

Emerging Challenges in Threat Modeling for GenAI-Augmented Systems: A View from the Trenches

Nicolás E. Díaz Ferreyra ✉ 🏠 

Institute of Software Security, Hamburg University of Technology, Germany

Manish Mahesh Kumar ✉ 🏠

Institute of Software Security, Hamburg University of Technology, Germany

Nohemí Villarreal ✉ 🏠

CREATUM GmbH, Hamburg, Germany

Pankaj Pantel ✉ 🏠

CREATUM GmbH, Hamburg, Germany

Immo Brueggemann ✉ 🏠

CREATUM GmbH, Hamburg, Germany

Riccardo Scandariato ✉ 🏠 

Institute of Software Security, Hamburg University of Technology, Germany

Abstract

Threat modeling remains a central task in secure software engineering, as it enables the identification of security issues from system architectures. As Generative Artificial Intelligence (GenAI) becomes increasingly pervasive across software systems, traditional threat modeling methods (e.g., STRIDE) are insufficient to assess emerging GenAI-specific risks. In this work, we present the first results from an exploratory assessment of GenAI-aware threat modeling methods in a Small and Medium Enterprise (SME) setting. For this, we conducted a rapid literature review to select relevant techniques and systematically applied three shortlisted methods to an industrial case study involving a GenAI-augmented system. The results highlight differences in the threats identified by each technique and reveal limited support for certain GenAI-specific risk categories, particularly those related to software supply chains and human-centered security issues. We further report practitioners' perceptions of the usability and integration of these methods in SME development workflows, including their perceived effort and adoption challenges.

2012 ACM Subject Classification Security and privacy; Security and privacy → Usability in security and privacy; Security and privacy → Software security engineering

Keywords and phrases Threat Modeling, Software Architectures, Generative AI, ML, Sec4AI

Digital Object Identifier 10.4230/LIPIcs.ESEM.2026.57

Category Emerging Results, Vision & Reflection Track Paper

Supplementary Material *Dataset:* <https://doi.org/10.5281/zenodo.21706295>

Funding This work was supported by the European Union under grant No. 101120393 (Sec4AI4Sec).

1 Introduction

The security of software-intensive systems is a multifaceted challenge that requires coordinated efforts from stakeholders across the entire development lifecycle. Threat Modeling (TM) is a key component of the security toolkit, as it provides a structured set of methods and artifacts to support the early identification of potential security threats at the design stage [34]. Over the years, several architecture-centered TM methods have been proposed and extensively discussed in the current literature [40]. One of the most widely adopted is STRIDE [11], a methodology introduced by Microsoft in 2008, which enables practitioners to



© Nicolás E. Díaz Ferreyra, Manish Mahesh Kumar, Nohemí Villarreal, Pankaj Pantel, Immo Brueggemann, and Riccardo Scandariato;
licensed under Creative Commons License CC-BY 4.0

20th International Symposium on Empirical Software Engineering and Measurement (ESEM 2026).
Editors: Robert Feldt, Maria Paasivaara, Daniel Mendez, Stefan Wagner, and Marvin Muñoz Barón;
Article No. 57; pp. 57:1–57:15



Leibniz International Proceedings in Informatics
LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

systematically identify threats by categorizing them into six classes: (S)poofing, (T)ampering, (R)epudiation, (I)nformation Disclosure, (D)enial of Service, and (E)levation of Privilege [22]. Similar approaches also rely on mnemonic threat taxonomies and structured representations of system components (e.g., data flow diagrams) to reason about potential attack vectors and security violations [30, 19, 16]. Furthermore, dedicated tool support is often available to facilitate their application, thereby reducing manual effort [8].

Motivation. General-purpose methods such as STRIDE often struggle to capture the full threat landscape of complex, domain-specific systems. Consequently, several variants have emerged over the years to address the specific security challenges of domains such as IoT [31], automotive systems [1], and industrial control systems [26] among others. These variants typically extend existing taxonomies with additional threat categories or introduce domain-specific modeling constructs to better represent the system under analysis. The rapid adoption of Generative Artificial Intelligence (GenAI) technologies has further motivated the adaptation of State-of-the-Art (SotA) TM techniques to account for risks specific to these systems, such as prompt injection and data leakage through model interactions [17, 9, 36, 13]. Still, there is limited empirical evidence on the applicability of these techniques in real-world settings. In particular, it remains unclear how well these approaches support practitioners, especially in Small and Medium Enterprise (SME) contexts, and where their most salient gaps and challenges lie [12].

Contribution and Research Questions. In this work, we explore the gaps and emerging challenges of TM techniques when applied to GenAI-augmented software systems. To this end, we first conducted a rapid literature review to identify SotA methods tailored to the analysis of GenAI-specific security threats. We then applied three shortlisted techniques to an industrial case study provided by a local SME to examine their performance in practice. Finally, we complemented our analysis with a practitioner survey to capture perceptions regarding the usability and integration of these techniques in real-world development workflows. Overall, we aim to answer the following Research Questions (RQs):

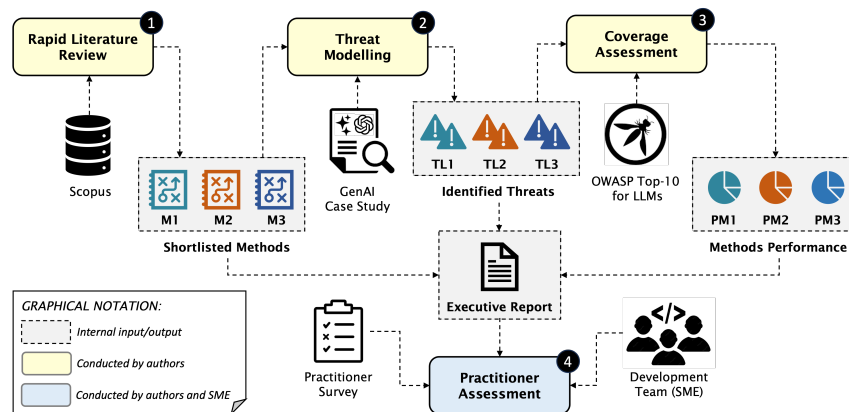
- **RQ1:** *How effective are state-of-the-art threat modeling techniques in identifying GenAI-specific security issues in an SME setting?* To answer this RQ we modeled and analyzed the system architecture of the given case study following the steps prescribed by each of the three selected TM methodologies. The outputs of each method (e.g., type and number of security threats) were then assessed in terms of their coverage against the OWASP Top 10 for LLM Applications [23], a taxonomy of threats specific to Large Language Models (LLM) systems developed by security practitioners.
- **RQ2:** *How do practitioners perceive the adoption and integration of these techniques in an SME context?* We conducted a follow-up survey with members of the development team of the SME that provided the case study, aiming to elicit their perceptions of each of the applied TM techniques. The survey included a set of items assessing dimensions such as perceived usefulness (e.g., ability to identify relevant and actionable threats), effort required for regular use, and integration with current tools and practices. Thereby, we provide initial insights into practitioners' perceptions, including practical gaps, challenges, and barriers to adoption and integration.

2 Background and Related Work

(i) Threat modeling in practice. Threat Modeling (TM) is central to many security-by-design frameworks and standards, as it supports the systematic identification of risks arising from system architectures and information flows. Typically, it involves modeling the target system, eliciting threats using structured taxonomies, and prioritizing them to inform mitigation strategies. Data Flow Diagrams (DFDs) are frequently used to create a graphical, high-level representation of the software architecture and depict how information moves across the main system components [33]. In their most basic variant, DFDs include four types of elements, namely (i) external entities, (ii) data flows, (iii) processes, and (iv) data stores [28]. During threat analysis, security and domain experts inspect each of these elements with the help of heuristics provided by the TM method to identify potential security threats. In STRIDE, these heuristics take the form of a structured mapping between DFD elements and threat categories (e.g., data stores associated to information disclosure threats) [11].

Because of its relative ease of use and adaptability, STRIDE has become a de facto baseline in security TM and has been evaluated across multiple studies. These evaluations have highlighted not only its lightweight design but also practical limitations regarding scalability and applicability to complex systems [8, 40]. In a nutshell, architecture-centered TM methods like STRIDE require significant manual effort and expert knowledge to model the system under analysis with sufficient detail and extract meaningful security threats [19]. Scandariato et al. [27] assessed the performance of STRIDE through a descriptive study and concluded that, despite its lightweight approach, it is relatively time-consuming and prone to *false negatives* (i.e., threats that analysts fail to identify). Recent work by Mbaka et al. [19] investigated the role of DFDs and supplementary material in validating the completeness of a STRIDE-driven threat analysis. Overall, their findings suggest that case study descriptions and threat descriptions are sometimes more useful than DFDs for assessing the coverage of a given analysis output. Nevertheless, proper tool support remains a critical factor for the usability of TM techniques, especially in collaborative team settings [2].

(ii) GenAI-specific security challenges. As GenAI becomes increasingly central to modern software architectures, it introduces new security threats that require systematic identification and mitigation [37]. The OWASP Top 10 for LLM Applications [23] provides a suitable reference taxonomy for characterizing these threats, covering issues related to the use of malicious prompts, mishandling sensitive data, and adversarial model behavior. Prompt injection emerges at the top of the list, representing the use of carefully crafted inputs to override LLMs' instructions and trigger unintended or unsafe behavior. Next comes sensitive information disclosure, a critical security issue that can affect both users and the LLMs themselves, for instance, when personal information or technical details (e.g., fragments of LLMs' source code) are leaked in their output [6]. The use of third-party pretrained LLMs can also introduce severe upstream vulnerabilities (e.g., biased outputs or system failures) if their security is compromised. Such vulnerabilities are often the consequence of data or model poisoning attacks, in which malicious actors manipulate the training data or the model itself to influence the behavior of downstream applications [41, 5]. LLMs are also prone to producing inaccurate or flawed outputs, especially when applied to complex tasks such as source code development [4]. Hence, over-reliance on these systems and granting them excessive agency can introduce significant security risks, particularly when model outputs are accepted or executed without sufficient human oversight or validation [7].

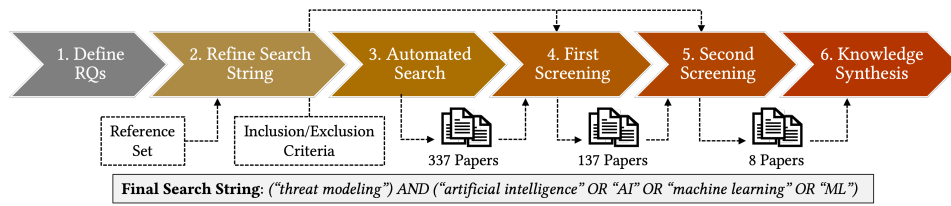


■ **Figure 1** Applied research methodology.

Recent TM approaches have begun incorporating AI- and GenAI-specific security threats into their analysis frameworks. Mauri and Damiani [17] extended STRIDE with AI-specific assets distributed across the ML lifecycle, namely data collection, model training, and deployment. Their method defines *failure models* for each asset category and derives the corresponding threats from violations of the associated security properties (e.g., authenticity, integrity, and confidentiality). Similarly, von der Assen et al. [36] introduced *ThreatFinderAI*, an approach geared towards contemporary AI and LLM-driven architectures in which assets such as pre-trained models and prompts are explicitly represented and analyzed. Threat identification is then supported through the construction of *attack graphs* based on prospective risk scenarios collected from multiple authoritative AI-security knowledge bases, including OWASP AI Exchange [24] and MITRE ATLAS [21]. Still, to the best of our knowledge, empirical evaluations of these and other methods alike remain limited, especially regarding their application to real-world scenarios and industrial settings. Moreover, little is known about practitioners' acceptance of these techniques, the effort required to integrate them into existing development workflows, and the practical frictions that may hinder their regular use.

3 Research Methodology

Fig. 1 illustrates the methodology we applied to address the RQs proposed in Section 1. We first conducted a Rapid Literature Review (RLR) to identify SotA TM techniques suitable for the analysis of security threats in GenAI-augmented systems (STEP 1). We shortlisted three candidate methods based on their maturity level (e.g., validation extent) and applied them to a case study provided by a local SME (STEP 2). The performance of each method was then assessed regarding its coverage of security threats included in the OWASP Top-10 for LLMs [23] relevant to such a case study (STEP 3). It is worth noting that the SME development team was kept informed of the progress at each method step before compiling the results (i.e., the RLR, the TM analysis, and the coverage assessment) into an Executive Report (ER). Once finalized, the team was then asked to review the ER and complete a survey on their perceptions of each method's suitability, the actionability of its outputs, and the anticipated integration effort in the current SME development setting (STEP 4). In the following subsections, we provide further details on each methodological step and the supporting artifacts employed throughout the study.



■ **Figure 2** Applied Rapid Literature Review (SLR) methodology (STEP 1).

3.1 Rapid Literature Review

Rapid Literature Reviews (RLRs) provide a lightweight framework for characterizing the SotA within a limited time frame while still following a structured and transparent process [25]. In contrast to more exhaustive evidence synthesis methods, RLRs typically involve narrower search scopes and simplified screening procedures, making them suitable for exploratory and early-stage studies. Given the emerging nature of GenAI-oriented TM research and the exploratory goals of this work, we considered an RLR appropriate for identifying candidate methods applicable to GenAI-augmented systems.

Fig. 2 illustrates the steps we conducted through the RLR. We first started defining the RQs that would guide the literature search. Particularly, we centered around the following RLR-specific RQ: “What threat modeling techniques documented in the current literature are suitable for the analysis of GenAI-augmented systems?”. Next, we defined the search engine and keyword string employed for the automated literature search. We selected Scopus¹ as the primary search engine due to its broad coverage of major software engineering publication venues, including ACM Digital Library and IEEE Xplore. To refine the search string, we constructed a small reference set of representative publications on TM for GenAI-augmented systems (i.e., [17, 36]) and iteratively adjusted the search terms until all reference studies were retrieved. The final query is depicted at the bottom of Fig. 2.

The automated search was conducted in June 2025 and returned 337 candidate studies, which were subsequently processed during the screening stages. We retained studies that (i) described architecture-centered TM techniques, (ii) considered security threats relevant to GenAI-based systems, (iii) were written in English, and (iv) were published at well-established software engineering or security venues. These inclusion and exclusion criteria were first applied to the title and abstract of the candidate studies during the initial screening stage, resulting in 137 potentially relevant papers. The same criteria were then applied to the full text of these studies during a second screening stage, ultimately leading to the selection of 8 primary studies that underwent a knowledge synthesis process. For this, we followed a lightweight open coding approach in which one author extracted emerging themes and patterns relevant to the proposed RLR-specific RQ. Another author reviewed these themes afterward and, if discrepancies arose, these were resolved through a negotiated agreement.

3.2 Methods Selection

The thematic analysis of the selected primary studies surfaced a set of recurring dimensions characterizing the identified TM methods. First, we analyzed the *System Type* targeted by each method, namely whether the technique was designed for traditional ML-based

¹ <http://www.scopus.com>

systems or more contemporary GenAI-oriented architectures involving LLMs and transformer-based components. Second, we examined the prescribed *Modeling Approach*, including the architectural representation and modeling artifacts employed throughout the TM process (e.g., traditional or custom DFD syntax). Third, we assessed the extent of *Tool Support*, particularly whether the method provided (semi-) automated support for threat identification. Finally, we considered the reported *Evaluation* strategy of each method, including proof-of-concept applications, case studies, and developer-centered usability evaluations.

■ **Table 1** Characterization of the selected primary studies.

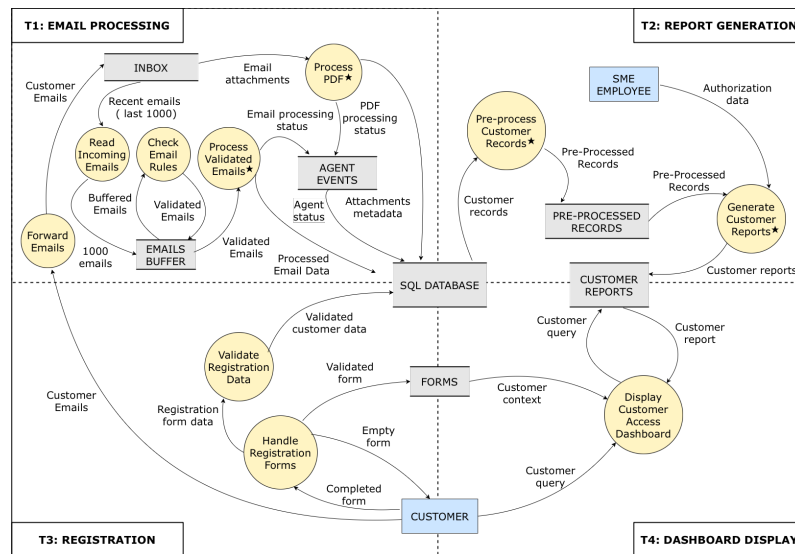
Source	Year	System Type	Modeling Approach	Tool	Evaluation
Von der Assen et al. [35]	2024	GenAI	Asset-Enriched DFD	YES	Case study
Von der Assen et al. [36]	2024	GenAI	Asset-Enriched DFD	YES	Case study
Mauri and Damiani [17]	2021	ML	Asset-Enriched DFD	NO	Case study
Mauri and Damiani [18]	2022	ML	Asset-Enriched DFD	NO	Case study
Wilhelm and Younis [38]	2020	ML	DFD	NO	Case study
Gülen et al. [10]	2024	GenAI	DFD	NO	Proof of concept
Messas et al. [20]	2024	ML	Unspecified	NO	Usability study
Kumar et al. [14]	2024	GenAI	DFD	NO	Case study

Table 1 summarizes the characterization of the eight primary studies. The rows highlighted correspond to the TM methods selected for their application to the case study. Our selection criteria centered on techniques (i) geared towards GenAI-enriched system architectures and (ii) validated through case studies. Since these criteria narrowed the selection to only two methods (i.e., [36, 10]), we decided to relax the latter and include one additional TM approach (i.e., [14]). This resulted in the following TM techniques being selected for application to the case study:

- **M1** – *AI-as-a-Service (AIaaS) Framework* [10]. Relies on DFD-based system representations and structures the analysis around asset groups such as *Data*, *Model*, *Environment*, and *Process*. For each asset category, the framework provides STRIDE-based threat listings along with prospective mitigation actions.
- **M2** – *Attacks on Dataset, Model and Input (ADMIn) Framework* [14]. It adopts an attack-centric perspective around three major attack surfaces, namely *Datasets*, *Models*, and *System inputs*. The analysis is structured through DFD-like architectural representations and predefined attack categories associated with each attack surface. The method further provides guidance on threat prioritization and mitigation activities.
- **M3** – *ThreatFinderAI* [36]. This approach extends traditional DFDs with AI-specific assets (e.g., prompts and pre-trained models), which are subsequently analyzed through attack graphs and risk scenarios. Threat identification is semi-automated through the integration of external security knowledge bases such as MITRE ATLAS [21] and OWASP AI Exchange [24]. Unlike M1 and M2, this method is tool-supported.

3.3 SME-Provided Case Study

Fig. 3 illustrates a DFD of the SME-provided case study. It describes the main architectural components (i.e., data stores, processes, data flows, and entities) of a business intelligence system that automatically processes e-mails from customers and generates customized market reports. It encompasses four major tasks, namely e-mail processing (T1), report generation (T2), customer registration (T3), and dashboard display (T4). Such tasks (T1 and T2 in particular) are partially supported by LLM agent processes (marked with a ★) in charge of (i) processing incoming emails and their attachments (i.e., PDF files), (ii) validating such emails



■ **Figure 3** Data Flow Diagram (DFD) of the analyzed case study.

(e.g., checking for duplicates), (iii) pre-processing internal records (e.g., customer profiles and historic data), and (iv) generating reports tailored to the requirements of each client. From a technological perspective, the system leverages Microsoft Power Platform² services for workflow orchestration and customer data management, while Azure OpenAI³ is employed to support the LLM-based processing components embedded within the architecture. The generated business intelligence reports and dashboards are further integrated with Power BI services for visualization and customer-facing reporting activities. Overall, the analyzed architecture comprises 10 processes, including 3 LLM-assisted components, as well as 7 data stores, 27 data flows, and 2 external entities.

3.4 Practitioner Survey

Seven members of the SME development team participated in a survey to identify potential adoption and integration challenges associated with the evaluated TM techniques. Particularly, they were first asked to review the content of the ER, namely a description of the three shortlisted TM methods, their output for the given case study, and their coverage regarding the OWASP Top-10 for LLMs. Then, we asked them to provide their perspectives on each method by assessing a set of statements across seven dimensions, including the method *effectiveness* (i.e., “The method helped identify important security threats for our system”), *actionability* (i.e., “The method provided clear mitigation actions for addressing the identified threats”), and *awareness* (i.e., “The method surfaced threats we would likely have missed otherwise”), among others. Each participant assessed each statement three times (i.e., for each method) using 6-point Likert values (i.e., *completely disagree*; *disagree*; *somewhat disagree*; *somewhat agree*; *agree*; *completely agree*). The survey concluded with the following open-ended questions aimed at collecting further insights into the added value, limitations, and adoption challenges of GenAI-aware TM techniques:

- **OE1:** What were the top 3 most valuable threats identified across the methods, and why?
- **OE2:** What would have to change for this method to be adoptable in your workflow?

² <https://www.microsoft.com/en-us/power-platform>

³ <https://azure.microsoft.com/en-us/products/ai-services/openai-service>

📁 All study materials are available in the paper’s **Replication Package** (Section 7), including the RLR protocol, the corresponding code book, the survey instrument, the aggregated survey results, and the OWASP Top-10 for LLMs mapping.

4 Results

In the following subsections, we report the findings obtained from the assessment of the selected TM methodologies (Section 4.2) and the practitioner survey (Section 5). We narrowed the former to the GenAI-aided processes within the system, along with their corresponding incoming and outgoing data flows. Thereby, we focus the analysis on the system components directly influenced by GenAI capabilities and, therefore, more likely to expose GenAI-specific security threats. In particular, we considered the following processes and data flows:

- **P1: PDF Processing**, with incoming flow *IF-1.1: Email attachments* and outgoing flows *OF-1.1: PDF processing status* and *OF-1.2: Attachment metadata*.
- **P2: Validated Email Processing**, with incoming flow *IF-2.1: Validated emails* and outgoing flows *OF-2.1: Processed email data* and *OF-2.2: Email processing status*.
- **P3: Customer Report Generation**, with incoming flows *IF-3.1: Authorization data* and *IF-3.2: Pre-processed records* and outgoing flow *OF-3.1: Customer reports*.
- **P4: Customer Record Pre-Processing**, with incoming flow *IF-4.1: Customer records* and outgoing flow *OF-4.1: Pre-processed records*.

4.1 RQ1: Effectiveness of Threat Modeling Techniques

Although all three TM techniques addressed some GenAI-related security threats, they differed substantially in terms of abstraction level, threat granularity, and analysis focus. M1 [10] adopts a STRIDE-inspired approach, while incorporating security threats traditionally associated with adversarial ML [15], including *model poisoning*, *model inversion*, *evasion*, *membership inference*, and *model extraction* attacks. In this context, *model inversion* and *membership inference* refer to attempts to infer sensitive information (e.g., data used for fine-tuning) through interactions with the LLM-based components, whereas *evasion attacks* aim to manipulate externally controlled inputs to alter model behavior or bypass intended processing logic. While *data poisoning* (i.e., injecting malicious or manipulated data into LLM-based workflows) is not explicitly represented in the taxonomy, it can still be operationalized through classical STRIDE categories such as tampering or manipulation during data transmission. As shown in Table 2, these threats, which emerge from all GenAI-aided processes and incoming data flows, largely align with two of the OWASP Top-10 threat categories for LLMs, namely *Sensitive Information Disclosure* and *Data and Model Poisoning*.

Compared to M1, M2 [14] more explicitly captures GenAI-specific runtime threats, particularly by incorporating *prompt injection* in addition to traditional adversarial ML attacks. These threats, which directly correspond to OWASP’s *Prompt Injection* category, were considered relevant to all analyzed GenAI-aided processes. Furthermore, the method identifies outgoing data flows as potential targets of membership inference attacks (referred to as “data exfiltration” attacks), which, similarly to M1, are conceptually related to OWASP’s *Sensitive Information Disclosure* category. Certain outgoing data flows, particularly operational status flows such as *OF-1.1* and *OF-2.2*, could not be naturally mapped to this category as they do not expose, in principle, semantically rich and potentially sensitive information.

■ **Table 2** Coverage of OWASP Top-10 for LLMs across the evaluated TM methods.

OWASP Top-10 for LLMs	M1	M2	M3
1. Prompt Injection	-	P1,P2,P3,P4	P1,P2,P3,P4
2. Sensitive Information Disclosure	P1,P2,P3,P4	OF-1.2,OF-2.1,OF-3.1	P1,P2,P3,P4,OF-1.2,OF-2.1,OF-3.1
3. Supply Chain Vulnerabilities	-	-	-
4. Data and Model Poisoning	P1,P2,P3,P4,IF-1.1,IF-2.1,IF-3.1,IF-3.2,IF-4.1	P1,P2,P3,P4,IF-1.1,IF-2.1,IF-3.1,IF-3.2,IF-4.1	IF-1.1,IF-2.1,IF-3.1,IF-3.2,IF-4.1,OF-1.2,OF-2.1,OF-3.1
5. Improper Output Handling	-	-	-
6. Excessive Agency	-	-	-
7. System Prompt Leakage	-	-	-
8. Vector and Embedding Weaknesses	N/A	N/A	N/A
9. Misinformation	-	-	-
10. Unbounded Consumption	P1,P3	P1,P3	P1,P3

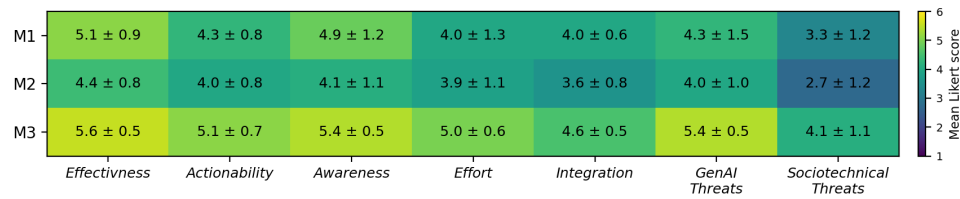
M3 [36] was the only approach providing dedicated tool support, together with a custom syntax and semantics for representing AI-related architectural components and threats. Consequently, the analysis was conducted on a different but semantically-equivalent DFD, requiring the identified threats to be mapped back to the original processes and data flows. Although this mapping was not always one-to-one, it remained relatively straightforward in practice (e.g., *data ingestion* were generalized to all incoming data flows), leading to results largely comparable to those of M2 (Table 2). Despite their differing abstractions and threat modeling strategies, all three methods identified processes *P1* and *P3* as potential targets of *Denial-of-Service* (*DoS*) attacks, mainly due to their reliance on computationally intensive LLM operations and externally controlled inputs. These threats are conceptually related to OWASP’s *Unbounded Consumption* category, as attackers may attempt to trigger excessive inference operations or resource exhaustion scenarios through malicious or oversized inputs. Nevertheless, the evaluated methods showed limited support for several OWASP Top-10 for LLMs categories, particularly those associated with agentic behavior, system prompt exposure, and unsafe downstream integration of generated outputs.

Q Findings RQ1. Although the evaluated TM techniques identified several GenAI-related issues, their coverage was strongest for input- and model-interaction threats (e.g., *Prompt Injection*, *Data and Model Poisoning*). Coverage was limited or absent for categories requiring broader architectural, supply-chain, or human-centered reasoning (e.g., *Supply Chain Vulnerabilities* and *Excessive Agency*). M3 provided the broadest support due to its tool-supported analysis and dedicated AI-oriented modeling syntax.

4.2 RQ2: Practitioner Perceptions

Fig. 4 shows participants’ average scores and standard deviations for each of the seven survey dimensions and TM methods. Overall, M3 received the highest scores across all dimensions, with particularly strong results regarding its capability to identify GenAI-specific security threats (*GenAI Threats*) and the effort required for regular use (*Effort*). This may be attributed to M3’s tool support and its incorporation of external, GenAI-specific security knowledge in the underlying threat catalog. Conversely, M1 and M2 received comparable scores regarding effort, which may be explained by their lack of automation.

Both M1 and M3 received relatively higher scores than M2 in terms of identifying system-relevant threats (*Effectiveness*) and security issues that participants would otherwise have overlooked (*Awareness*). This may be associated with the limited relevance of one of M2’s main attack surfaces (i.e., “Models”) in this particular case, since the SME does not develop



■ **Figure 4** Practitioner ratings across M1 (*AIaaS*), M2 (*ADMin*), and M3 (*ThreatFinderAI*).

its own ML models but rather relies on externally provided LLM services. Furthermore, both M1 and M3 provide a more fine-grained catalog enriched with detailed threat descriptions, affected assets, and, in the case of M3, also prescriptive appraisal mechanisms. This may also have affected practitioners' perception of M2's support for deriving clear mitigation actions (*Actionability*) and integrating the resulting outputs into existing development workflows (*Integration*), which were rated lower than those of M1 and M3.

In line with the mapping results (Section 4.2), all methods received their lowest scores in identifying socio-technical security threats. This is consistent with Table 2, where none of the evaluated methods surfaced threats related to OWASP's *Excessive Agency* category, which is the most closely related to socio-technical concerns in our analysis. Still, the slightly higher scores of M1 and M3 may be explained by their explicit consideration of potential threat actors, which can help practitioners reason about who may exploit a given threat.

When analyzing participants' answers to OE1, we observed that they consistently identified *prompt injection*, *data poisoning*, and *sensitive information disclosure* as the most valuable threats surfaced by the methods. These were perceived as particularly relevant because the system processes external, potentially untrusted inputs, while also handling sensitive customer information: “*Prompt injection was a real eye-opener. Data poisoning too – silent and sneaky. And data leakage through cloud APIs, honestly we wouldn't have caught that on our own.*” (P6). On the other hand, responses to OE2 emphasized that adoption would require lightweight integration into existing development routines, including ticketing workflows and clear ownership. Furthermore, they underscored the need for structured templates (e.g., checklists) and short training sessions on system modeling and GenAI security threats: “*Templates would make it so much easier. A quick onboarding session for the team. And just connect it to our tickets – that alone would make a huge difference.*” (P5).

Q Findings RQ2. Practitioners rated M3 highest across all survey dimensions, particularly for its support in surfacing *GenAI Threats* and its relatively low perceived *Effort*. M1 and M3 received higher scores than M2 for *Effectiveness*, *Awareness*, *Actionability*, and *Integration*. Across methods, *Socio-technical Threats* received the lowest scores, consistent with the RQ1 findings. Participants also emphasized the need for better integration with existing development workflows, especially issue-tracking and security review processes.

4.3 Threats to Validity

The findings of this study should be interpreted in light of its exploratory nature. *External validity:* The assessment was conducted on a single case study, which limits the generalizability of the results to other GenAI-augmented systems, domains, or organizational settings. *Construct validity:* The evaluated TM techniques differ in their abstraction levels, modeling assumptions, and expected inputs, making direct comparison challenging. To mitigate this issue, we normalized the analysis around GenAI-aided processes, their associated data flows,

and the OWASP Top-10 for LLMs as a common reference baseline. *Internal validity:* The mapping of method-specific threats to OWASP categories may have been influenced by the researchers' interpretation. Therefore, we followed a conservative mapping strategy and avoided assigning categories when the correspondence was not sufficiently clear. *Conclusion validity:* the practitioner survey involved a small number of participants from the involved development team, who assessed the methods based on the Executive Report rather than by applying the techniques themselves. Hence, the survey results should be understood as initial perceptions of usefulness, actionability, and adoption challenges rather than as evidence of actual long-term adoption in practice.

5 Prospective Research Pathways

In principle, the study results suggest that current TM techniques are useful for identifying GenAI-related threats, such as *prompt injection*, *data poisoning*, and *sensitive information disclosure*. However, they show limited support for some critical threat categories, particularly those directly associated with (i) the software supply chain and (ii) the interaction between LLMs and developers. As third-party GenAI solutions become an integral part of system architectures, TM methods should provide means for identifying security risks stemming from LLM dependencies [39]. Still, none of the three TM techniques we assessed offered a threat catalog and modeling syntax enriched with explicit supply-chain, GenAI-specific elements (e.g., third-party ML libraries, container images) that could help practitioners reason about such risks. Furthermore, in line with prior work [2], the survey results indicate that tool support should not be neglected, as M3 received the highest overall perception scores among participants. This motivates the following Research Pathway (RP):

✚ **RP1:** GenAI-aware TM approaches should explicitly incorporate knowledge of supply-chain security into their frameworks while supporting some degree of automation.

Prior work on the automatic extraction of security-enriched DFDs from source code [29] could, in principle, provide a promising basis for this direction. In particular, these extraction capabilities could be extended to cover supply-chain artifacts such as LLM-relevant dependency manifests, SBOMs, and container definitions. Alongside this, recent work by Jedrzejewski et al. [13] introduced a prompt-driven threat analysis concept that leverages LLMs with RAG to generate and maintain threat models over time. As vibe coding begins to displace traditional programming practices, such approaches could help align TM with emerging AI-assisted development workflows. This integration could be further supported by structured prompt templates and checklists, as suggested by participants in the survey:

✚ **RP2:** Future work should explore prompt-guided and checklist-supported workflows for integrating GenAI-aware TM into LLM-assisted development practices.

Finally, the limited coverage of human-centered security threats across all assessed TM techniques is concerning. Prior work has shown that LLMs may produce unreliable or hallucinated results under certain conditions; therefore, GenAI-aided processes require careful scrutiny before their outputs are integrated into downstream development activities. At the same time, multiple studies raise concerns about developers' over-reliance on such outputs and reduced critical engagement during their operationalization [7]. This calls for TM approaches that explicitly capture (i) where flawed LLM-generated content may propagate across GenAI-augmented architectures, (ii) which actors are responsible for validating or acting upon them,

and (iii) situations in which over-reliance may become security-threatening. Furthermore, suitable mitigation actions should go beyond traditional offline training and combine online behavioral interventions [3] with interactive, personalized security assistance [32] to support practitioners when engaging with LLM-generated outputs.

✚ **RP3:** GenAI-aware TM approaches should account for developers' over-reliance on LLM-generated outputs by modeling human-centered threat propagation paths and identifying intervention points across the system architecture.

6 Conclusion

TM remains a cornerstone of secure software engineering, enabling practitioners to systematically identify and mitigate security risks at the architectural level. However, as GenAI becomes increasingly embedded into modern software systems, traditional TM techniques require new abstractions and supporting mechanisms to account for the distinctive characteristics of LLM-based applications. In this paper, we presented emerging results from an exploratory assessment of three GenAI-aware TM techniques in an industrial SME case study. Our findings indicate that, although current techniques can capture prominent GenAI-related threats such as prompt injection and data poisoning, they provide only limited support for broader architectural, supply-chain, and human-centered risks.

Beyond threat coverage, our results highlight the importance of practical adoption. The survey results indicate that practitioners value TM techniques that are easy to integrate into existing development workflows while requiring minimal additional effort. As software development continues to evolve toward increasingly AI-assisted and prompt-driven practices, next-generation TM techniques should not only broaden their coverage of GenAI-specific risks but also facilitate their seamless incorporation into modern software development pipelines. Altogether, these findings suggest that the future of GenAI-aware TM depends on combining comprehensive threat coverage with lightweight, tool-supported, and actionable integration into software development practice.

7 Data Availability Statement

All study materials are available in the following link to the paper's **Replication Package**: <https://doi.org/10.5281/zenodo.21706295>. The package includes the RLR and extraction forms, the characterization of the selected TM methodologies, the survey instrument, aggregated survey results, and the OWASP Top-10 for LLMs mapping produced during the analysis.

References

- 1 Zaina Abuabed, Ahmad Alsadeh, and Adel Taweel. Stride threat model-based framework for assessing the vulnerabilities of modern vehicles. *Computers & Security*, 133:103391, 2023. doi:10.1016/J.COSE.2023.103391.
- 2 Karin Bernsmed, Daniela Soares Cruzes, Martin Gilje Jaatun, and Monica Iovan. Adopting threat modelling in agile software development projects. *Journal of Systems and Software*, 183:111090, 2022. doi:10.1016/J.JSS.2021.111090.
- 3 Chris Brown. Digital nudges for encouraging developer actions. In *2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion)*, pages 202–205. IEEE, 2019. doi:10.1109/ICSE-COMPANION.2019.00082.

- 4 Zhi Chen and Lingxiao Jiang. Evaluating software development agents: Patch patterns, code quality, and issue complexity in real-world github scenarios. In *International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 657–668, 2025. doi:10.1109/SANER64311.2025.00068.
- 5 Domenico Cotroneo, Cristina Improta, Pietro Liguori, and Roberto Natella. Vulnerabilities in ai code generators: Exploring targeted data poisoning attacks. In *Proceedings of the 32nd IEEE/ACM international conference on program comprehension*, pages 280–292, 2024. doi:10.1145/3643916.3644416.
- 6 Badhan Chandra Das, M Hadi Amini, and Yanzhao Wu. Security and privacy challenges of large language models: A survey. *ACM Computing Surveys*, 57(6):1–39, 2025. doi:10.1145/3712001.
- 7 Nicolás E. Díaz Ferreyra, Monika Swetha Gurupathi, Zadia Codabux, Nalin Arachchilage, and Riccardo Scandariato. Security Concerns in Generative AI Coding Assistants: Insights from Online Discussions on GitHub Copilot. In *Proceedings of the 2026 30th International Conference on Evaluation and Assessment in Software Engineering Companion*, 2026.
- 8 Daniele Granata and Massimiliano Rak. Systematic analysis of automated threat modelling techniques: Comparison of open-source tools. *Software Quality Journal*, 32(1):125–161, 2024. doi:10.1007/S11219-023-09634-4.
- 9 Kathrin Grosse, Lukas Bieringer, Tarek R Besold, and Alexandre M Alahi. Towards more practical threat models in artificial intelligence security. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 4891–4908, 2024.
- 10 Utku Gülen, Ömer Faruk Tuna, Boubakr Nour, Zakaria Laaroussi, Leyli Karaçay, and Ferhat Karakoç. Threat Modeling of AI-as-a-Service Framework. In *2024 20th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–7. IEEE, 2024. doi:10.1109/WIMOB61911.2024.10770432.
- 11 Michael Howard and Steve Lipner. *The security development lifecycle*, volume 8. Microsoft Press Redmond, 2006.
- 12 Felix Viktor Jedrzejewski. Threat modeling of ml-intensive systems: Research proposal. In *Proceedings of the IEEE/ACM 3rd International Conference on AI Engineering-Software Engineering for AI*, pages 264–266, 2024. doi:10.1145/3644815.3644975.
- 13 Felix Viktor Jedrzejewski, Davide Fucci, and Oleksandr Adamov. Thremolia: Threat modeling of large language model-integrated applications. In *Proceedings of the 29th International Conference on Evaluation and Assessment in Software Engineering*, pages 834–839, 2025. doi:10.1145/3756681.3757083.
- 14 Vimal Kumar, Juliette Mayo, and Khadija Bahiss. Admin: Attacks on dataset, model and input: A threat model for ai based software. In *Proceedings of the 10th International Conference on Information Systems Security and Privacy*, pages 170–178. SciTePress, 2024. doi:10.5220/0012394100003648.
- 15 Hsiao-Ying Lin and Battista Biggio. Adversarial machine learning: Attacks from laboratories to the real world. *Computer*, 54(5):56–60, 2021. doi:10.1109/MC.2021.3057686.
- 16 Pedro Lohmann, Carlos Albuquerque, and Raphael Machado. Systematic literature review of threat modeling concepts. In *Proceedings of the 9th International Conference on Information Systems Security and Privacy*, pages 163–173, 2023. doi:10.5220/0011783000003405.
- 17 Lara Mauri and Ernesto Damiani. STRIDE-AI: An Approach to Identifying Vulnerabilities of Machine Learning Assets. In *2021 IEEE International Conference on Cyber Security and Resilience (CSR)*, pages 147–154. IEEE, 2021. doi:10.1109/CSR51186.2021.9527917.
- 18 Lara Mauri and Ernesto Damiani. Modeling threats to AI-ML systems using STRIDE. *Sensors*, 22(17):6662, 2022. doi:10.3390/S22176662.
- 19 Winnie Bahati Mbaka, Xinran Zhang, Yunduo Wang, Tong Li, Fabio Massacci, and Katja Tuma. Assessing the usefulness of data flow diagrams for validating security threats. *Computers & Security*, 156:104498, 2025. doi:10.1016/J.COSE.2025.104498.
- 20 Gabriel Esteves Messas, Rodrigo Sanches Miani, and Bruno Bogaz Zarpelão. sAIf: Towards a Lightweight Threat Modeling Approach to Support Machine Learning Application Development. In *Proceedings of the XXIII Brazilian Symposium on Software Quality*, pages 1–10, 2024. doi:10.1145/3701625.3701640.

- 21 MITRE Corporation. MITRE ATLAS. <https://atlas.mitre.org>, 2026.
- 22 Nitin Naik, Paul Jenkins, Paul Grace, Dishita Naik, Shaligram Prajapat, and Jingping Song. A comparative analysis of threat modelling methods: Stride, dread, vast, pasta, octave, and linddun. In *The international conference on computing, communication, cybersecurity & AI*, pages 271–280. Springer, 2024. doi:10.1007/978-3-031-74443-3_16.
- 23 OWASP Foundation. OWASP Top 10 for Large Language Model Applications. <https://genai.owasp.org/llm-top-10/>, 2025.
- 24 OWASP Foundation. OWASP AI Exchange. <https://owaspai.org>, 2026.
- 25 Sebastián Pizard, Joaquín Lezama, Rodrigo García, Diego Vallespir, and Barbara Kitchenham. Using rapid reviews to support software engineering practice: a systematic review and a replication study. *Empirical Software Engineering*, 30(1):10, 2025. doi:10.1007/S10664-024-10545-6.
- 26 Olaf Saßnick, Thomas Rosenstatter, Christian Schäfer, and Stefan Huber. Stride-based methodologies for threat modeling of industrial control systems: A review. In *IEEE 7th International Conference on Industrial Cyber-Physical Systems*, pages 1–8, 2024. doi:10.1109/ICPS59941.2024.10639949.
- 27 Riccardo Scandariato, Kim Wuyts, and Wouter Joosen. A descriptive study of microsoft’s threat modeling technique. *Requirements Engineering*, 20(2):163–180, 2015. doi:10.1007/S00766-013-0195-2.
- 28 Simon Schneider, Nicolas E Diaz Ferreyra, Pierre-Jean Queval, Georg Simhandl, Uwe Zdun, and Riccardo Scandariato. How Dataflow Diagrams Impact Software Security Analysis: An Empirical Experiment. In *2024 IEEE International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 952–963. IEEE, 2024.
- 29 Simon Schneider and Riccardo Scandariato. Automatic extraction of security-rich dataflow diagrams for microservice applications written in java. *Journal of Systems and Software*, 202:111722, 2023. doi:10.1016/J.JSS.2023.111722.
- 30 Laurens Sion, Koen Yskout, et al. Security Threat Modeling: Are Data Flow Diagrams Enough? In *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, pages 254–257, 2020.
- 31 Kamakshi Srikumar, Komal Kashish, Kolja Eggers, Nicolás E Díaz Ferreyra, Julian Koch, Thorsten Schüppstuhl, and Riccardo Scandariato. Striped: A threat analysis method for iot systems. In *Proceedings of the 17th International Conference on Availability, Reliability and Security*, pages 1–6, 2022. doi:10.1145/3538969.3538970.
- 32 Catherine Tony, Mohana Balasubramanian, Nicolás E. Díaz Ferreyra, and Riccardo Scandariato. Conversational devbots for secure programming: An empirical study on skf chatbot. In *Proceedings of the 26th International Conference on Evaluation and Assessment in Software Engineering*, pages 276–281, 2022. doi:10.1145/3530019.3535307.
- 33 Katja Tuma, Christian Sandberg, Urban Thorsson, Mathias Widman, Thomas Herpel, and Riccardo Scandariato. Finding security threats that matter: Two industrial case studies. *Journal of Systems and Software*, 179:111003, 2021. doi:10.1016/J.JSS.2021.111003.
- 34 Dimitri Van Landuyt and Wouter Joosen. A descriptive study of assumptions in stride security threat modeling. *Software and Systems Modeling*, 21(6):2311–2328, 2022. doi:10.1007/S10270-021-00941-7.
- 35 Jan Von der Assen, Alberto Huertas, Jamo Sharif, Chao Feng, G r me Bovet, and Burkhard Stiller. ThreatFinderAI: Automated Threat Modeling Applied to LLM System Integration. In *2024 20th International Conference on Network and Service Management (CNSM)*, pages 1–3. IEEE, 2024.
- 36 Jan von der Assen, Jamo Sharif, Chao Feng, Christian Killer, G r me Bovet, and Burkhard Stiller. Asset-centric threat modeling for ai-based systems. In *2024 IEEE international conference on cyber security and resilience (CSR)*, pages 437–444. IEEE, 2024. doi:10.1109/CSR61664.2024.10679445.

- 37 Shang Wang, Tianqing Zhu, Bo Liu, Ming Ding, Dayong Ye, Wanlei Zhou, and Philip Yu. Unique security and privacy threats of large language models: A comprehensive survey. *ACM Computing Surveys*, 58(4):1–36, 2025. doi:10.1145/3764113.
- 38 Carl Wilhelm and Awad A Younis. A threat analysis methodology for security requirements elicitation in machine learning based systems. In *2020 IEEE 20th international conference on software quality, reliability and security companion (QRS-C)*, pages 426–433. IEEE, 2020. doi:10.1109/QRS-C51114.2020.00078.
- 39 Laurie Williams, Giacomo Benedetti, et al. Research Directions in Software Supply Chain Security. *ACM Transactions on Software Engineering and Methodology*, 34(5):1–38, 2025. doi:10.1145/3714464.
- 40 Wenjun Xiong and Robert Lagerström. Threat modeling—a systematic literature review. *Computers & security*, 84:53–69, 2019. doi:10.1016/J.COSE.2019.03.010.
- 41 Quan Zhang, Chijin Zhou, Gwihwan Go, Binqi Zeng, Heyuan Shi, Zichen Xu, and Yu Jiang. Imperceptible content poisoning in llm-powered applications. In *Proceedings of the 39th International Conference on Automated Software Engineering*, pages 242–254, 2024. doi:10.1145/3691620.3695001.