# Usefulness of data flow diagrams and large language models for security threat validation: a registered report

Winnie Bahati Mbaka
w.mbaka@vu.nl
Vrije Universiteit Amsterdam
The Netherlands

Katja Tuma
k.tuma@vu.nl
Vrije Universiteit Amsterdam
The Netherlands

## ABSTRACT

The arrival of recent cybersecurity standards has raised the bar for security assessments in organizations, but existing techniques don't always scale well. Threat analysis and risk assessment are used to identify security threats for new or refactored systems. Still, there is a lack of definition-of-done, so identified threats have to be validated which slows down the analysis. Existing literature has focused on the overall performance of threat analysis, but no previous work has investigated how deep must the analysts dig into the material before they can effectively validate the identified security threats. We propose a controlled experiment with practitioners to investigate whether some analysis material (like LLM-generated advice) is better than none and whether more material (the system's data flow diagram and LLM-generated advice) is better than some material. In addition, we present key findings from running a pilot with 41 MSc students, which are used to improve the study design. Finally, we also provide an initial replication package, including experimental material and data analysis scripts and a plan to extend it to include new materials based on the final data collection campaign with practitioners (e.g., pre-screening questions).

## KEYWORDS

STRIDE, Data Flow Diagrams, Large Language Models, Threat Validation, Empirical Software Engineering

**ACM Reference Format:**
Winnie Bahati Mbaka and Katja Tuma. 2018. Usefulness of data flow diagrams and large language models for security threat validation: a registered report. In *Proceedings of 18th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM'24).* ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

## 1 INTRODUCTION

Building secure software is a global concern that has in recent years spurred new regulations (e.g., EU Cybersecurity and Cyber-resilience Acts, and the US Cloud Act). CISA and 17 U.S. and international partners [27] recommend planning for countermeasures and reduce the risk of costly security breaches down the line, and for safety-critical systems, such as systems developed in the transportation sector, recent standards even require conducting a threat analysis of entire products (ISO/SAE 21434:2021 [19]). In light of the global gap in the cybersecurity workforce, these demands can be disruptive for organizations.

Threat analysis and risk assessment approaches help to elicit critical security threats and identify appropriate countermeasures. A myriad of threat analysis and risk assessment techniques already exists [41]. STRIDE [34] adopts a graphical representation of the software architecture under analysis, the *Data Flow Diagram* (DFD) [11, 35], which is simple and easy to learn [33]. DFD-like models (in essence, directed typed graphs) are extensively used in practice, for instance, in the automotive industry [24], at Microsoft [34], and agile organizations [4]. In a threat analysis session, security and domain experts explore the DFD (and the available material) to identify potential security threats through brainstorming [34]. But this process is time-consuming [33], and the resources dedicated to security are scarce in organizations. In addition, threat analysis lacks completeness guarantees about the identified security threats [9, 41]. Without enough objective measures for threat correctness and completeness [25], predicting threats for software that has not been built yet demands practitioners to make risk-based decisions under uncertainty [5]. In other words, practitioners often waste time revisiting the analysis material and *validating threat feasibility* [42] to assert that they have done a "complete job" and that they have not overlooked any important security threat. For example, practitioners may quickly assess the threat of SQL-injection on an exposed web interface as correct by reading the threat description and attack scenario, while for more domain-specific threats, they may have to revisit analysis material (such as the DFD, the requirements of the system, general or domain-specific security catalogues, such as MITRE ATT&CK knowledge base[1]).

Previous attempts to automate threat analysis have limitations [41], as the underlying system under analysis is under-specified at design-time. Large Language Models (LLMs) have recently been studied for providing security advice and summarizing information [7]. As an alternative to manual use of general security catalogs, practitioners could adopt LLMs as Model-in-the-loop [14], where the human decides based on summarized advice of the LLM. In fact, tools for leveraging advice from LLMs for STRIDE are emerging [2, 6]. For example, a customized GPT [6] provides suggestions for prompts, which generate a list of potential security threats. Even if in-house models are deployed in a secure environment, LLMs tend to hallucinate [16], so the generated list of threats may be eventually used as *additional* analysis material.

*ESEM'24, September 03–05, 2024, Barcelona, Spain*

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

[1] https://attack.mitre.org

*So, how deep must the analyst dig into the material before they can effectively validate security threats?*

We plan to investigate the usefulness of analysis material for threat validation, with the goal of understanding if reading "some" additional analysis material is better than "none" and if "more" available analysis material is better than "some". Since practitioners tend to refer to (and revisit) the DFD during threat validation by assessing threat priority and feasibility [42], it is interesting to include the presence of a DFD for this task as an intervention. In addition, having the DFD available could potentially help the practitioners in assessing the LLM advice, therefore it is also interesting to observe the presence of both DFD and LLM advice.

Due to known issues with hallucinations [16], we expect that if participants trust the first generated advice, this can increase the number of false positives. On the other hand, we expect that participants not fully trusting the first generated advice may perform better. Our results could inform practitioners about the minimal analysis material required for effective threat validation, and provide insights to practitioners that are on the fence about adopting LLMs for threat analysis.

We present a balanced design for a controlled experiment (approved by the Ethics Board of the primary institution conducting the research), a pilot study with 41 MSc students, and a plan for execution with practitioners. We built a ground truth using the Microsoft STRIDE [34] threat analysis approach on two scenarios (GitHub repository update, and pod deployment on K8), and since the final intervention is ordinal, we plan to conduct a statistical analysis using Helmert contrasts.

### Data availability statement

For replicability, we have provided an initial replication package[2] including a sample analysis script, the entry questionnaire, the experiment survey, both scenario descriptions, and a list of threats for each scenario. We plan to extend it in the final paper to include pre-screening questions for recruiting industry practitioners.

## 2 RELATED WORKS

### 2.1 Empirical Research of Threat Analysis

Several studies [25, 33] have empirically investigated the effectiveness of Microsoft's threat model, STRIDE. Both Scandariato et al [33] and Mbaka et al. [25] replicated a controlled experiments that sought to measure the productivity, precision, and recall of STRIDE.

Mbaka et al. [25] compared the productivity and precision of two STRIDE techniques, per-element and per-interaction. In the case of Mbaka et al. [25] STRIDE-per-interaction teams performed better than their counterparts. However, no significant location shift between the productivity and precision of the two variants was reported.

Other studies have investigated the challenges of adopting STRIDE in agile software development [4, 9]. Bernsmed et al. [4] conducted interviews with employees from four organizations that use agile software development practices and observed several challenges for adoption. Yet, practitioners were in agreement that performing a threat analysis leads to a more secure product.

Apart from STRIDE, other threat analysis techniques, have been empirically investigated. Two studies have compared the effectiveness of attack trees and misuse cases [21, 29]. Karpati et al. [21] observed that attack trees resulted in the identification of a higher number of threats compared to misuse cases, albeit the difference was not statistically significant. Although the experimental set-up implemented by Opdahl et al. [29] was different from that of Karpati et al. [21] (student participants vs industry practitioners) similar observations of better performance when using attack trees were also reported in the former [29].

Diallo et al. [12] compared the applicability of three techniques (i.e., the common criteria, attack trees, and misuse case) applied to wireless hotspots. The authors observed that each technique has strengths and weaknesses. For instance, while common criteria were complex to learn, they were easy to analyze. On the other hand, misuse cases had easier learnability, but its output was not easy to read.

Similar to [25, 33] we adopt the STRIDE methodology to measure the correctness of validating identified security threats. In contrast, instead of tasking the participants with developing a DFD, which is challenging [25], we present them with the same DFD in the experimental material. In addition, none of the empirical studies on threat analysis [12, 21, 25, 29, 33] have investigated how to support human reasoning around security threat validation.

### 2.2 Application of LLMs in Security

Related work has evaluated the usefulness of LLMs in the detection of code and software vulnerabilities [8, 28, 37, 39, 40]. Omar et al. [28] proposes a vulnerability detection framework, VulDetect leveraging the strengths of three models (GPT-2, BERT, and LSTM) to detect C and C++ source code vulnerabilities. The authors observed that VulDetect was able to outperform other state-of-the-art vulnerability detection techniques (SyseVR and VulDeBert) with a 92.65% accuracy.

Cheshkov et al. [8] sought to investigate the effectiveness of large language models (ChatGPT and GPT-3) in vulnerability detection.

In the study by Sun and colleagues [37] GPTScan (a combination of GPT and static analysis) was proposed and used to detect logic vulnerabilities in smart contracts.

Szabo and Bilicki [39] evaluated the efficacy of the GPT-3.5 and GPT-4 models in detecting the CWE-653 weakness, identifying sensitive data, and determining the protection levels of front-end applications.

Chen and colleagues [7] investigated the ability of two LLMs (ChatGPT and Bard) to refute popular security and privacy misconceptions. The study reported that LLMs report false positives (supporting misconceptions) which increase when the misconceptions are repeatedly queried. The study also reported on LLM hallucination, providing invalid URLs to support their output [7].

[2] https://anonymous.4open.science/r/LLM-for-threat-validation-298B/README.md

Despite some benefits of LLMs in discovering and assessing security vulnerabilities [7, 8, 28, 37, 39, 40], previous research has not investigated their use for early security design analyses, such as finding security weaknesses [39], and no previous work investigated their use for security threat validation.

## 3 RESEARCH QUESTIONS

Our research is motivated by previous findings pointing to the challenges with threat analysis reproducibility [25, 36] and lack of definition-of-done [9, 15]. A significant challenge is determining the feasibility of the identified threats, or in other words, validating the identified threats, an activity that can take place several times during the process and has been measured to cause detours and slow down the progress [17, 42]. Threat validation is not only an issue at the design phase, but also during threat intelligence gathering [18], where the collected data is also tainted by uncertainty.

On the other hand, previous research has investigated the effectiveness of providing additional textual or graphical material to aid in the comprehension of functional requirements [1] and safety compliance needs [10]. We postulate that understanding what material (if any) should be used to effectively validate threats may help in deriving threat feasibility faster.

To this end, we formulate the first research question:

**RQ1: What is the actual usefulness of having additional material like DFD or LLMs during threat validation**?

To measure the actual usefulness of having additional analysis materials (DFDs, LLMs, or both), we define several treatment groups. First, we consider the performance of participants validating threats without any additional material. To this end, we first compare their performance to those who received some (a DFD, or an LLM). Second, we compare the performance of those who received some material to those who received both a DFD and LLM. We hypothesize that actual usefulness should be different between groups that did not receive additional materials to those that did. That is, having some additional material (a DFD, or LLM) raises the performance of participants. In addition, having both a DFD and LMM should increase the participant's ability to correctly identify realistic threats. We therefore propose the following alternative hypothesis;

$H_{performance}$: *There is a statistically significant difference in the actual usefulness (i) between participants assessing the validity of threats without additional materials to those with some (either a DFD, or an LLM) and (ii) between participants assessing the validity of threats with some additional materials (DFD or LMM) to those with both (DFD and LLM).*

Second, some previous research measured differences [23] in the perceived usefulness of graphical models compared to textual information for security risk assessment. But, Labunets et al. [22] found that tabular and graphical methods are statistically equivalent to each other with respect to the actual and perceived efficacy. Following up on this result, we also measure perceived usefulness of the material provided to support threat validation and expect to confirm equivalence in our study.

**RQ2: What is the perceived usefulness of the additional material during threat validation?**

To this end, we check for statistical equivalence using the alternative hypothesis formulated below;

$H_{equiv-perc-both}$: *When given both the DFD and LLM, their perceived usefulness is statistically equivalent.*

Second, the sample means of the treatment group that received only the DFD will be compared to the one asked to assess the correctness of threats using only LLM. We formulate an alternative hypotheses;

$H_{equiv-perc-isolation}$: *The perceived usefulness of DFDs and LLMs when used in isolation is statistically equivalent.*

## 4 RECRUITMENT AND ETHICAL CONCERNS

### 4.1 Pilot study

We define two target populations for this study. First, for the pilot study, we recruited Master Computer Science students attending courses taught by the experimenters. To ensure that the students understand the experimental objects necessary to conduct the study we conducted a 4h30min training, evaluated their understanding of the training material, and included attention checks to filter dishonest responses.

### 4.2 Participant Recruitment

We plan to conduct the study with professionals recruited from crowd-sourcing platforms (e.g., Upwork, or Prolific). To this end, we will pre-screen and recruit participants with a background in software development, cybersecurity risk management. Since threat modeling is carried out by domain experts, such as developers, software architects, and security specialists [9], we consider such sample as representative.

In addition, we are conducting a think-aloud protocol with a few recruited practitioners as a second, more qualitative pilot. The results of the think-aloud will be transcribed and analysed for potential issues and concerns raised by the practitioners that need to be addressed before the final data collection.

### 4.3 Ethical Concerns

This experiment has received ethical approval from the ethical board of the institution under review number 2024-013. First, the study will provide an opt-in consent ("yes"/"no") form. Second, we do not anticipate any potential risk to the participants or researchers. Third, the study utilises GDPR-compliant tools to collect data. In addition, any personally identifiable information will be removed before data analysis. Fourth, for the pilot, the study was conducted as part of a course, and participants were only incentivised with a participation point which has a very small effect on their final grade. Importantly, students who did not provide consent for the analysis of their data also received a participation point. Finally, the participants were debriefed on the process of the experiment and the artefacts used.

## 5 METHODOLOGY

This section presents the design of the experiment including the research plan.

|  |  | Task (× 2) |  |  |
|---|---|---|---|---|
|  | Groups | DFD | LLM | Scenario |
| LLM+DFD | Group A | ✓ | ✓ | GH,K8 |
| DFD | Group B | ✓ | - | GH,K8 |
| LLM | Group C | - | ✓ | GH,K8 |
| / | Group D | - | - | GH,K8 |

**Table 1: Full experimental design used in the pilot.**

## 5.1 Experimental design

Assigning each participant to a different condition (noLLM, LLM) x (noDFD, DFD) x (GH, K8) would require $(2)^3$ groups. To avoid such a huge number of groups we will make use of a balanced orthogonal design which is also known as Taguchi Design [20]. Each participant will be randomly assigned to one of the four groups:

(1) LLM + DFD (A) receives the scenario descriptions with an accompanying data flow diagram instance and tasked with assessing the applicability of threats using an LLM
(2) noLLM + DFD (B) receives the scenario descriptions with an accompanying data flow diagram instance and tasked with self-assessing the applicability of threats
(3) LLM + noDFD (C) receives the scenario description without an accompanying data flow diagram instance and tasked with assessing the applicability of threats using an LLM
(4) noLLM + noDFD (D) receives the scenario description without an accompanying data flow diagram instance and tasked with self-assessing the applicability of threats

## 5.2 Experimental Objects

To ensure the objectives of the study are met, we prepared several experimental objects.

*Scenario selection.* First, we run the study with two scenarios to increase generalizability. One is based on updating a remote repository on GitHub and one on deploying a pod on Kubernetes, two common tasks in software development. In addition, both scenarios are inspired by real open-source platforms.

*LLM selection.* Several Large Language Models have been developed including Google's Gemini, OpenAI's chatGPT, GitHub's Copilot, and Microsoft's Copilot among others. Since the aim of our study is not to train an LLM or to benchmark different LLMs, but rather investigate their usefulness for helping human analysts in validating threats, we plan to leverage an open-source model. To this end, we opted for chatGPT-3.5 turbo model, as a first step.

*Training (4h30min).* For the pilot, the experimenters have prepared a lecture on threat modeling process and landscape (2h), a training lecture with a deep dive into security threats, STRIDE, DFD, and scenarios[3] (2h), and a walk-through presentation (30min) detailing each step of the task. The training was delivered in the beginning of the week on two consecutive days, and was also made available as recordings. The walk-through offers more understanding of what is to be expected during the actual experiment. For instance, groups A and C's walk-through includes an example of prompting the LLM to assess the correctness of the threat.

*Ground Truth.* The ground truth has been developed systematically by the authors. Half of the threats presented to the participant are correct and feasible, and half are bogus or incorrect. We define a bogus threat as the creation of a false claim to the existence of a potential security risk. Below is an example of a bogus threat:

*Scenario:* Updating a remote repository on GitHub
*Threat description:* An unauthenticated and non privileged attacker can still submit custom code into the remote repository to prepare the first step of another attack (e.g., turning off logging service or cause a Denial of Service).
*Assumption:* The attacker can reach the remote repository (e.g. through internet).
*STRIDE category:* Elevation of privilege and Tampering
*Location (in DFD):* The remote code repository

Explanation: GitHub allows owners of repositories to specify branch protection rules, which essentially disables 'force push' to the matching branches and prevents the matching branches from being deleted. When branch protection rules are implemented, an attacker cannot submit a custom code.

*Measures of Success.* Table 2 presents all the variables we consider in our experiment. This study considers two independent variables (or intervention), i.e., providing a DFD as part of the handout material and asking participants to perform the task of the experiment using an LLM. To achieve the aims of this study, we first plan to measure the background experience of participants in relation to the experimental objects of the study. In this case, participants will be required to self-report on their prior experience with secure design techniques, software design models, and their usage of LLMs, GitHub, and cloud deployment platforms. The responses to their background experience will be captured either on a 5-point Likert scale or using predefined multiple-choice options.

Second, to answer our first research question, we will analyse participants' performance , against the ground truth. The four possible outcomes of performance are discussed below;

(1) True Positive (TP), correctly identified realistic threats
(2) True Negative (TN), correctly identified bogus threats
(3) False Positive (FP), bogus threats selected as being real
(4) False Negative (FN), real threats that were considered bogus and therefore not selected

Third, we measure the perceived usefulness of graphical models and LLMs in threat validation. To this end, participants will be asked to what extent, on a 5-point Likert scale[4], do they think the additional analysis materials were useful in assisting them to correctly identify realistic threats. The responses to this question will be used to answer our second research question.

Lastly, we include several control questions to account for the varying levels of comprehension of the experimental objects among the participants. These control measures consist of questions about participants understandability of the the task, the sufficiency of time allocated to complete the task alongside the sufficiency of the training materials. In addition, we include several attention and

[3] The training delivered to groups A and B also included a DFD of the scenario.

[4] Where point 1 is labeled "strongly disagree", point 3 is labeled "neutral", and point 5 is labeled "strongly agree".

background checks for each of our target populations (see subsection 4.1). For student participants, we will measure their understanability of the experimental objects via attention checks. For industry practitioners, we plan to have several checks to ensure that they have a technical background. Finally, we collect the prompt history for each participant to control for prompt variability.

# 6 PILOTS

## 6.1 Execution

We present the steps taken to execute the pilot study.

Each participant $p$ joining the experiment;

(1) was randomly assigned to one of the four treatment groups: A, B, C, and D, see subsection 5.1.
(2) was presented with two scenario descriptions, one on modifying and updating repositories on GitHub (GH) and the other on pod deployment on Kubernetes (K8). We configured the survey tool[5] such that the presentation of the scenarios is randomised. That is, for two participants in the same groups, one received the Kubernetes scenario first followed by the GitHub scenario, and vice versa for the second participant.
(3) was presented with a list of threats (five bogus and five realistic threats) to each scenario description.
(4) was tasked with assessing the correctness of each threat and select the threats considered as realistic (likely to occur).

Each threat was accompanied by a threat description, assumptions, the associated STRIDE threat category that would be compromised if the attack occurred, and affected components. For each threat marked as realistic, participants were required to provide a short justification as to why they think it is a real threat.

When assessing the validity of security threats using LLMs (groups A and C), participants were allowed the freedom to prompt the LLM as they would in a real-world scenario. We provided them with the body of the prompt (threat description, assumption, STRIDE threat type, and affected components) to be posted on the LLM and asked them to prompt (in their own words) the LLM for assistance. We recorded the entire LLM interaction for each security threat selected as realistic by participant.

## 6.2 Preliminary results

*Demographics.* In total, 41 participants joined the pilot study, each participant received both scenarios, so we collected in total 82 responses. Before the training, about 2/3 reported to have used GitHub in a professional capacity (13) or during an internship (12), and more than 2/3 reported having attended either a few lectures (18) or a full course on Kubernetes (10). About half reported (22) to be new to the topic of secure design and most of the participants in groups A and C reported using an LLM several times a day.

*Results.* Groups that were tasked with validating security threats with the help of an LLM performed slightly better (the means for correctly identifying security threats was slightly higher in groups A ($\mu$TP_A= 8.1) and C ($\mu$TP_C= 9.4) compared to groups B ($\mu$TP_B= 6.0) and D ($\mu$TP_D= 7.4)). However, higher false positives were also reported in groups with access to an LLM (A ($\mu$FP_A= 7.4) and C ($\mu$FP_C= 4.7) compared to groups B ($\mu$FP_B= 3.5) and D ($\mu$FP_D= 3.1)). Similar observations (incorrectly assessing security information) have also been reported in prior studies on security misconceptions [7]. Chen and colleagues [7] reported that LLMs incorrectly support popular security and privacy misconceptions.

*Preliminary observations.* In addition to the descriptive statistics, we inspected the responses manually and made some interesting observations. However, the sample in the pilot is relatively small and these observations need to be validated with practitioners in the final study.

Interestingly, the group with no additional material (D) reported on average the least number of FPs but still reported a relatively high number of correct threats (7.4 out of 10). If this finding is validated with practitioners, this would indicate that less analysis material is actually needed to validate threats.

We investigated the responses of participants who received LLM advice (A, C) and observed that all participants in these groups took the first advice generated by LLM and also reported lower levels of previous knowledge about security (10 were novice to security, 9 attended some lectures, and only 2 had some hands-on experience). This could potentially also explain the higher numbers of FPs due to LLM hallucinations in those groups. In addition, when asked about the perceived usefulness of LLMs in assessing the validity of security threats after each scenario. We observed that 10/21 participants strongly agreed or agreed (points 5 and 4 on the Likert scale) with LLM's perceived usefulness concerning the threats relating to the GitHub scenario. On the other hand, 14/21 participants strongly agreed or agreed with LLM's perceived usefulness concerning the threats relating to the Kubernetes scenario. Thus, for security novices (e.g., practitioners in training), the use of LLM advice may still help in terms of recall (not overlooking threats) but not precision (due to a higher chance of false positives).

We also found that the presence of DFD did not make a significant difference in the collected measures of TN, FP, FN. Further, no major difference was observed for TPs when comparing the groups C and D to A and B (t-test of $\alpha$=0.05 returned a p-value of 0.048 and the Pearsons' correlation statistic was 0.5). We conclude that no strong positive or negative correlation exists between the availability of a DFD and the ability to correctly identify realistic threats.

This observation indicates that, instead of the full factorial design from Table 1, it is important to observe the following progression: performance differences when no material is given vs when LLM is used vs when DFD is given and LLM is used.

# 7 DATA ANALYSIS PLAN

Once the full data collection with practitioners has been completed, the results will be aggregated and statistically analysed.

*RQ1.* We aim to analyse the data using the Helmert contrast, a statistical analysis used to determine the smallest shift in location of the intervention from the control treatment [32]. We formulate the problem for difference as;

(1) noDFD&noLLM vs (noDFD&LLM union DFD&LLM)
(2) noDFD&LLM vs DFD&LLM

[5] www.qualtrics.com/

**Table 2: Experimental Variables**

| Name | Description | Operationalization |
|---|---|---|
| *Independent variables (design)* | | |
| DFD | Receiving a DFD to support the threat validation | Nominal (*) |
| LLM | Receiving LLM API to support the threat validation | Nominal (*) |
| *Background experience variables* | | |
| Secure design | Self-reported experience with secure design techniques | Ordinal scale (†) |
| Modeling | Self-reported familiarity with design models | Ordinal scale (‡) |
| Use of LLM | Self-reported frequency and the reason for using LLMs | Ordinal scale (†) |
| Scenario 1 | Self-reported experience with GitHub | Ordinal scale (†) |
| Scenario 2 | Self-reported experience with cloud deployment platforms | Ordinal scale (†) |
| *Dependent variables* | | |
| Performance of threat validation | Participants performance evaluated against a ground truth | Interval scale (≡) |
| Perception of additional materials | Self-reported perceived usefulness of additional analysis materials | Ordinal scale (†) |
| *Control measures* | | |
| Understanding | Self-reported understanding of what the task required them to do | Ordinal scale (‡) |
| Time | Self-reported sufficiency of the allotted time | Ordinal scale (‡) |
| Training | Self-reported sufficiency of training | Ordinal scale (‡) |
| Attention checks | To account for participants understanding of the experimental objects | Interval scale (≡) |
| Background checks (practitioners) | To account for the professional background of industry practitioners recruited from crowd-sourcing platforms | Interval scale (≡) |

(*) Qualtrics configured to automatically randomise the allocation of the independent variables.
(†) Multiple choice: For experience - attended some lecture, attended a full course, short internship, professional engagement.
For frequency- never, several times a day, once a week, few times a month, once in 3 months, or "other-asked to specify".
(‡) Responses captured on a 5-point Likert scale.
(≡) Responses evaluated against a predefined ground truth

To this end, if noDFD&LLM union DFD&LLM is greater than the control group (noDFD&noLLM), then *"some"* additional analysis material may improve the effectiveness of threat validation than having no materials at all. Similarly, if DFD&LLM is greater than noDFD&LLM, then both graphical models and LLMs may increase the actual effectiveness of threat validation as opposed to only having access to one of the additional analysis materials.

*RQ2.* Since our data for measuring perception of usefulness is ordinal and may not be normally distributed, we plan to use Mann Whitney U (MWU) with a level of significance equal to 0.05 ($\alpha = 0.05$) to test both equivalence and difference. We formulate the problem for testing the statistical equivalence as;

$$\begin{aligned} p_{low} &= MWU(\{x - \delta | x \in A\}, B, alt =' less') \\ p_{up} &= MWU(B, \{x + \delta | x \in A\}, alt =' less') \end{aligned}$$

Where $A$ and $B$ are the vectors of the dependent variables (perceived usefulness of DFD and LLM). The $\delta$ represents the range for which we consider the means of both groups to be equivalent. The value of $\delta$ will be determined before the analysis. While estimating the value of delta might seem arbitrary, similar approaches have been used in Food and Drug surveys [26].

## 8 THREATS TO VALIDITY

This section discusses the planned mitigations to potential internal and external threats to validity.

*Internal validity.* We removed the security threat used in the walk-through video from the experiment task to avoid the risk of influencing participant responses.

To ensure that the complexity of the two scenarios is comparable, the DFDs have the same graph topology. Namely, both DFDs contain 3 data store nodes, 1 external entity node, 6 process nodes, and 16 data flows connecting the same type of nodes.

We also consider the threat of introducing experimenter bias in the ground truth. To mitigate this threat, we built the material carefully involving four researchers. The DFD and list of security threats were built by two experimenters, one with more than 8 years of experience in threat analysis, and one with more than 4 years of experience with Kubernetes and cloud security. The ground truth was verified with two other group members (junior, and senior with 10+ years experience in controlled experimentation).

From the control measures, 29/41 participants reported either strongly agreeing or agreeing (points 5 and 4 on the Likert scale) to have a good understanding of what the task required them to do, and 30/41 either strongly agreed or agreed that the time allowed for the experiment was sufficient. To this end, we conclude that the training material and the time allowed to finish the task were sufficient for our target population.

*External validity.* We are aware of the challenges in recruiting participants from crowd-sourcing platforms [13, 30, 31], such as, participants self-reporting on their levels of expertise or background knowledge without having to provide evidence. To mitigate this challenge, we will pre-screen the participants as recommended by Alami and colleagues [3]. The pre-screening layer will include a set of questions on their self-reported knowledge and skills such as questions about the semantics of pseudocode for developers, questions about cybersecurity risk management practices, and questions about software architecture patterns. The pre-screening questions will be used to filter untrustworthy respondents.

We consider the threat of generalizability of our findings to real-world scenarios. To partially mitigate this risk, we use application of K8 pod deployment and Github remote repository update, two realistic and common tasks in software development, and allow participants to use LLMs in a controlled but not limiting way.

While students participating in empirical studies have been reported to have a good understanding of industry-level requirements [38], we decided to invite students in the first pilot study, but plan to conduct the study with practitioners.

## 9 CONCLUSION AND FUTURE WORK

This paper presents the research and execution plan to conduct a study to measure the actual and perceived usefulness of graphical and Large Language Models in validating security threats. To this end, we intend to use a balanced orthogonal design with two interventions, DFD and LLM. As a first step, we ran a pilot with 41 students and outlined the plans to carry out a think-aloud study before the final data collection with industry practitioners.

## ACKNOWLEDGEMENTS

This work was funded by the *Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO)* under the HEWSTI Project under grant no. 14261 and partly supported by the European Union under grant no. 101120393 (Sec4AI4Sec).

## CRediT statements

*Conceptualization:* WM, KT; *Methodology:* WM, KT; *Software:* NA ; *Validation:* WM, KT; *Formal analysis:* WM; *Investigation:* WM, KT; *Resources:*NA; *Data Curation:* WM; *Writing - Original Draft:* WM, KT; *Writing - Review & Editing:* WM, KT; *Visualization:* WM; *Supervision:* KT; *Project administration:* KT; *Funding acquisition:* KT;

## REFERENCES

[1] Silvia Abrahao, Carmine Gravino, Emilio Insfran, Giuseppe Scanniello, and Genoveffa Tortora. 2012. Assessing the effectiveness of sequence diagrams in the comprehension of functional requirements: Results from a family of five experiments. *IEEE transactions on software engineering* 39, 3 (2012), 327–342.

[2] Matthiew Adams. 2024. *AI-driven Threat Modeling with STRIDE GPT.* Technical Report. Open Security Summit. https://youtu.be/_eOcezCeM1M

[3] Adam Alami, Mansooreh Zahedi, and Neil Ernst. 2024. Are You a Real Software Engineer? Best Practices in Online Recruitment for Software Engineering Studies. *arXiv preprint arXiv:2402.01925* (2024).

[4] Karin Bernsmed and Martin Gilje Jaatun. 2019. Threat modelling and agile software development: Identified practice in four Norwegian organisations. In *2019 International Conference on Cyber Security and Protection of Digital Services (Cyber Security)*. IEEE, 1–8.

[5] Vicki Bier. 2020. The Role of Decision Analysis in Risk Analysis: A Retrospective. *Risk Analysis* 40, S1 (2020), 2207–2217. https://doi.org/10.1111/risa.13583 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/risa.13583

[6] Mikael Brejcha. 2024. *STRIDE Threat Modeling Mentor.* Technical Report. ChatGPT. https://chatgpt.com/g/g-gRWzMmly3-stride-threat-modeling-mentor

[7] Yufan Chen, Arjun Arunasalam, and Z Berkay Celik. 2023. Can large language models provide security & privacy advice? measuring the ability of llms to refute misconceptions. In *Proceedings of the 39th Annual Computer Security Applications Conference*. 366–378.

[8] Anton Cheshkov, Pavel Zadorozhny, and Rodion Levichev. 2023. Evaluation of chatgpt model for vulnerability detection. *arXiv preprint arXiv:2304.07232* (2023).

[9] Daniela Soares Cruzes, Martin Gilje Jaatun, Karin Bernsmed, and Inger Anne Tøndel. 2018. Challenges and experiences with applying microsoft threat modeling in agile development projects. In *2018 25th Australasian Software Engineering Conference (ASWEC)*. IEEE, 111–120.

[10] Jose Luis de la Vara, Beatriz Marín, Clara Ayora, and Giovanni Giachetti. 2020. An empirical evaluation of the use of models to improve the understanding of safety compliance needs. *Information and Software Technology* 126 (2020), 106351.

[11] Mina Deng, Kim Wuyts, Riccardo Scandariato, Bart Preneel, and Wouter Joosen. 2011. A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements. *Requirements Engineering* 16, 1 (2011), 3–32.

[12] Mamadou H Diallo, Jose Romero-Mariona, Susan Elliott Sim, Thomas A Alspaugh, and Debra J Richardson. 2006. A comparative evaluation of three approaches to specifying security requirements. In *12th Working Conference on Requirements Engineering: Foundation for Software Quality, Luxembourg*. 1–10.

[13] Felipe Ebert, Alexander Serebrenik, Christoph Treude, Nicole Novielli, and Fernando Castor. 2022. On recruiting experienced github contributors for interviews and surveys on prolific. In *International workshop on recruiting participants for empirical software engineering*.

[14] Guglielmo Faggioli, Laura Dietz, Charles LA Clarke, Gianluca Demartini, Matthias Hagen, Claudia Hauff, Noriko Kando, Evangelos Kanoulas, Martin Potthast, Benno Stein, et al. 2024. Who Determines What Is Relevant? Humans or AI? Why Not Both? *Commun. ACM* 67, 4 (2024), 31–34.

[15] Rafa Galvez and Seda Gurses. 2018. The odyssey: Modeling privacy threats in a brave new world. In *2018 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW)*. IEEE, 87–94.

[16] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *arXiv preprint arXiv:2311.05232* (2023).

[17] Danish Iqbal, Assad Abbas, Mazhar Ali, Muhammad Usman Shahid Khan, and Raheel Nawaz. 2020. Requirement validation for embedded systems in automotive industry through modeling. *IEEE Access* 8 (2020), 8697–8719.

[18] Chadni Islam, M Ali Babar, Roland Croft, and Helge Janicke. 2022. SmartValidator: A framework for automatic identification and classification of cyber threat data. *Journal of Network and Computer Applications* 202 (2022), 103370.

[19] ISO/SAE 21434:2021 2021. *Road vehicles – Cybersecurity engineering.* Standard. International Organization for Standardization, Geneva, CH.

[20] Raghu N Kacker, Eric S Lagergren, and James J Filliben. 1991. Taguchi's orthogonal arrays are classical designs of experiments. *Journal of research of the National Institute of Standards and Technology* 96, 5 (1991), 577.

[21] Peter Karpati, Yonathan Redda, Andreas L Opdahl, and Guttorm Sindre. 2014. Comparing attack trees and misuse cases in an industrial setting. *Information and Software Technology* 56, 3 (2014), 294–308.

[22] Katsiaryna Labunets, Fabio Massacci, and Federica Paci. 2017. On the equivalence between graphical and tabular representations for security risk assessment. In *Requirements Engineering: Foundation for Software Quality: 23rd International Working Conference, REFSQ 2017, Essen, Germany, February 27–March 2, 2017, Proceedings 23*. Springer, 191–208.

[23] Wenxing Liu, Yunduo Wang, Qixiang Zhou, and Tong Li. 2021. Graphical modeling vs. textual modeling: an experimental comparison based on istar models. In *2021 IEEE 45th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 844–853.

[24] Georg Macher, Eric Armengaud, Eugen Brenner, and Christian Kreiner. 2016. A review of threat analysis and risk assessment methods in the automotive context. In *Computer Safety, Reliability, and Security: 35th International Conference, SAFECOMP 2016, Trondheim, Norway, September 21-23, 2016, Proceedings 35*. Springer, 130–141.

[25] Winnie Mbaka and Katja Tuma. 2023. On the measures of success in replication of controlled experiments with STRIDE. *International Journal of Software Engineering and Knowledge Engineering* (2023).

[26] Michael Meyners. 2012. Equivalence tests–A review. *Food quality and preference* 26, 2 (2012), 231–245.

[27] CISA,NSA,FBI,ACSC,CCCS,CERT NZ,NCSC-NZ,NCSC-UK,BSI,NCSC-NL,NCSC-NO,NUKIB,INCD,KISA,NISC-JP,JPCERT/CC,CSA,CSIRTAMERICAS. 2023. *Shifting the balance: Principles and approaches for secure by design software.* Technical Report. Certified Information Systems Auditor (CISA). 36 pages. https://www.cisa.gov/sites/default/files/202310/SecureByDesign_1025_508c.pdf

[28] Marwan Omar. 2023. Detecting software vulnerabilities using Language Models. *arXiv preprint arXiv:2302.11773* (2023).

[29] Andreas L Opdahl and Guttorm Sindre. 2009. Experimental comparison of attack trees and misuse cases for security threat identification. *Information and Software Technology* 51, 5 (2009), 916–932.

[30] Irum Rauf, Tamara Lopez, Helen Sharp, and Marian Petre. 2022. Challenges of recruiting developers in multidisciplinary studies. (2022).

[31] Brittany Reid, Markus Wagner, Marcelo d'Amorim, and Christoph Treude. 2022. Software engineering user study recruitment on prolific: An experience report. *arXiv preprint arXiv:2201.05348* (2022).

[32] Stephen J Ruberg. 1989. Contrasts for identifying the minimum effective dose. *J. Amer. Statist. Assoc.* 84, 407 (1989), 816–822.

[33] Riccardo Scandariato, Kim Wuyts, and Wouter Joosen. 2015. A descriptive study of Microsoft's threat modeling technique. *Requirements Engineering* 20, 2 (2015), 163–180.

[34] Adam Shostack. 2014. *Threat modeling: Designing for security.* John Wiley & Sons.

[35] Laurens Sion, Koen Yskout, Dimitri Van Landuyt, and Wouter Joosen. 2018. Solution-aware data flow diagrams for security threat modeling. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*. 1425–1432.

[36] Laurens Sion, Koen Yskout, Dimitri Van Landuyt, Alexander van Den Berghe, and Wouter Joosen. 2020. Security threat modeling: are data flow diagrams enough?. In *Proceedings of the IEEE/ACM 42nd international conference on software engineering workshops*. 254–257.

[37] Yuqiang Sun, Daoyuan Wu, Yue Xue, Han Liu, Haijun Wang, Zhengzi Xu, Xiaofei Xie, and Yang Liu. 2023. When gpt meets program analysis: Towards intelligent detection of smart contract logic vulnerabilities in gptscan. *arXiv preprint arXiv:2308.03314* (2023).

[38] Mikael Svahnberg, Aybüke Aurum, and Claes Wohlin. 2008. Using students as subjects-an empirical evaluation. In *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*. 288–290.

[39] Zoltán Szabó and Vilmos Bilicki. 2023. A new approach to web application security: Utilizing gpt language models for source code inspection. *Future Internet* 15, 10 (2023), 326.

[40] Chandra Thapa, Seung Ick Jang, Muhammad Ejaz Ahmed, Seyit Camtepe, Josef Pieprzyk, and Surya Nepal. 2022. Transformer-based language models for software vulnerability detection. In *Proceedings of the 38th Annual Computer Security Applications Conference*. 481–496.

[41] Katja Tuma, Gül Calikli, and Riccardo Scandariato. 2018. Threat analysis of software systems: A systematic literature review. *Journal of Systems and Software* 144 (2018), 275–294.

[42] Katja Tuma, Christian Sandberg, Urban Thorsson, Mathias Widman, Thomas Herpel, and Riccardo Scandariato. 2021. Finding security threats that matter: Two industrial case studies. *Journal of Systems and Software* 179 (2021), 111003.