

USENIX

THE ADVANCED COMPUTING
SYSTEMS ASSOCIATION

Clear, Actionable and Confidence-Inspiring Recommendations? Comparative Study of AI-Generated and Human-Written PT Reports

Katarina Galanska, *Masaryk University;* Maria Pibilota Murumaa, *University of Tartu;* Vashek Matyas and Agata Kruzikova, *Masaryk University;* Mike Just, *Heriot-Watt University;* Tomas Cerny, *The University of Arizona*

https://www.usenix.org/conference/soups2026/presentation/galanska

This paper is included in the Proceedings of the Twenty-Second Symposium on Usable Privacy and Security.

August 24–26, 2026 • Hannover, Germany

ISBN 978-1-939133-59-5

Open access to the Proceedings of the Twenty-Second Symposium on Usable Privacy and Security is sponsored by USENIX.

Clear, Actionable and Confidence-Inspiring Recommendations? Comparative Study of AI-Generated and Human-Written PT Reports

Katarina Galanska
Masaryk University

Maria Pibilota Murumaa
University of Tartu *

Vashek Matyas
Masaryk University

Agata Kruzikova
Masaryk University

Mike Just
Heriot-Watt University

Tomas Cerny
The University of Arizona

Abstract

Penetration testing reports typically include remediation recommendations to guide organizations in addressing identified vulnerabilities. The usability of such recommendations influences whether the reports are effectively understood and acted upon. Yet, the usability implications of different recommendation formulations, including AI-generated content, remain underexplored. This paper presents a mixed methods study examining how participants perceive different versions of PT recommendations. In our study, total of 598 IT professionals and cybersecurity students evaluated five recommendation variants for each of two vulnerability descriptions. These variants included versions written by an experienced penetration tester or generated by AI, and differed in the level of detail provided, as well as in the inclusion or exclusion of an explicit responsibility role tag. Study participants evaluated each recommendation using scaled ratings for three aspects: clarity, actionability, and confidence in applying the guidance. They also selected their preferred recommendation in pairwise comparisons and provided reasoning.

Results reveal some significant differences in perceived usability across the recommendation variants. Recommendations with clear steps and an organized structure were rated higher for clarity, actionability, and confidence. Presenting guidance as a sequence of actionable steps helps users quickly understand what to do, reduces ambiguity, and decreases the time needed to interpret and implement the recommendation. These results suggest that both human and AI authors of PT reports should prioritize structured presentation to make security guidance easier to understand and act on.

*A substantial portion of the work presented in this paper was completed while the author was employed at Cybernetica AS.

Copyright is held by the author/owner. Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee.
USENIX Symposium on Usable Privacy and Security (SOUPS) 2026.
August 23–26, 2026, Hannover, Germany.

1 Introduction

There is a growing disclosure of vulnerabilities that can be exploited by malicious actors, some of them related to malware or ransomware attacks [24]. As a result, identifying and addressing vulnerabilities remains a crucial component of building a strong security posture for organizations. Penetration testing (PT) plays an important role in this process by systematically uncovering severe vulnerabilities in deployed systems [7, 13].

However, successful identification of vulnerabilities does not guarantee their effective remediation and security improvements. A critical factor is how well remediation guidance is communicated and understood by the recipients of PT reports [17].

Effective security guidance is critical to ensure that users can make informed decisions and mitigate risks. Recently, Generative Artificial Intelligence (Gen-AI) has been adopted to support and automate various tasks [3], including reporting [4]. In the context of PT, Gen-AI may be used to generate remediation recommendations or assist report recipients in understanding required remediation actions. Though it is not clear what impact it could have on the usability of PT reports, in terms of clarity, confidence in applying the guidance, and actionability of recommended remediation actions.

Previous research highlights usability challenges for the recipients of PT reports, including recommendations that are unclear, insufficiently actionable, or difficult to interpret [11, 26].

The primary objective of this paper is to evaluate and compare the usability of AI-generated and human-written recommendations addressing vulnerabilities identified during PT. The main contribution is to determine the clarity, actionability, and reader's confidence in applying the given security advice and to assess their impact on vulnerability remediation for a recipient of a PT report. In addition to a varied recommen-

dation source (AI or human), our study also considers the level of detail in recommendations, comparing basic versus extended guidance, and explores the use of responsibility or role labels, which indicate who should act to resolve each security issue.

The remainder of this paper is organized as follows. Section 2 provides background on usable security and related work on security recommendations, AI-generated guidance, and human-AI comparison studies. Section 3 describes the study design, including the recommendation corpus, how AI-generated recommendations were produced, the study conditions, participant sample, quantitative measures, and ethical considerations. Section 4 presents the quantitative results, followed by Section 5, which details the results. Section 7 aligns and interprets the quantitative and qualitative findings and discusses the implications and limitations of our study. Section 8 concludes the paper. In the appendix, we include the prompts used for generating AI recommendations and all vulnerabilities employed in the study, along with the full set of recommendation versions presented to participants.

2 Background and Related Work

Penetration testing is a critical process in identifying vulnerabilities within applications. Security experts skilled in exploiting vulnerabilities carry out assessments to simulate real-world attacks and recommend mitigations [13, 30]. The results are typically documented in detailed reports, including the methodology used, findings and suggested countermeasures [35]. These tests have long been a manual process that relied heavily on human judgment to interpret the results and provide actionable security advice [23].

While human-driven PT has been shown to be effective [21], automating the core activities (vulnerability discovery and exploitation) can improve efficiency and consistency [5,12,36]. Previous work has further explored the design and application of automation tools in security testing [10,34]. More recently, efforts have been extended to automating the PT reporting phase, as manual report writing is often time-consuming [17]. Recent research has also explored automating both the testing phase and the report writing phase by integrating traditional security tools with large language models [22, 29]. Previous research has identified limitations in existing penetration-testing automation approaches and highlighted the need for a comprehensive hybrid AI framework to improve the automation of penetration-testing activities [28]. PT report findings are a form of security advice. Despite extensive research on digital security advice, users often do not follow the advice or find it overwhelming, leading to poor implementation of secure behaviors. One recent study aims to develop an approach that makes security advice more actionable, manageable, and easier to follow for end users [27]. Security advice within PT reports often includes recommendations that are shaped by the experience of the penetration tester, knowledge of industry best practices, and understanding of the target environment [23]. Effective remediation depends on recommendations that are both clear and actionable, enabling security practitioners to quickly understand the required steps and implement them correctly [25].

When considering the generation of security advice, it is important to distinguish between general security advice and PT recommendations. General security advice is typically designed for broad audiences and aims to encourage secure behaviors and practices across a variety of contexts [1, 20, 27]. In contrast, PT recommendations are context-specific and evidence-based, derived from validated vulnerabilities and their associated findings. In terms of sources of advice, security practitioners view AI tools such as ChatGPT as beneficial for tasks such as vulnerability detection and penetration testing [14]. However, the generated guidance is often too generic. The responses are perceived more as broad security advice, which is not always suitable for real-world applications. Recent research used SHapley Additive exPlanation (SHAP) in the context of web applications to prove explainability to the AI-generated recommendations. Their analysis showed the ability to provide actionable recommendations and detailed information, including technical fixes and high-level details [16].

Study of practitioner interaction with security tools based on machine learning suggests that the perceived quality of explanations significantly affects confidence and willingness to act on security recommendations, motivating the need to carefully evaluate how mitigation guidance is presented in vulnerability reporting contexts [19].

When a vulnerability is not mitigated and is exploited, incident response (IR) process follows. Recent work has explored the integration of large language models into IR workflows, demonstrating their potential to assist with summarization and reporting, while also highlighting limitations in reasoning quality and factual reliability in security contexts [15].

Previous research examining human vs. AI-generated code has shown that AI-generated code provides more extensive documentation, but it may lack the nuanced context-specific commentary that human developers can provide, reflecting their deeper understanding of the intent and future usage of the code [8]. This highlights the complementary strengths of both approaches, suggesting that AI can improve documentation efficiency, while human developers can add valuable information based on the project's broader context.

3 Methodology

To evaluate and compare the usability of AI-generated and human-written recommendations for addressing vulnerabilities identified during PT, we conducted an exploratory study with 598 participants. Participants were presented with two versions of recommendations per PT finding and asked to evaluate them based on three key aspects: clarity, actionabil-

ity, and confidence in applying the guidance. In addition to rating scales, we collected open-ended responses to capture deeper insights into perceptions and reasoning of their choice.

We then performed a mixed methods analysis combining quantitative and qualitative approaches. The quantitative analysis examined differences in predicted ratings across recommendation versions, while the qualitative analysis involved thematic coding of participant reasoning for their choice to identify recurring usability-related themes. For the qualitative analysis, we collected reasoning only for two aspects: clarity and actionability. This decision was made to avoid overwhelming participants with too many open-ended questions and to ensure rich, detailed feedback on the aspects most directly related to usability.

We established two hypotheses for our study:

H1: AI-generated recommendations are equally clear, actionable, and confidence-inspiring when compared to human-written recommendations in PT reports.

H2: Additional details (i.e., labeling target groups or providing step-by-step instructions) improve the actionability and usability of recommendations.

3.1 Recommendation Corpus

We constructed a corpus of security recommendations centered on two selected vulnerability findings. We deliberately limited the study to two well-understood and widely-encountered security issues in order to reduce cognitive load for participants and to enable richer evaluation of recommendation quality. The first finding, which referred to as I-Type, concerns an information leak vulnerability. The second finding, labeled as C-Type, relates to a cross-site scripting vulnerability. Both vulnerabilities were derived from a real-world public PT report, ensuring that the study material reflects realistic findings observed in professional security assessments [6]. They were chosen since they are rather self-contained and concise making them appropriate for controlled evaluation of recommendation quality.

For each vulnerability, we prepared a set of five different versions of security recommendation, as shown in Table 1. Each set included one human-written recommendation created by a penetration tester taken directly from the public report and four AI-generated variants produced using a fixed prompt structure. The AI variants systematically manipulated two dimensions: the level of detail (basic vs. extended recommendation) and the presence or absence of an explicit responsibility label, indicating the entity responsible for taking action (e.g., backend developer). This design allowed us to isolate the effects of the source of the advice, the level of detail, and the indication of the responsibility.

To protect contextual privacy and avoid unintended identification, vulnerability descriptions were anonymized and intentionally simplified, retaining only the information necessary for participants to understand the security risk. The security recommendation written by the penetration tester was not subsequently modified.

Version	Description
10	Written by a (human) penetration tester.
20	AI-generated with basic level of detail.
30	AI-generated with basic level of detail and role label.
40	AI-generated with extended level of detail.
50	AI-generated with extended level of detail and role label.

Table 1: Recommendation versions for both I and C findings.

3.2 AI Generated Recommendations

The decision to use Gen-AI to produce PT recommendations was motivated by two key considerations. Firstly, generative AI tools are increasingly being adopted in industry, reflecting current practices in cybersecurity and report creation [32]. Using them ensures that our study remains relevant to real-world professional contexts. Secondly, Gen-AI is increasingly applied in defensive cybersecurity contexts, and evaluating its role in remediation communication and reporting becomes necessary [33]. Recent systems demonstrate the growing feasibility of automating various stages of ethical hacking, including analysis and reporting [2]. Specifically, AI powered conversational agents have shown promise in supporting information seeking tasks by maintaining context, generating queries, and summarizing result [9].

The four AI-based recommendations for each vulnerability were generated using GPT-4 (free access tier), accessed in April 2025, with temperature set to 0.7 and a maximum output length of 500 tokens. We selected GPT-based models for advice generation due to their widespread deployment in real-world advice-giving contexts and demonstrated capability in security-related content [18, 31]. The model parameters and prompts were kept constant in all recommendation items to ensure comparability.

To generate security advice, we provided the model with a structured prompt that included the anonymized vulnerability description and instructions for producing the guidance. The prompt explicitly instructed the model to provide a remediation recommendation and defined the level of detail and inclusion of the responsibility label. The complete prompt and all versions of the recommendations are in the Appendix A.

3.3 Quantitative Measures

We assessed participant perceptions of security recommendations using a set of quantitative measures designed to capture selected aspects of usable security. Following prior work on

the evaluation of security advice, participants rated each recommendation version along three selected aspects: clarity, actionability, and confidence in applying the guidance.

- **Clarity** of the recommendation was measured with the item: "How clear is this recommendation?"
- **Actionability** captured whether participants perceived the recommendation as actionable with the following item: “How actionable is this recommendation?”
- **Confidence** in their ability to follow the recommendation was measured with the item: “How confident would you feel implementing this recommendation?”

All items were measured on the four-point Likert scale (1-Not at all, 2-Somewhat, 3-Mostly, 4-Completely). Because these response categories are ordered but not interval-scaled, all outcomes were treated as ordinal variables.

We also collected basic demographic information to support exploratory analysis and control for individual differences, namely their gender, experience, and role.

3.4 Study Sample

Participants were recruited in three phases. In first phase, 318 participants were drawn from two universities, including 300 from Czechia and 18 from Estonia. Phase 2 involved 16 company employees, with 8 from the Czech Republic and 8 from Estonia. Phase 3 comprised 264 participants recruited through DevConf.cz, held in Czech Republic.

The study includes two main groups of participants: cybersecurity students and IT professionals. For students, the survey was administered within university cybersecurity courses and it was not mandatory, although participants could receive minor bonus course points for participation. IT professionals were recruited from organizations that regularly consume PT reports, as well as from attendees of the DevConf.cz, an annual open-source technology conference, where a study booth was available and from which interested individuals could voluntarily participate. To encourage engagement, attendees were offered small incentives, including swag items and ice cream. These gestures helped to create a welcoming and informal environment, fostering participation while maintaining the voluntary nature of the study.

The study sample comprised a total of 598 individuals, of whom 116 identified as female and 482 as male. The first and largest group, consisting of 318 students, were enrolled in cybersecurity-focused academic programs. These students were recruited from university-level cybersecurity courses, ensuring that they possessed foundational theoretical knowledge and emerging practical competencies in the field. The IT professionals group were further split into two groups, as follows. The second group included 248 participants affiliated with software development roles, representing individuals engaged primarily in development-related technical activities. The third group comprised 32 participants specializing in security, reflecting professionals with a dedicated focus on cybersecurity practices and related domains.

Participants were divided into three experience levels: novice (up to 3 years), intermediate (4 to 6 years), and expert (7+ years). The novice level included 415 individuals, primarily students and early-career professionals with limited industry exposure. The intermediate level comprised 84 participants who had several years of relevant experience and were capable of independently handling development or security-related tasks. The expert level consisted of 77 highly experienced professionals with advanced technical knowledge.

3.5 Study Conditions

Data collection was conducted in three phases, during which participant exposure to items varied to balance experimental coverage with participant burden.

In the first two phases, which included data collection within cybersecurity courses and data collection from selected organizations, participants evaluated two versions of recommendations for each vulnerability, which were randomly selected. During the third phase, where participants were conference attendees, the survey consisted only of two versions of the recommendation associated with a single vulnerability, which was randomly assigned. In both cases, it enabled within participant comparison across recommendation variants. This flexible exposure design reduced cognitive fatigue while allowing to collect evaluations across all conditions. The recommendation order was randomized for each participant to mitigate the ordering effects. The design allowed to compare the recommendations both within and between participants.

3.6 Ethics Considerations

The study was conducted using an anonymous, voluntary questionnaire and did not collect personally identifiable or sensitive personal data. On the first page of the online survey, participants were presented with a General Data Protection Regulation (GDPR) notice that outlines the purpose of data collection, data handling procedures, and participant rights. The notice stated that the responses would be used exclusively for research and analysis, that all data would be anonymized, securely stored, and deleted by the end of December 2026, and that participation was entirely voluntary.

All survey responses were optional, allowing participants to provide feedback at a level of detail with which they were comfortable. Participants were informed that they could withdraw from the study at any time while completing the survey questionnaire without providing a reason. In the event of withdrawal, their data would be deleted unless they explicitly agreed otherwise. The notice also specified the categories of

data collected, which were limited to demographic information (gender, role, and experience) and responses to the survey questions. No unnecessary or excessive data were collected. A link to the university's official Data Processing Statement was provided directly within the survey.

Student participation was entirely voluntary and had minimal impact on course standing. Participants received 2 of 102 bonus course points for participation only, independent of performance. Bonus credit was assigned by a researcher using unique participant codes submitted via email, and the course instructor had no access to survey responses. Emails were deleted immediately after credit assignment, ensuring no linkage between identities and responses.

All procedures complied with applicable data protection regulations and institutional research ethics standards.

4 Results

This section presents the findings of our mixed methods analysis. We first report the quantitative results evaluating three selected aspects in the recommendation versions. Then, we present the qualitative findings derived from the reasonings of pairwise choice participants provided.

4.1 Quantitative Results

Prior to analysis, all participant responses were inspected for completeness and consistency. We excluded survey submissions that were incomplete or invalid. Specifically, any participant response that lacked ratings for one or more of the evaluation aspects (clarity, actionability, confidence) was removed. Survey submissions with Likert ratings for given recommendations without reasoning (i.e., why they perceived certain recommendations as clearer or more actionable) were retained in quantitative analysis.

To analyze the data, we used Cumulative Link Mixed Models. This modeling approach is appropriate for ordinal outcomes and allows for the inclusion of fixed and random effects. Mixed effects modeling is the most appropriate approach because each participant provided multiple ratings and evaluated only a subset of the available versions, resulting in an unbalanced and incomplete repeated-measures design.

Separate models were fit for each outcome measure (clarity, actionability, confidence) and for each version (see Table 1). In each model, the version was included as the primary fixed effect. The level of experience and role of the participant were included as additional fixed effects to examine whether these characteristics of the participants were associated with systematic differences in ratings. A random intercept for participant ID was included to account for within-participant dependence across multiple ratings.

Pairwise comparisons between versions were conducted using these marginal means, with p-values adjusted for multiple comparisons. The results are reported with 95% confidence intervals. Analysis was conducted in R using the ordinal and emmeans packages.

By combining absolute ratings with pairwise comparisons, the quantitative measures capture both overall perceptions of advice quality and participants' relative evaluations.

```
formula: Rating ~ Version + Experience +
Role + (1 | participant_id)
```

Within the quantitative analysis, we conducted pairwise comparisons across the three outcome measures: clarity, actionability, and confidence in applying the guidance. For recommendations for the I-type vulnerability finding, no significant differences were observed for actionability or confidence. However, clarity analysis revealed two statistically significant differences, with I50 rated higher than I20 and I30 (all $p < 0.05$), as illustrated in Figure 1.While I50 improved participants' perceived clarity, it did not significantly affect their ability to act on the recommendation or their confidence in applying it.

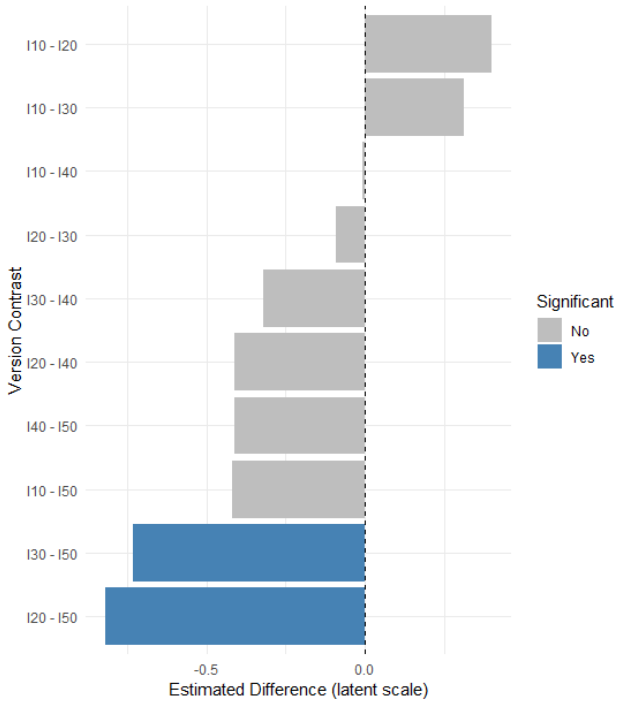


Figure 1: Pairwise comparisons between I versions for clarity outcome for different recommendation versions (see Table 1). Points show estimated marginal mean differences from cumulative link mixed models, with horizontal bars indicating 95% confidence intervals. Estimates are averaged over role and experience and account for participant-level random effects. Comparisons whose confidence intervals do not cross zero indicate statistically significant differences between versions.

For the C-type recommendation type, the pairwise compar-

isons revealed a substantially higher number of statistically significant differences. Specifically, 13 significant pairwise differences were identified across the evaluated usability aspects. These results are summarized in Table 4.1. The higher number of significant findings indicates greater variability in how participants perceived the usability of this recommendation type compared to the first type.

Outcome	Contrast	Estimate	p-value
Clarity	C10 - C40	-0.9512	4.67e-04
	C10 - C50	-1.0545	8.52e-05
	C20 - C40	-1.1101	5.97e-06
	C20 - C50	-1.2134	7.74e-07
	C30 - C40	-0.6576	2.71e-02
	C30 - C50	-0.7609	7.11e-03
Actionability	C10 - C40	-0.8274	3.45e-03
	C10 - C50	-1.1588	1.05e-05
	C20 - C40	-0.8527	1.21e-03
	C20 - C50	-1.1841	2.26e-06
	C30 - C50	-0.7577	7.23e-03
Confidence	C20 - C40	-0.8394	1.97e-03
	C20 - C50	-0.9358	3.97e-04

Table 2: Pairwise comparisons between recommendation versions (see Table 1) for C-type vulnerability across the three evaluation aspects. Estimates are differences in predicted latent ratings from cumulative link mixed models.

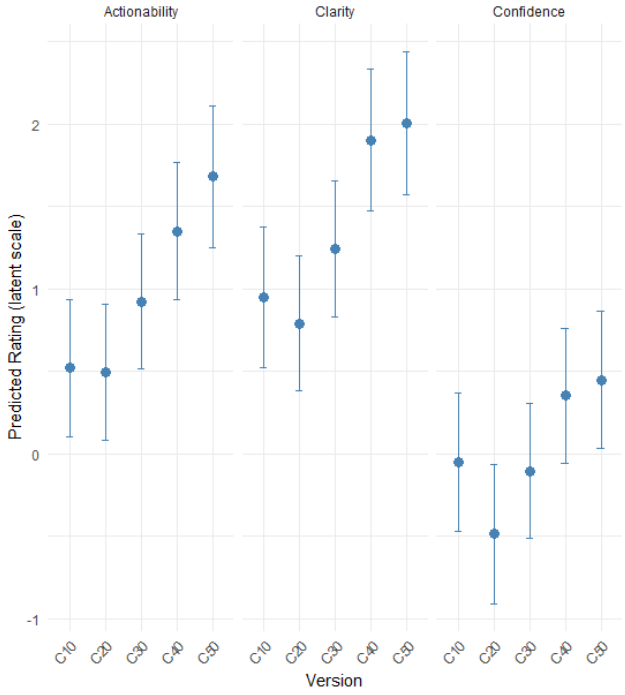


Figure 2: Predicted ratings for different recommendation versions (see Table 1) for the C-type vulnerability with 95% confidence intervals.

Pairwise comparisons of C-type recommendations indicate that versions C40 and C50 were consistently rated higher across all outcome measures, as they were perceived as significantly clearer, more actionable, and more confidence-inspiring than other versions (all $p < 0.05$), with the largest differences observed between C20 and C50.

The predicted ratings for different recommendation versions (see Table 1) addressing the C-type vulnerability were analyzed with 95% confidence intervals to assess differences in perceived clarity and actionability. The results, in Figure 2, indicate that certain versions, such as C40 and C50, received significantly higher predicted ratings in all aspects compared to other versions. The overall ratings for participants' confidence in acting on these recommendations were lower across all versions. Points represent the estimated marginal means from ordinal mixed effects models for each version of the recommendation, and vertical error bars indicate 95% confidence intervals. Separate panels show predicted ratings for each outcome measure: clarity, actionability, and confidence in applying the guidance. Higher points indicate higher predicted ratings on the latent scale.

For the I-type vulnerability, predicted ratings indicate that recommendation versions I50 and I40 remain the highest in terms of clarity and actionability, consistent with the pattern observed in the C-type findings. However, the rating of I10 for clarity increased in comparison to clarity for human written recommendation for C-type finding. Participants also reported higher confidence in acting on I10 compared to the other versions, although confidence remains relatively low overall.

With respect to role and experience for the I-type finding, the three outcome measures (clarity, actionability, and confidence) consistently show a statistically significant effect. Specifically, experience predicted higher ratings of clarity ($\beta = 0.0267,\ z = 2.43,\ p = 0.007$), actionability ($\beta = 0.0267, z = 2.31,\ p = 0.007$), and confidence ($\beta = 0.386,\ z = 2.98,\ p = 0.0006$).

For the C-type finding, participant experience was positively and significantly associated with all three outcome measures. Experience predicted higher ratings of clarity ($\beta = 0.282, z = 2.61, p = 0.009$), actionability ($\beta = 0.266, z = 2.40, p = 0.017$), and confidence ($\beta = 0.392, z = 3.45, p < 0.001$). As with the I-type finding, the strongest association was observed for confidence.

The participant's security role does not have a statistically significant effect on any of the three outcome measures for either of the two vulnerability findings. A student role has a significant negative effect ($\beta = -0.72357,\ z = -3.007,\ p = 0.002641$) on confidence in applying the guidance.

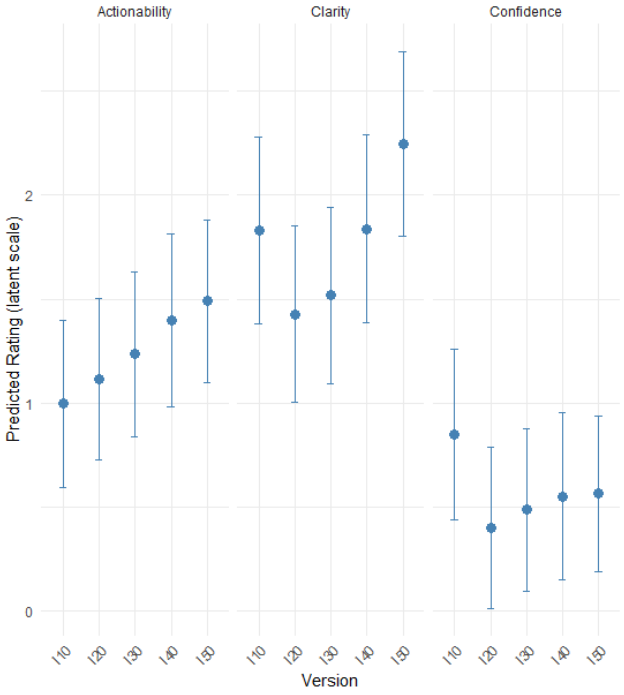


Figure 3: Predicted ratings for different recommendation versions (I-Type vulnerability) with 95% confidence intervals.

4.2 Qualitative Results

We conducted a qualitative analysis to better understand how participants reasoned and compared different versions of the recommendations. We collected pairwise preferences when participants evaluated two versions of the recommendation that address the same vulnerability. After providing a scaled rating for both recommendation items, participants were asked to indicate which of the two they perceived as clearer and which they perceived as more actionable together with the reasoning (we focused on just these two aspects to avoid overwhelming participants – see Section 3). These forced-choice comparisons were used to capture relative preferences between advice versions, complementing the Likert scale measure. These qualitative data are designed to support a deeper analysis of user reasoning and to enrich interpretations of the quantitative results.

For the first vulnerability finding (I-type), we collected 151 qualitative responses explaining clarity ratings and 105 responses explaining actionability ratings. For the second vulnerability finding (C-type), participants provided 140 responses on clarity and 110 responses on actionability, as shown in Table 3. These responses formed the basis for inductive coding, enabling the identification of patterns.

Responses that were entirely blank or non-interpretable were excluded, while partial explanations were retained if they contained meaningful content. The responses were analyzed using an inductive thematic coding approach. Two independent researchers reviewed all participant explanations to identify recurring concepts and patterns related to perceived clarity and actionability. The coders initially examined a subset of responses to generate preliminary codes without imposing predefined categories, allowing themes to emerge naturally from the data.

Finding Type	Clarity (N)	Actionability (N)
I-type finding	151	105
C-type finding	140	110
Total	291	215

Table 3: Number of qualitative responses per finding type and evaluation aspect.

After initial coding, the two researchers compared their code sets and discussed any discrepancies to reach consensus, ensuring reliability and interpretive consistency. When necessary, codes were refined or merged to capture nuanced aspects of participants' reasoning. This iterative process was repeated until saturation was achieved and all responses were consistently coded.

To investigate collected data, we categorized all coded responses into two aspects, actionability and clarity. For each category, we first computed the code frequencies to identify the top ten codes representing the most prevalent actionable suggestions and clarity issues, respectively. To explore patterns beyond individual code prevalence, we also conducted a co-occurrence analysis, calculating the number of times pairs of codes appeared together within the same feedback instance. These co-occurrence data were then visualized as network graphs, where nodes represent individual codes and edges indicate the frequency of co-occurrence.

Figure 4.2 presents the main categories of reasoning, derived from the systematic coding of participants' responses for clarity. The most frequently cited reasoning for clarity included ease of understanding, detailed recommendation, clear steps, and structured presentation. Other considerations, such as providing context or relevant technical details, further highlight that clarity depends not only on what information is communicated, but also on how it is presented. These coded categories provide insight into the key elements that make PT recommendations clear and actionable.

The most frequently cited reasoning for actionability included clear steps, easy-to-implement guidance, and direct, straightforward recommendations, highlighting the importance of concrete instructions that can be applied. The participants also emphasized technical aspects, structured presentation, and the sense that the recommendation effectively mitigates risk.

The heatmap in Figure 4 visualizes the co-occurrence of

R	Clarity Codes	N	Actionability Codes	N
1	Detailed Specific In Depth	55	Clear Steps	60
2	Easy To Understand	55	Easy To Implement	37
3	Clear-Steps	54	Technical Aspect	34
4	Structure	49	Direct Straightforward	33
5	Technical Aspect	33	Feels Secure Risk Mitigated	29
6	Reasoning Context	27	Easy To Understand	19
7	Formatting	26	Structure	19
8	Solution	25	Detailed Specific In Depth	19
9	Role Label	23	Practical	18
10	Direct Straightforward	18	Confidence	10

Table 4: Top reasoning codes for clarity and actionability, derived from systematic coding of participants' qualitative responses where R represents the rank and N represents the count.

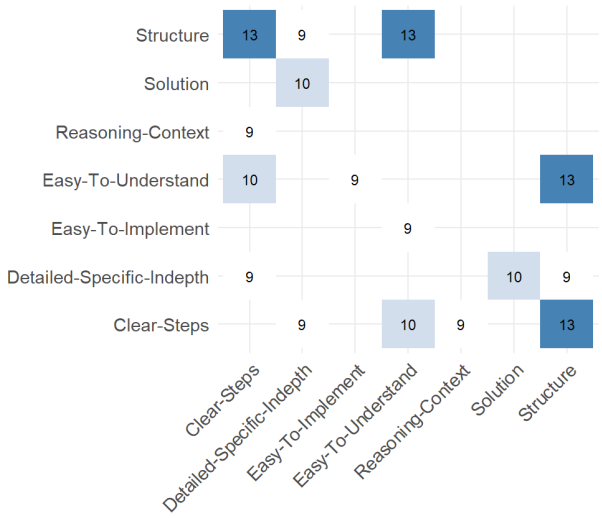


Figure 4: Heatmaps of most frequent co-occurring codes for clarity, where the horizontal and vertical axes represent the two codes in each pair and cell color intensity indicates co-occurrence count.

coded reasoning categories for usability of recommendations, with the color intensity representing the frequency of co-coding. Darker shades indicate higher counts of participants citing the corresponding combination of categories. For example, mentions of structure appear frequently alongside both providing clear steps and presenting a solution to the problem, each with a count of 13. Other notable co-occurrences include the ease of understanding with structure and clear steps, reflecting that clarity is frequently linked to stepwise and organized recommendations. Moderate co-occurrences, such as proving a solution with providing detailed recommendation or ease of understanding with providing a solution, suggest that participants also value specificity and completeness in recommendations.

The clarity network graph shown in Figure 5 clusters of codes that often appeared together, highlighting common combinations of actionable suggestions provided by participants. The network of actionability codes forms a multi-hub star, with five central codes.

Similarly, the heatmap for actionability in Figure 6 illustrates the co-occurrence of coded reasoning categories, with darker shades representing higher frequencies. The strongest co-occurrence was observed between mentions of technical details and a secure feeling of mitigating the risk with the count of 16, indicating that participants frequently linked technically sound recommendations with a perception that the suggested actions effectively mitigate risk. Other notable co-occurrences include the ease of implementation with clear steps and having direct recommendation with clear steps. Additional patterns include mentions of technical details with the recommendation being not only practical, but also with a secure feeling of mitigating the risk, reflecting that participants value recommendations that are not only feasible but also enhance confidence in security outcomes.

The actionability network graph shown in Figure 7, similarly as for clarity, shows clusters of codes that often appeared together, highlighting common combinations of actionable suggestions provided by participants. In particular, the codes related to structure, clear steps, and ease of understanding were strongly linked, suggesting that participants perceive these aspects as mutually reinforcing.

4.3 Themes

Through an iterative process of comparison, consolidation, and discussion, six overarching themes emerged. These themes capture the perspectives of the participants on system navigation, clarity of presentation, urgency and timing, assurance, reasoning, and context, and recipient focus. Each theme is defined in Table 5, along with associated codes and illustrative quotations from participants.

Navigation theme reflects how participants perceive and follow the recommendations in a structured and usable way. Participants frequently mentioned that they appreciated being able to tick the recommendation off as a to-do list, enabling better tracking of tasks. They valued guidance that directly addressed the problem and provided clear steps on what to do, giving them a concrete place to start. Recommendations that offered a primary action to focus on, were practical, included examples, and were actionable were consistently rated as easier to follow and implement. For example, one participant noted, "I feel like it gave clearer more structured and precise

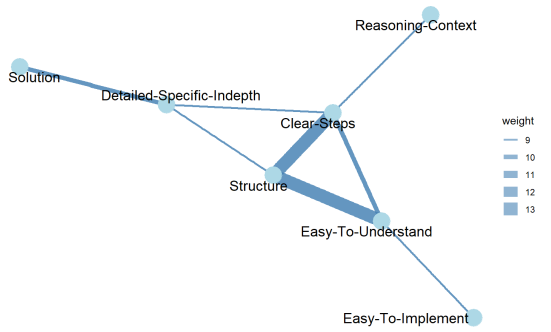


Figure 5: Network graphs of code co-occurrences for clarity, where nodes represent codes, edges connect codes that co-occurred, and edge thickness reflects the frequency of co-occurrence.

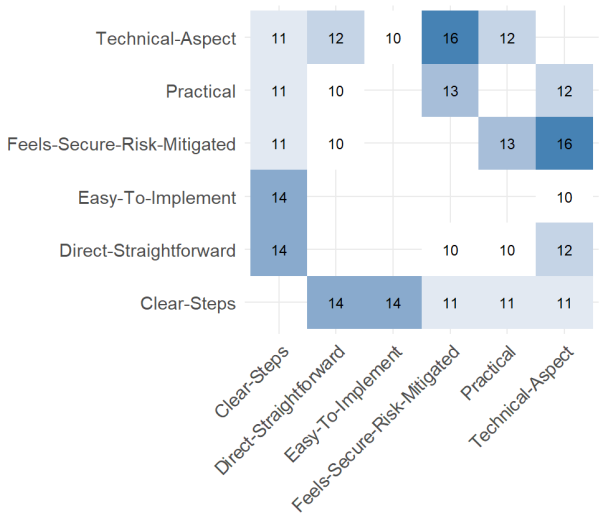


Figure 6: Heatmaps of most frequent co-occurring codes for actionability, where the horizontal and vertical axes represent the two codes in each pair and cell color intensity indicates co-occurrence count.

instructions how to handle the problem", while another described the value of direct instructions, stating, "It provides a clear and direct instruction to remove environment variables from the API response entirely, which is straightforward to implement and immediately reduces the risk of sensitive data exposure."

Presentation and Clarity theme captures how the structure and formatting of recommendations affect the understanding and evaluation of participants. Participants noted that having a structured recommendation helped improve clarity, a pattern that is also reflected in the heatmap of code co-occurrences (see Figure 4), showing strong connections between structured presentation and other clarity-related codes. Recommendations were positively evaluated when they were comprehensive, providing sufficient context and detail, and participants frequently mentioned that terminology and language selection influenced their preference. Formatting elements such as bullet points and highlighting were repeatedly cited as aiding comprehension. Most of the participants positively rated short recommendations for clarity and ease of understanding, although some also appreciated longer, more verbose guidance for providing additional context.

One participant stated, "I50: bulletpoints/steps, bold text, colored text, easy to understand instructions, isn't stripped of important stuff to implement like I20, the colors, steps and bold text makes it look short (which it actually is too), so i wasn't bothered by reading the whole thing", highlighting the value of formatting. Participants also reflected on the language selection, as one noted, "because it was structured and phrased in a clear way."

Urgency and Timing theme reflects how perceptions of participants of time and urgency influence their engagement with the recommendations. Participants indicated that they prioritized certain recommendations due to the time they would have to spend in case they would actually be mitigating the vulnerability. Additionally, recommendations that explicitly conveyed urgency, such as warnings like "never do this", effectively prompted participants to act or avoid risky behaviors. For example, one participant emphasized the need to respond promptly, stating: "the I20 does some tricks to human emotions (fear) when you read it whereas the second one (I10) does not. It would be very right to say that I would react to both, but this time I'm going to be honest and say that I would most likely act on I20." At the same time, participants also considered the time required to effectively address issues. Another participant observed: "I20 if i am in a hurry or I50 if i have more time and resources".

Security Assurance theme captures the desire from participants for confidence that the system or guidance reliably mitigates risks. Participants emphasized the importance of long-term security measures and layered approaches and explained that they chose a particular recommendation because it made them feel secure. For example, one participant stated "it feels more secure as even the pseudonymization can leak some information". Others described selecting actions that directly mitigated potential risks, with one noting: "it's an easy first step that immediately reduces the risk, making it a practical and high-priority action". Participants also highlighted the role of testing and validation, observing that evidence of effective safeguards further reinforced their trust.

Reasoning and Context theme reflects the participants' desire to understand the rationale behind recommendations and the context of potential problems. Participants emphasized that knowing why a problem occurred and the reasoning for suggested mitigation helped them make informed deci-

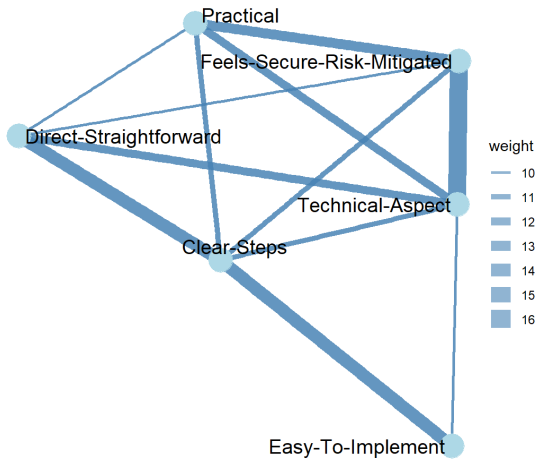


Figure 7: Network graphs of code co-occurrences for actionability, where nodes represent codes, edges connect codes that co-occurred, and edge thickness reflects the frequency of co-occurrence.

sions. For example, one participant explained "there was a better explanation of why should I do such a thing", indicating that understanding the reasoning and context enhanced their confidence in selecting and implementing recommendations.

Recipient Focus theme captures the participant' need for guidance that is tailored to their role, skills, and familiarity with the system. The participants highlighted the importance of familiarity, explaining that the content aligned with their experience or previous knowledge. One participant noted: "Each point had enough explanation to understand what action was needed and why. That made it easier to follow, especially for someone who might not be deeply familiar with every security control mentioned."

Figure 8 references the most used codes related to clarity and Figure 9 related to actionability. In both cases, the Navigation theme, Recipient Focus, and Reasoning and Context emerge as the top three for each, though in different order.

As shown in Figure 10, theme counts in all versions reveal distinct patterns in participant ratings. Navigation appears to be rated higher in more detailed AI-generated versions, showing consistently strong counts across most themes, suggesting improved clarity in these versions. For Recipient Focus, I50 stands out as the best-rated version, indicating that participants in this version perceived the recommendations to be particularly relevant and understandable. From the perspective of Reasoning & Context, the extended versions with labels I20 and C50 exhibit the highest counts, demonstrating that these iterations provided more detailed explanations and contextual information to support the choices of the participants. In particular, for I-type findings, the recommendations written by the penetration tester exhibit better clarity from a Recipient Focus perspective compared to AI-generated recommendations of similar level of detail.

Theme	Codes
Navigation	To-Do List, Direct & Straightforward, Clear Steps, Place to Start, Primary Action, Practical, Actionability
Presentation and Clarity	Structured, Comprehensive, Terminology, Language Selection, Formatting, Length Short, Length Long
Urgency and Timing	Time Consideration, Serious, Urgent
Security Assurance	Long-Term Security Posture, Layered Security Approach, Testing, Feels Secure Risk Mitigated, Relevant, Effective
Reasoning and Context	Problem Origin, Reasoning / Context, Mitigation Choice Offer, Detailed / Specific / In-depth, Technical Aspect
Recipient Focus	Not Overwhelming, Easy to Implement, Easy to Understand, Confidence, Role Label, Target Group, Learnability, Familiarity

Table 5: Themes and associated codes.

Figure 11 illustrates theme counts across versions for the actionability aspect. Navigation was mentioned as a positive aspect of AI-generated recommendations. AI-generated recommendations that are not extended in the level of detail receive similar ratings to the human-written recommendation, with the notable exception of I20, the basic AI-generated version without the role tag, which received the highest Navigation scores across all versions. Recipient Focus shows little difference between written recommendations by penetration testers and extended AI-generated ones, with two exceptions, I40 and C30. AI-generated recommendation C30 received the highest scores for Security Assurance. Compared to the Clarity heatmap, where Urgency & Timing played a minor role, where urgency clearly influenced participants' choices, particularly for penetration tester written I-type recommendations and the basic AI-generated versions, emphasizing the role of time-sensitive guidance in supporting actionable decision-making.

5 Discussion

Hypothesis **H1** received partial support. Relative to the human-written recommendation, AI-generated recommendations were associated with higher clarity ratings across both vulnerability types and higher actionability ratings for C-type vulnerability. However these advantages were not observed consistently across all outcomes. In particular, confidence

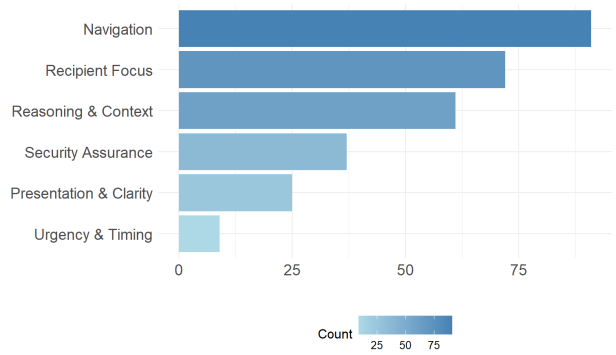


Figure 8: The frequency with which participants referenced codes related to clarity.

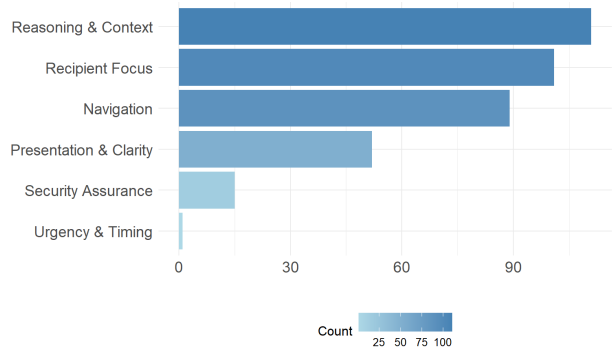


Figure 9: The frequency with which participants referenced codes related to actionability.

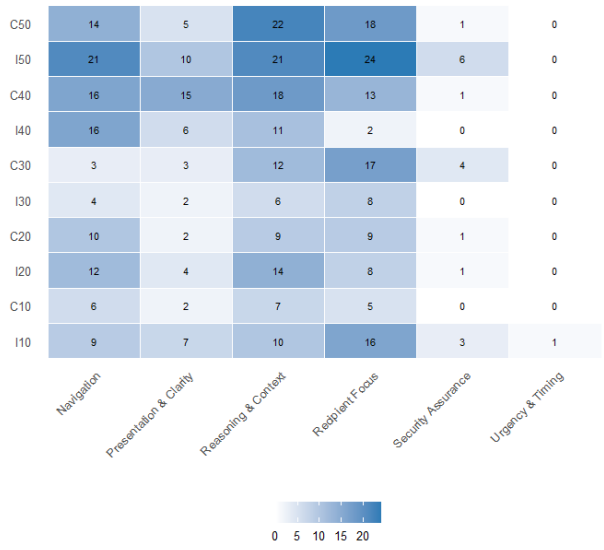


Figure 10: Heatmap showing the frequency of coded themes across al recommendation versions for clarity. Each cell represents the sum of all codes mapped to a specific theme for the corresponding recommendation version. Higher counts are represented by darker shades of blue, and numeric values are shown within each cell.

ratings showed limited differences for C-type vulnerability and were highest for the human-written recommendation in the I-type vulnerability. This pattern suggests that while AI-generated recommendations may improve aspects such as clarity and perceived actionability, they do not uniformly enhance their confidence.

For **H2** results show a consistent effect on perceived usability. Across both detail levels, recommendation variants that included an explicit role label were generally rated higher than equivalent versions without a role label. This pattern was particularly evident for I-type vulnerabilities, where responsibility labeling was associated with higher clarity and actionability ratings. These findings suggest that explicitly identifying the responsible role could support users' interpretation of mitigation guidance by clarifying ownership of the recommended action.

High-detail recommendation variants consistently received higher quantitative ratings and were also most frequently described in the qualitative feedback as clearer and more actionable. This alignment across quantitative and qualitative evidence strengthens the suggestion that more structured and detailed recommendations enhance perceived usability.

Qualitative results suggest that from a clarity perspective, users prefer recommendations that are sufficiently detailed to explain what needs to be done, while remaining structured and understandable enough to avoid cognitive overload. For actionability, users prefer recommendations that provide concrete and direct implementation guidance while also clearly communicating how remediation could improve security.

The observed co-occurrences for clarity suggest that participants frequently mentioned structure, clear steps, and easy-to-understand language together when explaining why recommendations were perceived as clear. This indicates that users associate clarity not only with the amount of information provided, but also with how effectively that information is organized and communicated.

For actionability, participants frequently mentioned clear implementation steps together with straightforward phrasing and ease of implementation, suggesting that users perceive recommendations as more actionable when they can be directly translated into practice with minimal interpretation effort. Similarly, the frequent co-occurrence of technical aspects with perceptions of improved security implies that technically grounded recommendations may strengthen user confidence in the effectiveness of the proposed remediation.

Our findings suggest that usable security depends not only on the technical information of a recommendation, but also on how clearly and effectively it is communicated. Participants

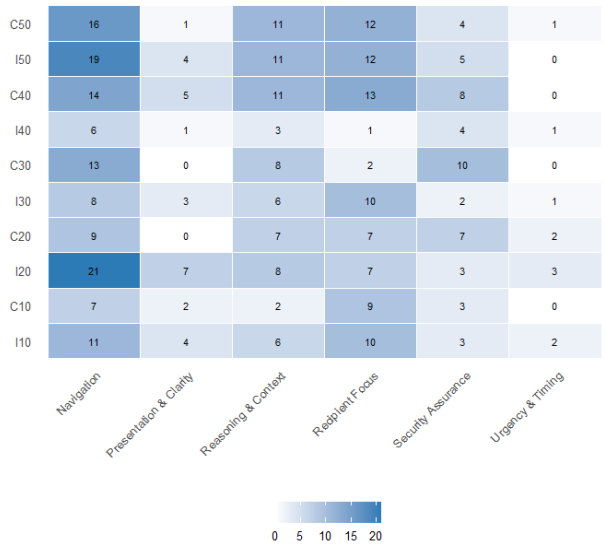


Figure 11: Heatmap showing the frequency of coded themes across all recommendation versions for actionability. Each cell represents the sum of all codes mapped to a specific theme for the corresponding recommendation version. Higher counts are represented by darker shades of blue, and numeric values are shown within each cell.

prefer recommendations that are structured, context-aware, and targeted to the right recipient, which increases both perceived clarity and actionability. For penetration testers, designers of security tools, and AI-generated guidance, this means providing detailed, well-structured, recipient-focused recommendations.

While AI-generated recommendations were generally rated higher overall, human-written recommendations still performed well in key aspects such as focus on the recipient, providing context and reasoning, and clear navigation towards vulnerability mitigation. This suggests that while AI excels at producing detailed, structured and highly rated guidance when prompted to do so, humans continue to provide value in tailoring advice to the audience, embedding situational context.

5.1 Limitations

In this study, there are several limitations that should be considered. First, participants assessed only a small number of vulnerabilities at a time, which does not reflect typical real-world PT workflows where analysts review long reports and must prioritize vulnerabilities. However, the low-volume setting provides valuable insight into how practitioners evaluate individual recommendations, serving as a baseline for future research in other realistic reporting scenarios. Future work is needed to understand how these effects translate to larger reports and prioritization tasks. As such, this study represents a necessary first step in assessing the impact of AI-generated recommendations under conditions where IT professionals can focus on individual findings.

Second, the recommendations were generated using GPT-4, an AI model that was not specifically trained for PT or security advisory tasks. As a result, the output may vary in quality, relevance, or accuracy, and differences may arise with other AI models or future versions.

Third, for each PT finding, we included only one human-written recommendation alongside four AI-generated versions. This imbalance could potentially influence participants' choices or perceptions. However, this design was intentional to allow us to investigate additional dimensions, such as level of detail (basic versus extended) and inclusion of role labels indicating responsibility, which are likely to affect how participants evaluate recommendations. Although this may limit direct comparisons between human and AI recommendations, it provides valuable insight into how structural and contextual features influence usability and decision-making.

Fourth, limitations are common for user studies. The study format may have introduced self-selection bias, where only participants interested in security chose to respond, potentially affecting the representativeness of the results. Responses are also self-reported, which can introduce a bias where participants may overestimate their understanding, compliance, or security practices.

A further limitation of this study is the use of single-item measures for clarity, actionability, and confidence. While single-item measures may have not fully captured the multidimensional nature of these aspects compared to validated multi-item scales, they reduced participant burden and survey length.

6 Conclusion

This work examines the intersection of AI-generated guidance and human judgment, identifying design considerations for usable and trustworthy security recommendations that support effective decision-making in practice.

To explore the differences in usability of PT reports created by human vs AI, we conducted a mixed methods study with 598 participants who chose PT recommendations through pairwise comparisons and provided reasoning for their preference. This combined approach allowed us to align and interpret the findings, showing why certain recommendations were preferred. We focused on three key aspects: clarity, actionability, and confidence in applying the guidance. Overall, the results highlight that while AI-generated recommendations are often highly rated for structure and detail, human input remains important for context, reasoning, and tailoring guidance to recipients, suggesting a hybrid approach is most effective for usable security.

Acknowledgments

We would like to thank the IT professionals from TNS and Cybernetica, as well as the students from the University of Tartu and Masaryk University, and all other participants who contributed to this study.

This work was partly supported by the European Union under Grant Agreement No. 101087529 (CHESS – Cybersecurity Excellence Hub in Estonia and South Moravia).

AI-based language tools were used to assist with language editing and improve the readability of the manuscript. All content was reviewed and validated by the authors.

References

[1] Yasemin Acar, Christian Stransky, Dominik Wermke, Charles Weir, Michelle L. Mazurek, and Sascha Fahl. Developers need support, too: A survey of security advice for software developers. In *2017 IEEE Cybersecurity Development (SecDev)*, pages 22–26, 2017.

[2] Haitham S. Al-Sinani and Chris J. Mitchell. Introducing PenTest++: an AI-augmented, Automated, Ethical Hacking System. In *2025 IEEE 8th Congress on Information Science and Technology (CiSt)*, pages 565–570, 2025.

[3] Ban AlOmar and Zouheir Trabelsi. Integrating generative ai in cybersecurity curricula. In *2025 IEEE Global Engineering Education Conference (EDUCON)*, pages 1–9, 2025.

[4] Apostolos Charalambous, Andriani Piki, Joakim Kävrestad, and Eliana Stavrou. Balancing generative ai and critical thinking to develop written communication skills in cybersecurity. In *2025 IEEE Global Engineering Education Conference (EDUCON)*, pages 1–10, 2025.

[5] Ge Chu and Alexei Lisitsa. Poster: Agent-based (BDI) modeling for automation of penetration testing. In *2018 16th Annual Conference on Privacy, Security and Trust (PST)*, pages 1–2, 2018.

[6] Cure53. Penetration test and source code audit report: authentik idp web, api, and sso. https://cure53.de/pentest-report_authentik.pdf, 2023. Accessed: 2025-03-14.

[7] Chinekezi Chinyere Echefunna, Jude Osamor, Celestine Iwendi, Pius Owoh, Moses Ashawa, and Anand Philip. Evaluation of information security in web application through penetration testing techniques using owasp risk methodology. In *2024 International Conference on Advances in Computing Research on Science Engineering and Technology (ACROSET)*, pages 1–21, 2024.

[8] Tasneem Muhammed Eltabakh, Nada Nabil Soudi, and Doaa Shawky. Quality of ai-generated vs. human-generated code. In *2024 34th International Conference on Computer Theory and Applications (ICCTA)*, pages 200–205, 2024.

[9] Matthew Frazier, Shaayal Kumar, Kostadin Damevski, and Lori Pollock. Investigating user perceptions of conversational agents for software-related exploratoryweb search. In *2022 IEEE/ACM 44th International Conference on Software Engineering: New Ideas and Emerging Results (ICSE-NIER)*, pages 51–55, 2022.

[10] Sahil G K and Chandrashekhar Pomu Chavan. Automated penetration testing tool. In *2025 World Skills Conference on Universal Data Analytics and Sciences (WorldSUAS)*, pages 1–3, 2025.

[11] Katarina Galanska, Agata Kruzikova, Maria Pibilota Murumaa, Václav Matyáš, and Mike Just. From reports to actions: Bridging the customer usability gap in penetration testing. *IEEE Access*, 13:73975–73986, 2025.

[12] Lit Malem Ginting, Suhardi, Kusprasapta Mutijarsa, and Anthon Roberto Tampubolon. Web app penetration testing method: A review. In *2024 International Conference on Information Technology Systems and Innovation (ICITSI)*, pages 609–613, 2024.

[13] Arvind Goutam and Vijay Tiwari. Vulnerability assessment and penetration testing to enhance the security of web application. In *2019 4th International Conference on Information Systems and Computer Networks (ISCON)*, pages 601–605, 2019.

[14] M. Mehdi Kholoosi, M. Ali Babar, and Roland Croft. A qualitative study on using chatgpt for software security: Perception vs. practicality. In *2024 IEEE 6th International Conference on Trust, Privacy and Security in Intelligent Systems, and Applications (TPS-ISA)*, pages 107–117, 2024.

[15] Diana Kramer, Lambert Rosique, Ajay Narotam, Elie Bursztein, Patrick Gage Kelley, Kurt Thomas, and Allison Woodruff. Integrating large language models into security incident response. In *Twenty-First Symposium on Usable Privacy and Security (SOUPS 2025)*, pages 133–148, Seattle, WA, August 2025. USENIX Association.

[16] Manish Kumar, H P Mohan Kumar, S Haridas, and B R Prakash. Web Application Vulnerability Detection and Response using Generative and Explainable AI. In *2025 IEEE International Conference on Contemporary Computing and Communications (InC4)*, pages 1–5, 2025.

[17] Nityashre Jaya Kumar, Sumendra Yogarayan, and Thinesh Ganesan. Design and implementation of penetration testing report engine. In *2025 6th International Conference on Advanced Information Technologies (ICAIT)*, pages 1–6, 2025.

[18] Jingyue Li, Per Håkon Meland, Jakob Svennevik Notland, André Storhaug, and Jostein Hjortland Tysse. Evaluating the impact of chatgpt on exercises of a software security course. In *2023 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, pages 1–6, 2023.

[19] Jaron Mink, Hadjer Benkraouda, Limin Yang, Arridhana Ciptadi, Ali Ahmadzadeh, Daniel Votipka, and Gang Wang. Everybody's got ml, tell me what else you have: Practitioners' perception of ml-based security tools and explanations. In *2023 IEEE Symposium on Security and Privacy (SP)*, pages 2068–2085, 2023.

[20] Lorenzo Neil, Harshini Sri Ramulu, Yasemin Acar, and Bradley Reaves. Who comes up with this stuff? interviewing authors to understand how they produce security advice. In *Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)*, pages 283–299, Anaheim, CA, August 2023. USENIX Association.

[21] Tanveer Ahmed Osman, Jun Russell Ragasa, Hope Posesione, Andrew David, and Masoud Shakiba. Comparative analysis of penetration testing tools in assessing web application vulnerabilities. In *2025 Interdisciplinary Conference on Electrics and Computer (INTCEC)*, pages 1–6, 2025.

[22] A. R. Poroshina and A. V. Obukhov. The methodology of penetration testing using artificial intelligence technology. In *2025 VI International Conference on Control in Technical Systems (CTS)*, pages 234–236, 2025.

[23] Nikhil Rane and Amna Qureshi. Comparative analysis of automated scanning and manual penetration testing for enhanced cybersecurity. In *2024 12th International Symposium on Digital Forensics and Security (ISDFS)*, pages 1–6, 2024.

[24] Recorded Future. H1 2025 malware and vulnerability trends. https://www.recordedfuture.com/research/h1-2025-malware-and-vulnerability-trends, 2025. Accessed: 2025-08-18.

[25] Elissa M. Redmiles, Noel Warford, Amritha Jayanti, Aravind Koneru, Sean Kross, Miraida Morales, Rock Stevens, and Michelle L. Mazurek. A comprehensive quality evaluation of security and privacy advice on the web. In *29th USENIX Security Symposium (USENIX Security 20)*, pages 89–108. USENIX Association, August 2020.

[26] Steven Lamarr Reynolds, Tobias Mertz, Steven Arzt, and Jörn Kohlhammer. User-centered design of visualizations for software vulnerability reports. In *2021 IEEE Symposium on Visualization for Cyber Security (VizSec)*, pages 68–78, 2021.

[27] Anna Lena Rotthaler, Harshini Sri Ramulu, Lucy Simko, Sascha Fahl, and Yasemin Acar. "it's time. time for digital security.": An end user study on actionable security and privacy advice. In *2025 IEEE Symposium on Security and Privacy (SP)*, pages 2228–2245, 2025.

[28] Verina Saber, Dina ElSayad, Ayman M. Bahaa-Eldin, and Zt Fayed. Automated penetration testing, a systematic review. In *2023 International Mobile, Intelligent, and Ubiquitous Computing Conference (MIUCC)*, pages 373–380, 2023.

[29] Maher Salem and Mohammad Mrian. AI-Driven Penetration Testing: Automating Exploits with LLMs and Metasploit-A VSFTPD Case Study. In *2025 International Conference on New Trends in Computing Sciences (ICTCS)*, pages 89–96, 2025.

[30] Devin Sweigert, Md Minhaz Chowdhury, and Nafiz Rifat. Exploit security vulnerabilities by penetration testing. In *2022 IEEE International Conference on Electro Information Technology (eIT)*, pages 527–532, 2022.

[31] Yuichiro Tateiwa. Development of dialogue feature between participants and chatgpt in network security exercise system. In *2024 7th International Conference on Information and Computer Technologies (ICICT)*, pages 479–484, 2024.

[32] Thomson Reuters. 2025 Generative AI in Professional Services Report. https://www.thomsonreuters.com/content/dam/ewp-m/documents/thomsonreuters/en/pdf/reports/2025-generative-ai-in-professional-services-report-tr5433489-rgb.pdf, 2025. Accessed: 2026-06-04.

[33] Shrikumar Pramod Ugale and Anupa Sinha. Generative AI in Cybersecurity: Threats, Vulnerabilities, and Protective Measures. In *2025 International Conference on Future Technologies (ICFT)*, pages 1–8, 2025.

[34] Benlong Wu, Guoqiang Chen, Kejiang Chen, Xiuwei Shang, Jiapeng Han, Yanru He, Weiming Zhang, and Nenghai Yu. Autopt: How far are we from the fully automated web penetration testing? *IEEE Transactions on Information Forensics and Security*, 20:9657–9672, 2025.

[35] Mohd Nizam Zakaria, Poon Ai Phin, Nurfarahin Mohmad, Saiful Adli Ismail, Mohd Nazri Kama, and Othman Yusop. A review of standardization for penetration testing reports and documents. In *2019 6th*

International Conference on Research and Innovation in Information Systems (ICRIIS), pages 1–5, 2019.

[36] Peng Zhang, Wenhui Wang, Daojuan Zhang, Hongbin Wu, Longxi Han, and Qi Zhao. Automated security penetration testing based on machine learning. In *2024 4th International Conference on Electronic Information Engineering and Computer Communication (EIECC)*, pages 1593–1597, 2024.

Appendix A Prompt for Generating Recommendations

This appendix presents the prompts used to generate the AI-written recommendations evaluated in the study.

```
You are a security expert writing
a recommendation for a penetration
testing report.
Based on the following finding,
write a remediation recommendation.
Keep it basic/detailed.
Do not mention role responsibility/
explicitly mention the responsible
role for mitigating the issue.
The recommendation should help
developers understand what they
need to do to fix the issue.
Finding:
```

Appendix B Vulnerabilities

This appendix provides the full text of the recommendation variants evaluated in the study. Each vulnerability was presented with multiple versions of recommendations that differed in three aspects: 1. authorship (human-written vs. AI-generated), 2. level of detail (basic vs. extended), and 3. inclusion of an explicit responsibility role tag.

B.1 Type-I Finding

B.1.1 Informational Leak (Severity Low)

Finding Description

During the assessment of the backend features of the application, it was discovered that the system endpoint returns content from the environment variables to the client in plaintext. This is considered a bad practice because it increases the risk of data being disclosed to third-parties unnecessarily.

Request:

```
GET /api/v3/admin/system/ HTTP/2
Host: app.test.com
Cookie: app_session=
```

Affected Response:

```
HTTP/2 200 OK [...] \
{"env": {[...] "POSTGRESQL__PASSWORD":"Johrio4oche[redacted]
"SECRET_KEY":"k18 v9& [redacted]",
"SOURCE_CONSUMER_KEY":"JvQqX0YJ iACV [redacted]
"BOOTSTRAP_TOKEN":"0K wqH [redacted]", [...
"AUTHENTIK_BOOTSTRAP_PASSWORD":"n* KV> [redacted]
```

Figure 12: Finding I

Sensitive data, such as passwords and secrets, should not be returned in cleartext in responses sent back to the clients. To protect passwords and secrets, returns should only be done in pseudonymized formats, for example by replacing them with "*".

Figure 13: Recommendation I10

Update the /api/v3/admin/system/ endpoint to exclude environment variables from the response. Sensitive information such as database passwords, secret keys, and tokens should never be exposed via APIs. Review the code handling this endpoint and remove any logic that outputs environment data. If environment details are needed for debugging, restrict this functionality to internal logs only and ensure it is never returned to clients in responses. Re-deploy the application after verifying the output is clean.

Figure 14: Recommendation I20

To address the information disclosure issue, ensure that sensitive environment variables, such as passwords and secret keys, are not exposed in API responses. Follow these steps:

1. **Mask Sensitive Data:** Implement logic to sanitize the output so sensitive fields like AUTHENTIK_POSTGRESQL_PASSWORD, AUTHENTIK_SECRET_KEY, and others are replaced with masked values (e.g., ********).
2. **Access Control:** Restrict access to this endpoint to authorized users only by applying proper authentication and authorization mechanisms. Use roles or permission levels to limit access to sensitive data.
3. **Environment Variable Management:** Ensure that environment variables and secret keys are managed securely, using tools like Vault or secure storage solutions. Avoid hardcoding sensitive data in the codebase.
4. **Response Filtering**: Review and filter the data that is returned from the backend to the client. Ensure that non-essential data, such as internal configuration, is never exposed in the response body
5. **Security Testing**: After implementing these changes, conduct a security assessment to confirm that no sensitive information is being disclosed and that the endpoint behaves securely.

Figure 15: Recommendation I40

The I30 and I50 versions had the addition “Responsible Role: Backend Developers”.

Responsible Role: Backend Developers

Figure 16: Role Label

B.2 Type-C Finding

B.2.1 XSS via CAPTCHA JavaScript URL (Severity Medium)

Finding Description

Any arbitrary JavaScript can be loaded using the Captcha stage. This is also limited to superusers. As its name suggests, the "Captcha Stage" of the stage setting is there to enable configurations of CAPTCHAs. Specifically, a superuser can set a JavaScript URL for a CAPTCHA library. However, it was discovered that this option allows setting arbitrary JavaScript URLs, resulting in Cross-Site Scripting (XSS). The issue can be reproduced by setting a JavaScript URL in the "*JS URL*" field, adding it to a flow and then opening the page.

Request:

```
GET /api/v3/admin/system/ HTTP/2
Host: app.test.com
Authorization: Bearer
Content-Type: application/json
{ "name": "XSS Test Stage",  "type": "captcha",
  "js_url": "data:text/javascript,alert(document.domain)"}
```

Response:

```
HTTP/1.1 201 Created
Content-Type: application/json
 <  / b >  < bid = "docs-internal-guid-cc70a23e-7fff-7e98-7a2f-e
```

Only superusers can perform this action. While they are generally trusted, this feature still lets them impersonate other users. Because the XSS is stored and runs when the page is opened, it could also be used to target other superusers.

Figure 17: Finding C

In order to prevent potential exploitation, it is recommended to limit the setting to allowlisted JavaScript URLs only.

Figure 18: Recommendation variant C10

Ensure that the “*JS URL*” field in the CAPTCHA configuration only accepts trusted, validated URLs. Implement input sanitization to prevent arbitrary JavaScript from being injected. Use a strict Content Security Policy (CSP) to block inline scripts, and limit the scope of superuser access to reduce potential attack vectors.

Figure 19: Recommendation variant C20

To resolve the XSS vulnerability, you need to take the following steps:

1. **Input Validation:** Implement input validation on the "*JS URL*" field to only allow trusted URLs. Reject any input that includes malicious payloads or unexpected protocols (e.g., *data:, javascript:*).
2. **Sanitize Input:** Ensure that the URL input is sanitized to strip out any executable JavaScript code. This can be done by whitelisting allowed URL schemes (e.g., *https://*), and preventing any scripts from executing.
3. **Content Security Policy (CSP):** Configure a CSP header to block inline JavaScript execution. This will prevent the execution of malicious JavaScript code even if it bypasses URL validation.
4. **Limit Superuser Access:** Review the need for superusers to configure CAPTCHA JavaScript URLs. Restrict this capability to a minimal number of trusted users, and implement proper auditing of actions taken by superusers.
5. **Test the Changes**: After implementing these measures, thoroughly test to ensure that no arbitrary JavaScript can be injected via the CAPTCHA configuration

Figure 20: Recommendation variant C40

The C30 and C50 versions had the addition “Responsible Role: Backend Developers”.

Responsible Role: Backend Developers

Figure 21: Role Label