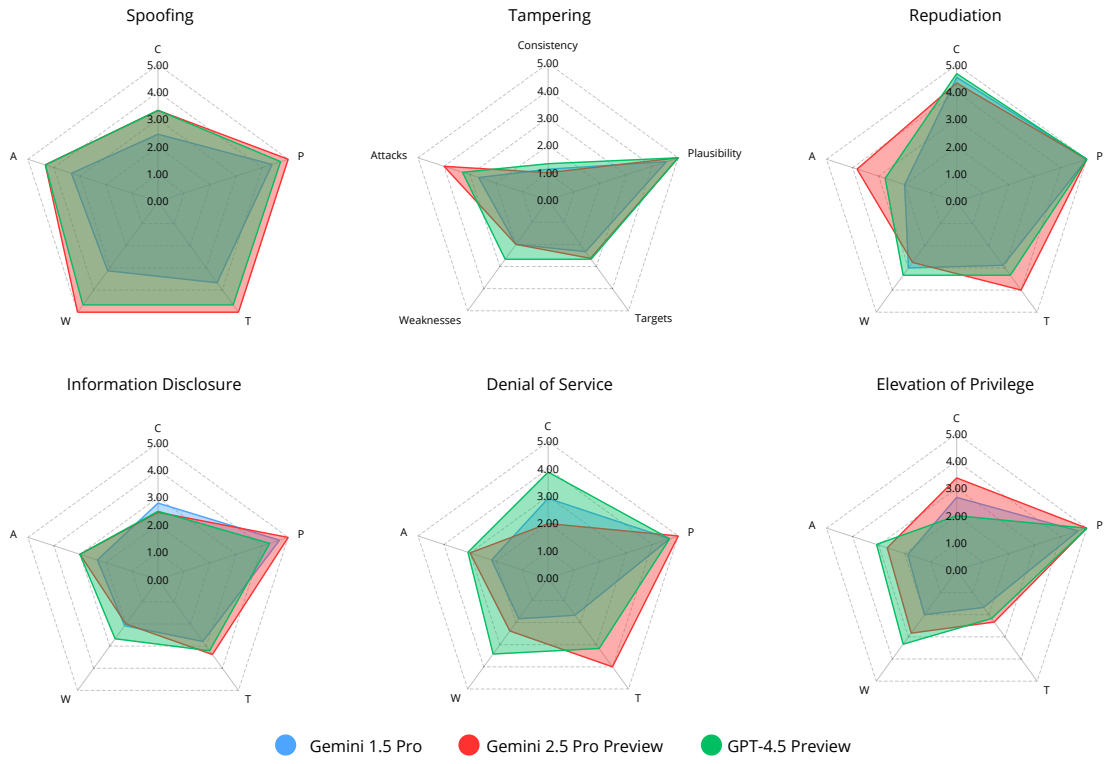


Figure 11: LLM-as-a-Judge Results for the Few-Shot Prompt (Three Iterations Average)

Table 13: Interpretation of LLM-as-a-Judge Results for the Few-Shot Prompt (Three Iterations Average)

	Gemini 1.5 Pro	**Gemini 2.5 Pro Preview**	**GPT-4.5 Preview**
S	3.200 ± 0.488	4.367 ± 0.345	4.204 ± 0.293
	The technical information is well-presented (high P), and the threat scenarios reflect concrete vulnerabilities from the reference, yet the descriptions retain a level of generality (medium T, W, and A). This variation in alignment leads to an average C value	The technical information is well-presented (high P), and the threat scenarios are similar to those in the reference (high T, W, and A), which leads to a medium-high C value	The technical information is well-presented (high P), and the threat scenarios are diverse (high T, W, and A). The introduction of new, advanced threats alongside those from the reference leads to an average C value
T	2.192 ± 0.452	2.525 ± 0.109	2.625 ± 0.353
	The technical information is well-presented (high P), but the threat scenarios lack sufficient detail (medium T, W, and A), resulting in an low C value	The technical information is well-presented (high P), but the threat scenarios lack sufficient detail (medium T, W, and A), resulting in an low C value	The technical information is well-presented (high P), but the threat scenarios lack sufficient detail (medium T, W, and A), resulting in an low C value
R	3.438 ± 0.492	3.917 ± 0.280	3.783 ± 0.453
	The technical information is well-presented (high P), and the threat scenarios are highly repetitive (medium-to-low T, W, and A), but consistently align with the reference's core problem, which leads to a high C value	The technical information is highly detailed (high P), and the threat scenarios are diverse, aligning perfectly with the reference's focus (high T, W, and A), which leads to a high C value	The technical information is well-presented (high P), and the threat scenarios are somewhat repetitive (medium T, W, and A). This consistent, albeit incomplete, alignment results in a high C value
I	2.746 ± 0.357	2.913 ± 0.163	2.954 ± 0.141
	The technical information is well-presented (high P), and the threat scenarios consistently blend infrastructure vulnerabilities from the reference with some unrelated threats (medium T, W, and A). This mix of relevant and unrelated threats leads to a medium C value	The technical information is highly detailed and specific (high P), and the threat scenarios focus on different elements (medium T, W, and A), which leads to a medium C value	The technical information is well-presented (high P), and the threat scenarios consistently blend infrastructure vulnerabilities from the reference with some unrelated threats (medium T, W, and A). This mix of relevant and unrelated threats leads to a medium C value
D	2.475 ± 0.256	2.975 ± 0.130	3.563 ± 0.206
	The technical information is well-presented (high P), but the threat scenarios are generic and do not align with the specific attack vectors and protection gaps in the reference (low T, W, and A), which leads to a medium C value	The technical information is highly detailed (high P), and the threat scenarios are diverse and align with the reference (medium-high T, W, and A), but with some hallucinations (medium C)	The technical information is well-presented (high P), and the threat scenarios align well with the reference's focus (medium-high T, W, and A), leading to a high C value
E	2.375 ± 0.347	3.079 ± 0.478	2.817 ± 0.284
	The technical information is well-presented (high P), but the threat scenarios are highly repetitive (low T, W, and A), which leads to a medium C value	The technical information is highly detailed (high P), and the threat scenarios align well with the core elements of the reference (medium T, W, and A), which leads to a medium C value	The technical information is well-presented (high P), but the threat scenarios lack sufficient detail (medium T, W, and A), resulting in an low C value

5.2.4 NO-Few-Shot-CoT Prompting

To achieve better results both in terms of detail (via Chain-of-Thought prompting) and in filtering out generic threats, a combination of the two prompting strategies was employed.

Immediately, by examining Figure 12, it becomes evident that one model stands out across most categories in terms of Consistency: **GPT-4.5**. **Gemini 2.5 Pro** follows closely behind, also achieving solid scores across the other evaluation dimensions. This performance suggests that these newer models demonstrate a better capacity to understand and retain more information. In contrast, **Gemini 1.5 Pro** scored significantly lower, which is expected given its older version and limited ability to process extended context.

Table 14 presents the final average results from the three iterations for each category. Serving as a complement to the figure, it reinforces that Gemini 1.5 Pro performed poorly compared to the other models, with Elevation of Privilege having the worst score across all tests.

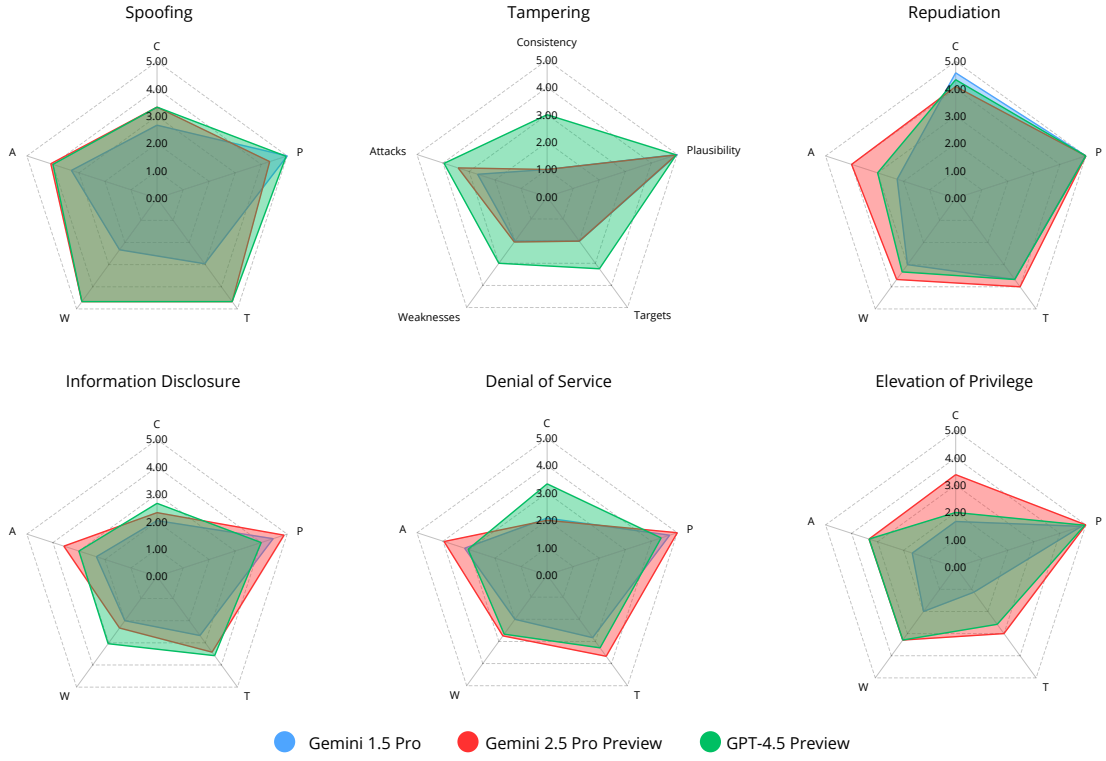


Figure 12: LLM-as-a-Judge Results for the Few-Shot Prompt (Three Iterations Average)

Table 14: Interpretation of LLM-as-a-Judge Results for the NO-Few-Shot-CoT Prompt (Three Iterations Average)

	Gemini 1.5 Pro	**Gemini 2.5 Pro Preview**	**GPT-4.5 Preview**
S	3.017 ± 0.359	3.117 ± 0.551	4.163 ± 0.474
	The technical information is well-presented (high P), and the threat scenarios still show a mix of some generic attacks and specific application vulnerabilities from the reference (medium T, W, and A). This variation in alignment leads to an average C value	The technical information is highly detailed and specific (high P), and the threat scenarios are diverse and advanced (high T, W, and A). The introduction of new, advanced threats alongside those from the reference leads to an average C value	The technical information is well-presented (high P), and the threat scenarios are diverse and advanced (high T, W, and A). The introduction of new, advanced threats alongside those from the reference leads to an average C value
T	2.133 ± 0.115	2.288 ± 0.373	3.437 ± 0.249
	The technical information is well-presented (high P), but the threat scenarios are still generic (low T, W, and A), which leads to a low C value	The technical information is highly detailed and specific (high P), and the threat scenarios are diverse and advanced (medium T, W, and A). While plausible, they diverge from the reference's specific examples, resulting in a low C value	The technical information is well-presented (high P), and the threat scenarios align well with the reference by focusing on both application and infrastructure vulnerabilities (high T, W, and A), which leads to a medium-high C value
R	3.658 ± 0.364	4.059 ± 0.106	3.800 ± 0.173
	The technical information is well-presented (high P), and the threat scenarios are highly repetitive (medium-to-low T, W, and A), but consistently align with the reference's core problem, which leads to a high C value	The technical information is highly detailed (high P), and the threat scenarios align perfectly with the reference's focus (high T, W, and A), which leads to a high C value	The technical information is well-presented (high P), and the threat scenarios are somewhat repetitive (medium T, W, and A). This consistent, albeit incomplete, alignment results in a high C value
I	2.458 ± 0.240	3.054 ± 0.220	3.125 ± 0.393
	The technical information is well-presented (high P), but the threat scenarios are generic and do not align with the specific attack vectors and protection gaps in the reference (medium-to-low T, W, and A), which leads to a low C value	The technical information is well-presented (high P), but the threat scenarios only align with some of the specific attack vectors and protection gaps in the reference (medium T, W, and A), which leads to a medium C value	The technical information is well-presented (high P), but the threat scenarios only align with some of the specific attack vectors and protection gaps in the reference (medium T, W, and A), which leads to a medium C value
D	2.696 ± 0.332	3.175 ± 0.241	3.238 ± 0.589
	The technical information is well-presented (high P), but the threat scenarios are generic and do not align with the specific attack vectors and protection gaps in the reference (medium T, W, and A), which leads to a low C value	The technical information is highly detailed (high P), and the threat scenarios are diverse and align with the reference (medium T, W, and A), but with hallucinations (low C)	The technical information is well-presented (high P), and the threat scenarios align well with the reference's infrastructure-focused DoS vectors (medium T, W, and A), which leads to a medium-high C value
E	1.942 ± 0.216	3.438 ± 0.349	2.938 ± 0.409
	The technical information is well-presented (high P), but the threat scenarios are mostly generic, failing to capture the specific and diverse vulnerabilities from the reference (low T, W, and A), which leads to a low C value	The technical information is highly detailed (high P), and some threat scenarios align with the reference (medium T, W, and A), leading to a medium C	The technical information is highly detailed (high P), and the threat scenarios are diverse and align with the reference (medium T, W, and A), but with hallucinations (low C)

5.3 Discussion

The aggregated analysis of the LLM-as-a-judge results, shown in Figure 13, provides a comparative overview of the performance of the three LLMs (Gemini 1.5 Pro, Gemini 2.5 Pro Preview, and GPT-4.5 Preview) across the six STRIDE categories, under the influence of four distinct prompting strategies. By combining the insights from this overall visualization with those from the previous individual results, the following subsections aim to discuss the performance of the models, the prompts, and each of the categories. Ultimately, these findings support a conclusion about which approach may offer the most balanced and effective way to automatically generate threats using AI.

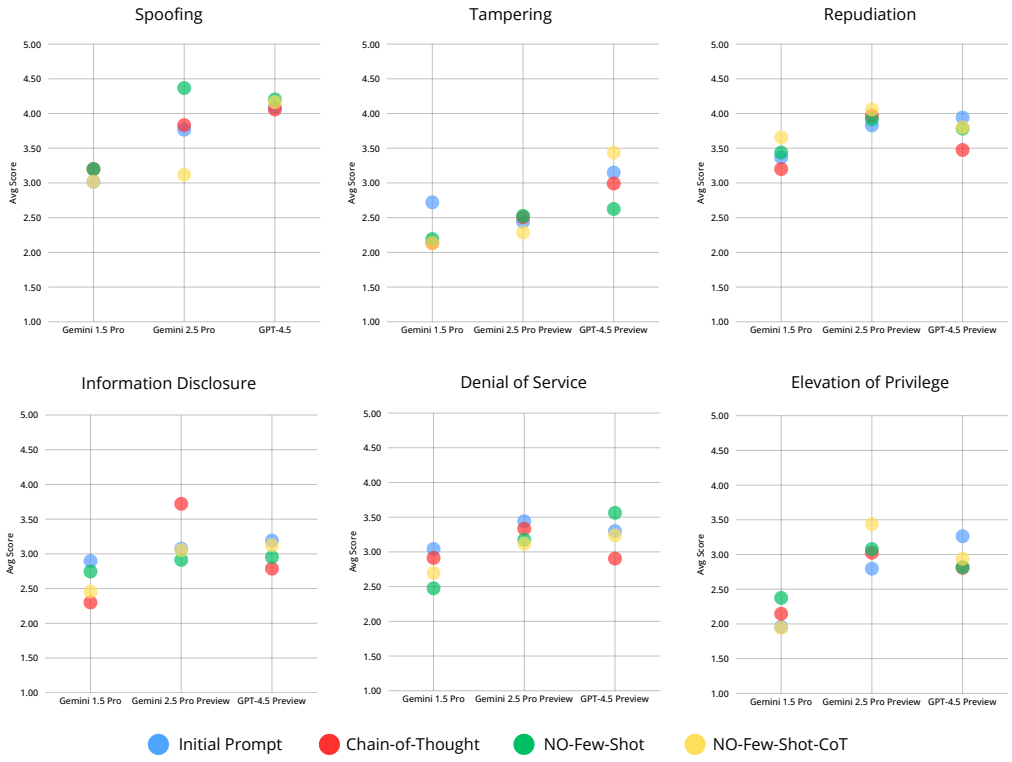


Figure 13: Average Score in Each Category Over the Four Prompts

5.3.1 Models Performance

The newer models, Gemini 2.5 Pro and GPT-4.5, clearly outperform Gemini 1.5 Pro. It seems that in almost all aspects, Gemini 1.5 Pro is the lowest performer irrespective of the prompting strategy employed. The degree of underperformance is striking in threat types that require more contextually comprehension like Spoofing, Information Disclosure, and Elevation of Privilege where it rarely exceeds a score of 3. On the other hand, both Gemini 2.5 Pro and GPT-4.5 attain these scores or even exceed them in these categories with GPT-4.5 especially strong in Spoofing, coming closest to the ground truth.

5.3.2 Prompting Strategies Effectiveness

Figure 13 reveals considerable variation in scores, with no single prompting strategy deemed superior; the effectiveness of each method depends on the model and threat category.

The CoT strategy significantly benefits Gemini 2.5 Pro, resulting in higher scores, especially in the Information Disclosure and Elevation of Privilege categories, where it surpasses the initial prompt. Guiding the model to reason step by step improves its ability to produce more detailed and relevant threat descriptions.

NO-Few-Shot proves effective for handling generic threats, as seen in the Spoofing category across all models, which is considered the most prone to standard threats. GPT-4.5 exhibits notable gains with NO-Few-Shot-CoT and maintains strong performance across multiple categories, particularly Tampering.

5.3.3 Performance Variation Across STRIDE Categories

The STRIDE categories present different levels of difficulty for the models.

Repudiation appears to be one of the easier categories, with models consistently scoring high and showing little variation across prompting strategies. This is due to the fact that repudiation threats tend to be less ambiguous and easier to identify.

On the other hand, Tampering and Elevation of Privilege seem to be the most difficult categories, with all models and prompting methods showing noticeably lower average scores.

A general analysis across all categories revealed that many threats are assigned to the wrong category; in other words, they would fit better in a different one, which could potentially lead to better results when compared to the ground truth.

5.3.4 Overall Synthesis

It can be concluded that there is no single prompting strategy that consistently produces the best threat model overall, as effectiveness varies significantly between models and threat categories. The only category where a clearly better prompting strategy can be identified is Spoofing, for which the NO-Few-Shot approach stands out; it helps avoid the generic nature of Spoofing threats such as phishing and social engineering. For other categories, considerable variability remains.

Table 15 maps each category to the most suitable prompting strategy, based on the top-performing model among the three evaluated for each strategy–category pair. Additionally, it provides examples of corresponding threat scenarios, demonstrating the potential of LLMs to generate valuable threat scenarios aligned with the specific components of the application.

Table 15: Mapping of STRIDE Categories to the Best LLM and Prompting Strategy

	Best Prompting Strategy	**LLM Example**	**Reference Example**
S	Gemini 2.5 Pro + NO-Few-Shot	A cross-site scripting (XSS) vulnerability in the React frontend could allow an attacker to inject malicious JavaScript. This script could then steal the active JWT token from the browser's localStorage, sending it to an attacker-controlled server.	The platform uses JWT tokens for authentication and authorization, but these are stored in the browser's localStorage, which makes them vulnerable to XSS attacks, where an attacker can inject malicious code into the frontend to steal the token.
T	GPT-4.5 + NO-Few-Shot-CoT	An attacker who breaches a Docker container (e.g., the frontend container via React application vulnerabilities) could manipulate frontend logic, allowing unauthorized actions or leading users to submit erroneous data unknowingly.	The application is part of an infrastructure that includes some outdated libraries. The various libraries and frameworks used must be continuously reviewed and updated, as attackers may exploit outdated versions or incompatibilities between them.
R	Gemini 2.5 Pro + NO-Few-Shot-CoT	An administrator deletes a user account. The application's audit log for this event is insufficient, only recording that 'an admin deleted a user' without specifying which administrator performed the action. The rogue admin can later deny their involvement.	An Administrator deletes Users, Entities, Work Packages, Tasks, LCA Models and versions without an immutable record of the operation. If something within the project is deleted, especially unintentionally, it may be very difficult to determine who performed the action and the reason for the deletion.
I	Gemini 2.5 Pro + Chain-of-Thought	A misconfiguration in the Node.js or Apache server causes detailed error messages and stack traces to be sent back to the user's browser upon an application crash. An attacker could intentionally trigger such errors to gather information about the internal system architecture, file paths, library versions, and database schemas, which can be used to plan more sophisticated attacks.	Improper redirection handling in the Apache server may allow attackers or malicious users to access sensitive configuration files that should otherwise remain hidden. It may reveal details about the internal structure of the application, helping the attacker identify other vulnerabilities.
D	GPT-4.5 + NO-Few-Shot	Poorly managed Docker container resources or missing resource limits could cause resource exhaustion within the Docker network.	Containers typically have CPU and memory limits configured. If these limits are absent or set too high, a container can consume excessive resources, potentially affecting the stability and performance of the entire system.
E	Gemini 2.5 Pro + NO-Few-Shot-CoT	The JWT stored in localStorage has a 15-day validity period. A user logs into SUNDIAL from a shared workstation and forgets to explicitly log out. Another individual can then use the same browser to access the active session.	A 15-day validity period for a JWT may be excessive if there is no short-lived access token combined with a refresh token mechanism. In the event the JWT is compromised, an attacker could maintain unauthorized access to the application for up to 15 days without reauthentication.

Chapter 6

Conclusions

This chapter summarizes the work carried out throughout the document, the contributions made, along with suggestions for possible future research.

6.1 Objectives Accomplished

Threat modeling has become more urgent than ever. Still, its complexity and the time required for its execution often hinder its adoption. LLMs have indeed shown significant improvements in their capabilities for cybersecurity instructions, allowing them to better focus on the provided context and produce more detailed outputs, which created an opportunity for more in-depth exploration in threat model generation.

Thus, the main objective was to analyze whether the use of LLMs for this task is useful and reliable. To this end, both the defined research questions and the practical objectives have been systematically addressed throughout this document. The following summary outlines how each of these elements was fulfilled:

- **Research Foundation**: Chapter 2 begins by providing the necessary background to understand the principles of Threat Identification and the motivation for exploring the use of LLMs, given their capabilities. Also, three research questions are systematically answered through a methodology based on PRISMA. From the analysis of two relevant studies, it was possible to examine how and when Threat Modeling is being implemented in practice (RQ1). Additionally, three recent works were identified in which automated Threat Modeling through LLMs was explored (RQ2), and within RQ3 the concept of LLM-as-a-Judge was introduced, proposing a set of evaluation criteria to assess the quality of LLM-generated responses in cybersecurity-related tasks, as demonstrated across two selected studies.
- **Definition of Ground Truth for the Case Study**: Comprehensive information was collected regarding the selected case study application, enabling its decomposition and facilitating a structured analysis. With this process, a STRIDE-based Threat Model Ground Truth was constructed, which served as a benchmark for comparison against LLM-generated threat models.
- **Use of LLMs to Generate Threat Models**: Threat models were generated for three LLMs (Gemini 1.5 Pro, Gemini 2.5 Pro, and GPT-4.5) using the STRIDE GPT tool. This was accomplished through four prompting strategies: STRIDE GPT's basic Initial Prompt, Chain-of-Thought (CoT), Negative-Only Few-Shot, and a combined strategy (No-Few-Shot-CoT).

- **Implementation of a Systematic Evaluation Framework**: The need to evaluate the performance of different prompting strategies in generating threats for each STRIDE category led to the implementation of TMEval as a dedicated evaluation tool. Incorporate four distinct metrics: BLEU, ROUGE, BERTScore, and LLM-as-a-Judge. BLEU, ROUGE, and BERTScore, although commonly used in NLP tasks, proved less effective in this context due to their reliance on surface-level lexical similarity or semantic equivalence. These methods fall short in capturing the nuanced technical and contextual differences inherent in cybersecurity threat scenarios.
- **Evaluation**: Based on the LLM-as-a-Judge results, guided by the defined evaluation criteria, an individual and comparative analysis was conducted. The findings indicate that each LLM and prompting strategy has its own strengths and weaknesses in terms of the specificity, generality, and technical depth of the generated STRIDE threats. Consequently, there is no universally optimal model or prompting strategy for automated Threat Modeling.

6.2 Contributions

With the goal of evaluating whether AI can assist non-experts in the creation of threat models, a structured methodology and a series of steps were followed to generate results that could provide meaningful insights. Despite some suboptimal outcomes, the findings confirmed that LLMs do hold potential to contribute meaningfully to enhancing the security of applications, not only within INEGI, but also across other organizations and personal projects.

The use of STRIDE GPT (Adams & Shibata, 2024) provided an accessible and open-source platform that enabled experimentation with different prompting strategies and investigation into the best approaches to improve the model's generated scenarios. A note of appreciation is extended to this tool, which lays the groundwork for continued exploration in this field.

Similarly, TMEval, the evaluation framework implemented in this study, is also publicly available on GitHub as an open project (Batista & Pereira, 2025). It allows a focus on evaluating threat models based on specific criteria, aiming to make the assessment as close as possible to a human-made one, while automating the process and consequently saving time in the evaluation step. Its availability encourages experimentation, adaptation, and further investigation into automated threat modeling.

6.3 Future Work

The present study was centered on a single case study, involving only three LLMs, four prompting strategies, and three iterations per configuration. While this provided valuable insights for SUNDIAL, the overall scope remains limited. Thus, the results may provide useful insights for similar contexts but cannot be directly generalizable to other applications.

Therefore, exploring additional prompting techniques and more advanced approaches to LLM optimization may be interesting given their rapid emergence. Also, risk prioritization and mitigation, crucial steps following threat identification, remain unexplored in this work and present promising opportunities for future research regarding their potential automation through LLMs.

Bibliography

Adams, M., & Shibata, K. (2024). Stride gpt: An ai-powered threat modeling tool.

Batista, A., & Pereira, N. (2025). https://github.com/nampereira/TMEval

Bolton, R., Sheikhfathollahi, M., Parkinson, S., Basher, D., & Parkinson, H. (2025). Multi-stage retrieval for operational technology cybersecurity compliance using large language models: A railway casestudy. https://arxiv.org/abs/2504.14044

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language models are few-shot learners. https://arxiv.org/abs/2005.14165

Casola, V., De Benedictis, A., Mazzocca, C., & Orbinato, V. (2024). Secure software development and testing: A model-based methodology. *Computers & Security*, *137*, 103639. https://doi.org/https://doi.org/10.1016/j.cose.2023.103639

Chowdhury, M. M., Rifat, N., Ahsan, M., Latif, S., Gomes, R., & Rahman, M. S. (2023). Chatgpt: A threat against the cia triad of cyber security. *2023 IEEE International Conference on Electro Information Technology (eIT)*, 1–6. https://doi.org/10.1109/eIT57321.2023.10187355

Conklin, L. (2025). https://owasp.org/www-community/Threat_Modeling_Process

Drake, V. (2020). https://owasp.org/www-community/Threat_Modeling

Ghosh, R., Farri, O., von Stockhausen, H.-M., Schmitt, M., & Vasile, G. M. (2024). Cve-llm : Automatic vulnerability evaluation in medical device industry using large language models. https://arxiv.org/abs/2407.14640

Google. (2025). https://ai.google.dev/gemini-api/docs/models

Honkaranta, A., Leppänen, T., & Costin, A. (2021). Towards practical cybersecurity mapping of stride and cwe — a multi-perspective approach. *2021 29th Conference of Open Innovations Association (FRUCT)*, 150–159. https://doi.org/10.23919/FRUCT52173.2021.9435453

HuggingFace. (2018). https://huggingface.co/docs/transformers/model_doc/roberta

IPP. (2020). Código de boas práticas e de conduta. https://www.iscap.ipp.pt/regulamentos/CodigoboaspraticasedecondutaIPP.pdf

Kaheh, M., Kholgh, D. K., & Kostakos, P. (2023). Cyber sentinel: Exploring conversational agents in streamlining security tasks with gpt-4. https://arxiv.org/abs/2309.16422

Kreitz, M. (2020). Security by design in software engineering. *SIGSOFT Softw. Eng. Notes*, *44*(3), 23. https://doi.org/10.1145/3356773.3356798

Krishna, V. B. (2024). Attackqa: Development and adoption of a dataset for assisting cybersecurity operations using fine-tuned and open-source llms. https://arxiv.org/abs/2411.01073

Kushwaha, M. K., David, P., & Suseela, G. (2024). Automation and devsecops: Streamlining security measures in financial system. *2024 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, 1–6. https://doi.org/10.1109/CONECCT62155.2024.10677271

Kwarteng, E., Cebe, M., & Kwarteng, J. (2024). Cyberllama2 - medicalharm threat modeling assistant. *2024 International Conference on Machine Learning and Applications (ICMLA)*, 930–934. https://doi.org/10.1109/ICMLA61862.2024.00136

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 74–81. https://aclanthology.org/W04-1013/

Liu, Y., Iter, D., Xu, Y., Wang, S., Xu, R., & Zhu, C. (2023). G-eval: Nlg evaluation using gpt-4 with better human alignment. https://arxiv.org/abs/2303.16634

MITRE Corporation. (2025). Common vulnerabilities and exposures (cve). https://www.cve.org/

Page, M. J., McKenzie, J. E., Bossuyt, P. M., Boutron, I., Hoffmann, T. C., Mulrow, C. D., Shamseer, L., Tetzlaff, J. M., Akl, E. A., Brennan, S. E., Chou, R., Glanville, J., Grimshaw, J. M., Hróbjartsson, A., Lalu, M. M., Li, T., Loder, E. W., Mayo-Wilson, E., McDonald, S., ... Moher, D. (2021). The prisma 2020 statement: An updated guideline for reporting systematic reviews. *BMJ*, *372*. https://doi.org/10.1136/bmj.n71

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: A method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, 311–318. https://doi.org/10.3115/1073083.1073135

Plotly. (2021). https://plotly.com/

Sion, L., Yskout, K., Van Landuyt, D., & Joosen, W. (2018). Solution-aware data flow diagrams for security threat modeling. *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, 1425–1432. https://doi.org/10.1145/3167132.3167285

Tian, S., Zhang, T., Liu, J., Wang, J., Wu, X., Zhu, X., Zhang, R., Zhang, W., Yuan, Z., Mao, S., & Kim, D. I. (2025). Exploring the role of large language models in cybersecurity: A systematic survey. https://arxiv.org/abs/2504.15622

Vu, T., Iyyer, M., Wang, X., Constant, N., Wei, J., Wei, J., Tar, C., Sung, Y.-H., Zhou, D., Le, Q., & Luong, T. (2023). Freshllms: Refreshing large language models with search engine augmentation. https://arxiv.org/abs/2310.03214

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). Chain-of-thought prompting elicits reasoning in large language models. https://arxiv.org/abs/2201.11903

Wu, T., Yang, S., Liu, S., Nguyen, D., Jang, S., & Abuadbba, A. (2025). Threatmodeling-llm: Automating threat modeling using large language models for banking system. https://arxiv.org/abs/2411.17058

Appendix A

Common Dimensions Prompts

The following listings present the prompts for each dimension defined in the LLM-as-a-Judge framework.

```
You will be given an input text, with a list of threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Consistency (1-5): the factual alignment between the input and the reference. A factually consistent input contains only statements that are entailed by the reference. Annotators were also asked to penalize input that contained hallucinated facts.

Evaluation Steps:

1. Read the reference carefully and identify the main facts and details it presents.
2. Compare the input with the reference. Check for factual errors or unsupported statements.
3. Assess additional threats that may initially seem dissimilar in the input, checking for consistency with the context of the reference.
4. Assign a score for consistency based on the evaluation criteria.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 6: Consistency Prompt

```
You will be given an input text, with a list of threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Plausibility (1-5): Do the threats listed in the input make technical sense given the systemâĂŹs inferred characteristics from the reference? This evaluation criteria assesses whether the threats in the input are logically coherent and free from technical inaccuracies or contradictions in relation to the reference.

Evaluation Steps:

1. Read the input and the reference threat lists carefully.
2. Compare the input against the reference and identify any technical inconsistencies, incorrect assumptions, or implausible threat scenarios.
3. Assign a score for plausibility based on the evaluation criteria.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 7: Plausibility Prompt

Appendix B

Spoofing Dimensions Prompts

```
You will be given an input text, with a list of Spoofing threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Entity Coverage (1-5): Assesses whether the input includes the same key entities that can be spoofed as those in the reference. This includes (but is not limited to):

- User accounts
- System services or APIs
- Hardware devices
- Application components or microservices

Evaluation Steps:

1. Carefully read both the input and the reference spoofing threat list.
2. Identify the entities in both the input and the reference that are susceptible to impersonation.
3. Compare the entities mentioned in the input with those in the reference, checking for alignment and completeness.
4. Based on the analysis above, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Entity Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 8: Entity Coverage Prompt

```
You will be given an input text, with a list of Spoofing threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Authentication Gaps Coverage (1-5): Assesses whether the input identifies comparable weaknesses in authentication mechanisms as those listed in the reference. This includes (but is not limited to):

- Basic password vulnerabilities
- Inadequate or missing Multi-Factor Authentication (MFA) implementation
- Weak or improperly managed session handling
- Exploitable flaws in identity providers or authentication protocols

Evaluation Steps:

1. Carefully read both the input and the reference spoofing threat list.
2. Identify the authentication-related gaps or weaknesses mentioned in the input and the reference.
3. Compare whether the input covers the same or comparable authentication weaknesses found in the reference.
4. Based on this analysis, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Authentication Gaps Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 9: Authentication Gaps Coverage Prompt

```
You will be given an input text, with a list of Spoofing threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Attack Vectors Coverage (1-5): Assesses whether the input identifies similar specific methods or techniques that an attacker might use to impersonate identities or spoof components in the system. This includes (but is not limited to):

- Phishing campaigns targeting user credentials
- Session hijacking
- Credential stuffing using leaked or weak passwords
- API key theft and subsequent API impersonation
- DNS spoofing or ARP poisoning
- Exploiting trust in federated identity providers
- Fake device ID submission
- Manipulating authentication headers in insecure API endpoints

Evaluation Steps:

1. Carefully read both the input and the reference spoofing threat list.
2. Identify the attack methods or spoofing techniques described.
3. Compare whether the input includes similar vectors as the reference.
4. Based on your comparison, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Attack Vectors Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 10: Attack Vectors Coverage Prompt

Appendix C

Tampering Dimensions Prompts

```
You will be given an input text, with a list of Tampering threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Asset Coverage (1-5): Assesses whether the input identifies the same critical assets susceptible to unauthorized modification as those listed in the reference. This includes (but is not limited to):

- Databases
- Configuration files
- Data in transit
- Firmware on embedded devices
- Source code repositories
- Compiled binaries

Evaluation Steps:

1. Carefully read both the input and the reference tampering threat list.
2. Identify the specific assets mentioned in the input and in the reference that are vulnerable to unauthorized modification.
3. Compare whether the assets mentioned in the input are consistent with those in the reference.
4. Based on the analysis above, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Asset Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 11: Asset Coverage Prompt

```
You will be given an input text, with a list of Tampering threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Integrity Gaps Coverage (1-5): Assesses whether the input identifies similar missing or weak protection mechanisms that are meant to ensure the integrity of systems and data, as referenced. This includes (but is not limited to):

- Absence of cryptographic hashes for file or data integrity verification
- Lack of digital signatures to ensure code authenticity
- Inadequate or missing integrity checks in communication protocols
- Insufficient input validation, leading to unauthorized modification of data
- Missing integrity controls in database operations or configuration files

Evaluation Steps:

1. Carefully read both the input and the reference tampering threat list.
2. Identify integrity-related gaps or missing protections described in each.
3. Compare whether the input includes the same or equivalent protection weaknesses mentioned in the reference.
4. Based on your analysis, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Integrity Gaps Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 12: Integrity Gaps Coverage Prompt

```
You will be given an input text, with a list of Tampering threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Tampering Methods Coverage (1-5): Assesses whether the input identifies similar attack techniques used to alter data, code, or system behavior. This includes (but is not limited to):

- SQL Injection (SQLi) to manipulate database contents
- Man-in-the-Middle (MITM) attacks intercepting and modifying data in transit
- Direct file system or configuration file manipulation
- Firmware tampering on embedded or IoT devices
- Code injection or modification
- Altering application logic through vulnerable APIs or interfaces
- Exploiting weak access controls to modify critical assets

Evaluation Steps:

1. Carefully read both the input and the reference tampering threat list.
2. Identify the attack methods or tampering techniques described.
3. Compare whether the input includes similar tampering methods as the reference.
4. Based on your analysis, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Tampering Methods Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 13: Tampering Methods Coverage Prompt

Appendix D

Repudiation Dimensions Prompts

```
You will be given an input text, with a list of Repudiation threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Action Coverage (1-5): Assesses whether the input identifies the same critical actions within the system that a malicious actor could later deny having performed, as also referenced in the reference. This includes (but is not limited to):

- Configuration changes
- Data deletions or edits
- Administrative commands
- Sensitive user actions
- System or application-level interactions that must be auditable

Evaluation Steps:

1. Carefully read both the input and the reference repudiation threat list.
2. Focus on actions that the attacker might perform and later repudiate.
3. Identify the specific types of actions in the input and in the reference.
4. Compare whether the actions in the input align with those in the reference in terms of auditability concerns.
5. Based on the above, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Action Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 14: Action Coverage Prompt

```
You will be given an input text, with a list of Repudiation threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Logging Gaps Coverage (1-5): Assesses whether the input identifies similar deficiencies in auditing and logging mechanisms that would enable repudiation, as mentioned in the reference. This includes (but is not limited to):

- Missing or incomplete timestamps in log entries
- Lack of unique identifiers for actions
- Failure to log critical events
- Logs that are vulnerable to tampering, deletion, or lack integrity controls
- Logging mechanisms with insufficient granularity to reconstruct user actions
- Absence of centralized or secure log storage

Evaluation Steps:

1. Carefully read both the input and the reference repudiation threat list.
2. Identify the specific logging gaps or audit deficiencies in both.
3. Compare whether the input lists the same (or comparable) issues as the reference.
4. Based on the analysis, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Logging Gaps Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 15: Logging Gaps Coverage Prompt

```
You will be given an input text, with a list of Repudiation threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Attack Vectors Coverage (1-5): Assesses whether the input identifies similar attack techniques or methods that enable an attacker to deny actions or cover their tracks. This includes (but is not limited to):

- Log tampering or deletion to erase evidence of actions
- Timestamp manipulation to confuse event timelines
- Use of anonymization or proxy services to hide identity
- Exploiting insufficient authentication for audit trails
- Replay attacks to obfuscate original actors
- Exploiting weak or missing digital signatures on critical operations
- Manipulating system clocks or audit mechanisms

Evaluation Steps:

1. Carefully read both the input and the reference repudiation threat list.
2. Identify the attack methods or repudiation techniques described.
3. Compare whether the input includes similar repudiation attack methods as the reference.
4. Based on your analysis, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Attack Vectors Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 16: Attack Vectors Coverage Prompt

Appendix E

Information Disclosure Dimensions Prompts

```
You will be given an input text, with a list of Information Disclosure threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Data Coverage (1-5): Assesses whether the input identifies the same sensitive data categories that could be exposed as listed in the reference. This includes (but is not limited to):

- Credentials
- Intellectual property
- Financial data
- Internal system architecture details

Evaluation Steps:

1. Carefully read both the input and the reference Information Disclosure threat list.
2. Identify the specific sensitive data categories mentioned in the input and in the reference.
3. Compare whether the data categories identified in the input align with those in the reference.
4. Based on the above, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Data Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 17: Data Coverage Prompt

```
You will be given an input text, with a list of Information Disclosure threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Protection Gaps Coverage (1-5): Assesses whether the input identifies similar missing or weak protection mechanisms aimed at preserving confidentiality of sensitive data, as described in the reference. This includes (but is not limited to):

- Absence of encryption for data at rest
- Absence of encryption for data in transit
- Weak or misconfigured access controls
- Lack of proper data masking for sensitive fields
- No Data Loss Prevention mechanisms in place
- Inadequate network segmentation exposing confidential data
- Logging or error messages leaking sensitive information

Evaluation Steps:

1. Carefully read both the input and the reference Information Disclosure threat list.
2. Identify any gaps in confidentiality-preserving controls mentioned in both.
3. Compare if the protection gaps identified in the input match or align with the reference.
4. Based on the comparison, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Protection Gaps Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 18: Protection Gaps Coverage Prompt

```
You will be given an input text, with a list of Information Disclosure threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Attack Methods Coverage (1-5): Assesses whether the input identifies similar attack techniques that could lead to the exposure of sensitive information. This includes (but is not limited to):

- Exploitation of misconfigured access controls to read sensitive data
- Network sniffing or packet capture to intercept data in transit
- SQL Injection or NoSQL Injection to extract confidential data
- Exploiting unencrypted data storage or backups
- Leveraging insecure APIs or endpoints to access private information
- Side-channel attacks revealing sensitive data indirectly
- Social engineering to obtain sensitive credentials or information

Evaluation Steps:

1. Carefully read both the input and the reference Information Disclosure threat list.
2. Identify the attack methods or techniques described that lead to data exposure.
3. Compare whether the input includes similar attack vectors as the reference.
4. Based on the analysis, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Attack Methods Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 19: Attack Methods Coverage Prompt

Appendix F

Denial of Service Dimensions Prompts

```
You will be given an input text, with a list of Denial of Service threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Resource Coverage (1-5): Assesses whether the input identifies the same critical resources susceptible to exhaustion or overload that could lead to denial of service, as listed in the reference. This includes (but is not limited to):

- CPU
- Network bandwidth
- Database connections
- Memory
- Available threads
- Application-level queues

Evaluation Steps:

1. Carefully read both the input and the reference Denial of Service threat list.
2. Identify the specific critical resources mentioned in the input and in the reference that could be exhausted or overwhelmed.
3. Compare whether the resources identified in the input align with those in the reference.
4. Based on the above, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Resource Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 20: Resource Coverage Prompt

```
You will be given an input text, with a list of Denial of Service threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Protection Gaps Coverage (1-5): Assesses whether the input identifies similar missing or weak mitigation mechanisms designed to prevent or reduce the impact of Denial of Service (DoS) attacks. This includes (but is not limited to):

- Absence of rate limiting or throttling on authentication or API endpoints
- Lack of resource usage monitoring and alerting
- No timeout mechanisms or request quotas for users/sessions
- Missing input validation leading to resource-intensive operations
- Vulnerable or misconfigured load balancers/firewalls
- Inadequate autoscaling or failover strategies
- Centralized resource bottlenecks
- Lack of separation between user workloads and system-critical processes

Evaluation Steps:

1. Carefully read both the input and the reference Denial of Service threat list.
2. Identify the missing or weak mitigations mentioned in each.
3. Compare whether the DoS protection gaps identified in the input are consistent with the reference.
4. Based on this analysis, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Protection Gaps Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 21: Protection Gaps Coverage Prompt

```
You will be given an input text, with a list of Denial of Service threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Attack Types Coverage (1-5): Assesses whether the input describes similar methods used to launch Denial of Service (DoS) attacks. This includes (but is not limited to):

- Network-level flooding attacks
- Application-layer attacks
- Resource starvation via legitimate but excessive requests
- Logic bombs or scheduled triggers that exhaust resources
- Exploitation of protocol weaknesses or malformed packets
- Amplification attacks

Evaluation Steps:

1. Carefully read both the input and the reference Denial of Service threat list.
2. Identify the attack types or DoS techniques described.
3. Compare whether the input includes similar DoS attack types as the reference.
4. Based on your analysis, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Attack Types Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 22: Attack Types Coverage Prompt

Appendix G

Elevation of Privilege Dimensions Prompts

```
You will be given an input text, with a list of Elevation of Privilege threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Vulnerability Point Coverage (1-5): Assesses whether the input identifies the same privileged resources or functionalities targeted by attackers in an attempt to escalate their privileges, as listed in the reference. This includes (but is not limited to):

- Administrative user accounts or privileged roles
- System-level configuration interfaces
- Protected operating system services or commands
- Restricted application features or data only accessible to higher-privilege users

Evaluation Steps:

1. Carefully read both the input and the reference Elevation of Privilege threat list.
2. Identify the specific privileged resources or functionalities targeted in the input and in the reference that an attacker may attempt to access through privilege escalation.
3. Compare whether the targets identified in the input align with those in the reference.
4. Based on the above, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Vulnerability Point Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 23: Vulnerability Point Coverage Prompt

```
You will be given an input text, with a list of Elevation of Privilege threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Control Gaps Coverage (1-5): Assesses whether the input identifies similar missing or weak security controls that, if properly implemented, would prevent privilege escalation. This includes (but is not limited to):

- Inadequate Role-Based Access Control (RBAC) implementation
- Lack of sandboxing or isolation for untrusted code
- Weak privilege separation between users or processes
- Services running with excessive privileges
- Insecure inter-process communication (IPC)
- Unpatched known vulnerabilities
- Default or overly permissive configurations
- Missing validation on flows that alter permissions or access levels

Evaluation Steps:

1. Carefully read both the input and the reference Elevation of Privilege threat list.
2. Identify the control weaknesses or gaps mentioned in each.
3. Compare whether the control gaps in the input align with those in the reference.
4. Based on the analysis above, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Control Gaps Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 24: Control Gaps Coverage Prompt

```
You will be given an input text, with a list of Elevation of Privilege threats identified for an application. The table is given in a Markdown format with the following columns
Threat Type, Scenario, Potential Impact.

Your task is to rate the input on one metric.

Please make sure you read and understand these instructions carefully. Please keep this document open while reviewing, and refer to it as needed.

Evaluation Criteria:

Exploit Methods Coverage (1-5): Assesses whether the input describes similar techniques used to achieve privilege escalation. This includes (but is not limited to):

- Token theft or impersonation
- Container breakout or escape from sandbox environments
- Exploiting insecure sudo or privilege delegation configurations
- DLL hijacking or binary planting attacks
- Leveraging insecure service or daemon configurations
- Exploiting kernel vulnerabilities or flawed access controls
- Abusing misconfigured Role-Based Access Control (RBAC) policies

Evaluation Steps:

1. Carefully read both the input and the reference Elevation of Privilege threat list.
2. Identify the exploit techniques or methods described.
3. Compare whether the input includes similar privilege escalation exploit methods as the reference.
4. Based on your analysis, assign a score from 1 (very poor coverage) to 5 (excellent coverage) for the Exploit Methods Coverage evaluation criterion.

Reference: {reference}

Input: {input}

Evaluation Form (scores ONLY):
```

Listing 25: Exploit Methods Coverage Prompt

Appendix H

Ground Truth Threat Model

Table 16 shows the manually defined STRIDE threats used as the ground truth.

Table 16: SUNDIAL Threat Model

Threat Type	Scenario	Potential Impact
Spoofing	The platform uses JWT tokens for authentication and authorization, but these are stored in the browser's localStorage, which makes them vulnerable to XSS attacks, where an attacker can inject malicious code into the frontend to steal the token.	The attacker acts as a legitimate user, thereby gaining unauthorized access to the platform's data and functionalities.
Spoofing	The access link generated by the 2FA process and sent via email can allow anyone with access to the email account and the INEGI network to gain access to the platform on behalf of the authenticated user. A critical risk also arises if multiple individuals are able to access the same session simultaneously.	Anyone with access to the user's email receives the 2FA link and can impersonate them on the platform from within the internal network.
Tampering	The application is part of an infrastructure that includes some outdated libraries. The various libraries and frameworks used must be continuously reviewed and updated, as attackers may exploit outdated versions or incompatibilities between them.	Changes in the application's behavior or the introduction of vulnerabilities (CVEs), compromising the system's integrity.
Tampering	Docker containers if running with root privileges pose a significant security risk. If a container from any project accepts data through API requests without properly validating for malicious shell commands, an attacker could potentially exploit the container.	Could potentially escape the container and compromise other project containers or even the underlying host system.

Threat Type	Scenario	Potential Impact
Repudiation	Repudiation can occur, for example, if a user modifies information in an LCA model but the JWT is not properly validated, and the logs are missing or fail to track the action.	The inability to trace who made a change can undermine the system's reliability. If a user is incorrectly assigned to a model, it could lead to even more serious organization issues.
Repudiation	An Administrator deletes Users, Entities, Work Packages, Tasks, LCA Models and versions without an immutable record of the operation.	If something within the project is deleted, especially unintentionally, it may be very difficult to determine who performed the action and the reason for the deletion.
Information Disclosure	Despite the presence of a firewall and VPN controlling external access to the internal network, the absence of an internal firewall creates a critical vulnerability. Without internal traffic restrictions and monitoring, malicious actors or compromised services can move laterally within the network, increasing the risk of attacks like server-side request forgery (SSRF).	Without proper internal segmentation, an attacker exploiting an SSRF vulnerability could potentially access sensitive services, including the database and Docker container configurations.
Information Disclosure	The shared MySQL database may permit unauthorized access to data from other projects due to misconfigurations in table-level permissions and/or schema design flaws.	Data exposure between projects and confidentiality breaches could occur.
Information Disclosure	Improper redirection handling in the Apache server may allow attackers or malicious users to access sensitive configuration files that should otherwise remain hidden.	It may reveal details about the internal structure of the application, helping the attacker identify other vulnerabilities.
Information Disclosure	By default, services running inside containers are not accessible from outside. However, if isolation is not properly configured, its ports may be exposed externally, allowing outside access to the service.	Information leakage may occur, particularly if the MySQL database container is exposed. If it is running a version with known vulnerabilities, an attacker could exploit it, potentially gaining full control over the service.

Threat Type	**Scenario**	**Potential Impact**
Denial of Service	An attacker or malicious user could exploit the fact that the MFA process does not have an appropriate consumption limit setting, allowing multiple logins and excessive email links to be sent.	Delays or failures in delivering MFA links for other authentication attempts may occur.
Denial of Service	A DoS attack against the proxy or Apache could be executed by sending a massive volume of malformed requests. If the firewall only blocks IP addresses and does not detect abnormal behavior, and if the proxy is not configured to limit the number of requests per second per IP, the system remains vulnerable to such attacks.	A complete service interruption could occur, affecting all projects hosted on SUNDIAL.
Denial of Service	Containers typically have CPU and memory limits configured. If these limits are absent or set too high, a container can consume excessive resources, potentially affecting the stability and performance of the entire system.	A single container can cause a denial of service to others if the internal network is not properly segmented. For example, if a frontend container can communicate with another project's frontend container, even though the projects are independent.
Elevation of Privilege	The virtual machine hosting the web server and its containers may have services running with excessive privileges or open ports that are not essential to the application's functionality.	Unnecessary open ports can serve as entry points for an attack.
Elevation of Privilege	A user account with limited privileges on the database may be able to exploit a recently discovered vulnerability in MySQL (CVE 2025-21567) and gain access to data or resources they should not be allowed to access.	Access to sensitive data or even escalating their privileges within the system.
Elevation of Privilege	A 15-day validity period for a JWT may be excessive if there is no short-lived access token combined with a refresh token mechanism.	In the event the JWT is compromised, an attacker could maintain unauthorized access to the application for up to 15 days without reauthentication.
Elevation of Privilege	Is the JWT token scoped to specify the project for which it is authenticated? If not, a token initially issued and used for one project could be reused or injected into other projects, potentially allowing unauthorized access by users who should not have permission to those projects.	The user is able to perform actions or access data from another project for which they were not originally authorized.