Reading Response 2: Let's Hash and Writing on Usable Security

CSEC 759 - Mehul Sen

Geirhaas et al. discuss the importance of implementing robust authentication systems and proper password policies in their paper "Let's Hash: Helping Developers with Password Security" [1]. They aim to provide developers with secure code that can be conveniently copied and pasted from a resource directly into their implementations. The authors' objective was to create a more secure and user-friendly code, and they investigated whether incorporating a wizard, which asked developers questions and designed the code based on their responses, would improve the usability and security of their resource. To evaluate their tool, they conducted an online study with 179 freelance developers from Freelancer.com who were randomly divided into three groups: the group with their resource "Let's Hash," which provided developers access to secure code snippets directly (LH), a version of their resource that provided a wizard for the technical configurations (LH-W), and a control group that had no such resource available to them. The participants were tasked with three programming exercises relating to password storage, password policies, and two-factor authentication. They then completed a survey assessing their experiences. This study was conducted online through the Developer Observatory tool, and it was assessed using System Usability Scale (SUS) values, number of clicks, and time. Geirhaas et al. discovered that their tool significantly improved the security of the developers' code who used Let's Hash, regardless of their version. Additionally, they observed that the wizard configuration addition to Let's Hash and the direct code snippet resource had identical usability ratings, and the wizard did not have any significant improvements. In contrast, the security ratings of the direct code snippet were higher. The authors also noted that trust and perceived usability were critical factors for the successful establishment of Let's Hash.

Schechter's "Common Pitfalls in Writing about Security and Privacy Human Subjects Experiments, and How to Avoid Them"[2] provides guidelines and common pitfalls that many research papers covering human subject experiments of security and privacy should observe. Based on these guidelines, we reviewed the paper by Geirhaas et al.. The experimental design of their research clearly stated the hypothesis and acquired the appropriate approvals from the IRB while complying with GDPR. They also provided an appendix for additional information. However, they did not clarify the information their participants might have learned through the study or how their behavior might have been influenced in the experiment. For example, the participants in LH and LH-W, who had to use the Let's Hash tool, were likely expecting all their solutions to be available within the tool itself and may not have considered any alternatives. On the other hand, the control group would be more inclined to look through additional resources, such as private blogs and personal cheat sheets, that they might not have wanted to access during the study. Even though this study was done entirely online, it retained its ecological validity because many developers work from home and program online. However, one key difference between this study and actual tasks was that the participants might have put in less effort or had different priorities in a real task since the tasks mentioned were short and did not involve finding the resources but only their implementations. The authors mention this fallback in the limitations section of their paper. When presenting results, the authors used detailed charts and tables with clear labeling, as recommended by Schechter. However, Figures 2, 3, and 4 only showed LH and LH-W, omitting the results from the control group. Additionally, using bar graphs to show SUS values, clicks, and time was not the ideal option to compare the results. Using a line graph with an overlay for each category would make comparisons between their categories much more straightforward.

Additionally, of the metrics used by the authors, using time and clicks to measure usability may not provide the most accurate results since a tool is not considered usable if it is used in the least amount of time or requires the least amount of clicks.

References

[1]: "Let's Hash: Helping Developers with Password Security" Lisa Geierhaas, Anna-Marie Ortlhoff, Matthew Smith, Alena Naiakshina, *USENIX Symposium on Usable Privacy and Security*, Aug. 2022.

[2]: "Common Pitfalls in Writing about Security and Privacy Human Subjects Experiments, and How to Avoid Them" Stuart Schechter, 2013.