ISCTE IUL
REPOSITÓRIO
INSTITUTO UNIVERSITÁRIO DE LISBOA

Repositório ISCTE-IUL

Deposited in *Repositório ISCTE-IUL*:
2025-12-17

Deposited version:
Accepted Version

Peer-review status of attached file:
Peer-reviewed

Citation for published item:
Iosif, A., Lechner, U., Pinto-Albuquerque, M. & Gasiba, T. (2024). Serious game for industrial cybersecurity: Experiential learning through code review. In A. Bollin, I. Bosnic, J. Brings, M. Daun, M. Manjunath (Ed.), Software Engineering Education Conference, Proceedings. Würzburg, Germany: IEEE.

Further information on publisher's website:
10.1109/CSEET62301.2024.10663058

Publisher's copyright statement:
This is the peer reviewed version of the following article: Iosif, A., Lechner, U., Pinto-Albuquerque, M. & Gasiba, T. (2024). Serious game for industrial cybersecurity: Experiential learning through code review. In A. Bollin, I. Bosnic, J. Brings, M. Daun, M. Manjunath (Ed.), Software Engineering Education Conference, Proceedings. Würzburg, Germany: IEEE., which has been published in final form at https://dx.doi.org/10.1109/CSEET62301.2024.10663058. This article may be used for non-commercial purposes in accordance with the Publisher's Terms and Conditions for self-archiving.

Use policy

Creative Commons CC BY 4.0
The full-text may be used and/or reproduced, and given to third parties in any format or medium, without prior permission or charge, for personal research or study, educational, or not-for-profit purposes provided that:

- a full bibliographic reference is made to the original source
- a link is made to the metadata record in the Repository
- the full-text is not changed in any way

The full-text must not be sold in any format or medium without the formal permission of the copyright holders.

Serviços de Informação e Documentação, Instituto Universitário de Lisboa (ISCTE-IUL)
Av. das Forças Armadas, Edifício II, 1649-026 Lisboa Portugal
Phone: +(351) 217 903 024 | e-mail: administrador.repositorio@iscte-iul.pt
https://repositorio.iscte-iul.pt

Serious Game for Industrial Cybersecurity: Experiential Learning through Code Review

Andrei-Cristian Iosif*, Ulrike Lechner†, Maria Pinto-Albuquerque‡, Tiago Espinha Gasiba*
*Siemens AG, Munich, Germany — {andrei-cristian.iosif, tiago.gasiba}@siemens.com
†Universität der Bundeswehr München, Munich, Germany — ulrike.lechner@unibw.de
‡Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR, Lisbon, Portugal — maria.albuquerque@iscte-iul.pt

***Abstract*—Every stage of the industrial software development process is crucial for ensuring high-quality results in a time of increasing digitalization and complexity. Code review is a method to enhance software quality and also promote knowledge exchange among teams. It is generally accepted that the earlier that software bugs and vulnerabilities are caught during product development, the more costs can be saved. As such, code review can play an important role in industrial software development.**

However, industry experience showcases that code review can be resource-intensive, and the direct impact on code quality can be hard to quantify. Related work shows that practitioners performing code reviews do not focus specifically on security, partly due to a gap in awareness of the topic.

Our research focuses on improving the efficiency and effectiveness of code review practices, particularly in identifying and addressing security issues in an industrial context.

The present work showcases results from using a serious game as a means to empower developers, by exhibiting code review best practices and raising awareness of security concerns. We collect results over a series of 11 experiments conducted in an industrial setting together with a total of 175 industrial practitioners, serving as a pilot stage, based on which we discuss and conclude on important aspects of the design of the game.

***Index Terms*—Code Review, Cybersecurity, Compliance, Action Design Research, Developer Empowerment**

I. Introduction

Cybersecurity attacks continue to rise in number and impact, with cybercrime costs expected to exceed $10B per year, according to a study by Forbes in 2023 [1]. To counter this trend, companies secure the software development lifecycle (SDLC) and employ the DevSecOps paradigm, which lays out practices to integrate security into all steps of the SDLC [2].

One of the cornerstones of DevSecOps is shifting security *left* to avoid vulnerabilities in software at earlier stages of development and reduce the costs of remediating the flaw. While DevSecOps focuses on automating security activities, the human factor cannot be discarded. One of the ways to secure development at earlier stages includes manual effort. Securing the development stage through multiple strategies ensures that more bugs are spotted earlier in the lifecycle, as per Rajapakse et al. [3].

One way to improve the level of security of code in the earlier stages of software development is by performing manual code reviews. Code review is the practice of manual source code inspection by developers other than the author [4].

However, according to Braz et al. [5], the lack of security awareness from the developers' side and the lack of resources are the main causes impacting the effectiveness of code review toward detecting vulnerabilities and generally increasing code quality. Although the security awareness challenge is known, limited studies have been conducted on code review practices with the objective of enhancing cybersecurity. We thus motivate our research to explore increasing the quality of code review practices towards cybersecurity.

Problem Statement. This work proposes an innovative approach to code review as part of the SDLC, aimed at raising the level of industrial standards compliance, strengthening secure coding awareness, enhancing the quality of code review, and empowering software developers for effective code reviews.

To this end, we propose a serious game (i.e., a game with an end goal other than pure entertainment [6]) for leveraging code review practices in the industry and enhancing developers' awareness of security best practices.

The design process of the serious game artifact is done in an iterative process, following Action Design Research principles. Drawing on insights from professional experience, this study initiates the development of a serious game prototype aimed at both evaluating and improving secure code review practices.

The paper is organized as follows: Section II introduces related work. Section III follows by describing the design methodology and its reflection within the current implementation. Section IV presents the experimental setup and evaluation strategies. Results are discussed in section V. Finally, Section VI concludes the paper and lays out plans for further work.

II. Background and Related Work

This section presents an array of findings from selected academic works, and gray literature meant to outline the current challenges of the SDLC in terms of ensuring security through efficient code review.

Our problem statement is positioned at the intersection of four core areas: security, industry standards, awareness, and empowerment. Whereas the first two areas are mainly technical (although technically speaking they should also consider human factors), the latter two consider the human aspect of our research. We borrow from Hänsch et al. [7] and Tessem et al. [8] for the definitions of, respectively, awareness, and empowerment.

A. Awareness and Empowerment

Hänsch et al. [7] define IT security awareness across three dimensions: *perception* (knowledge of existing software vul-

nerabilities), *protection* (knowing how to secure software), and *behaviour* (intent to perform security activities).

Empowerment in the organizational context is delineated through structural and psychological dimensions, as per Tessem et al. [8]. It comprises power, information, knowledge, reward, and cognitions of meaningfulness, competence, self-determination, and impact. This multifaceted approach underlines the interaction between an individual's access to resources and their intrinsic motivational states. Empowerment can thus be loosely defined as fostering an environment conducive to enhanced autonomy and effectiveness within the workplace.

We utilize both perspectives, technical and human, to address the real-world problem of cybersecurity awareness and strive for the adoption of our ideas by users and in industrial software engineering contexts. Our artifact aims to provide practitioners with insight into the first two dimensions of awareness, aiming to empower them and positively shape their future behavior.

We chose serious games as a medium for information delivery. Previous work has indicated they are effective in disseminating takeaways to users of such an artifact, as laid out in an experience report of Namin et al [9] and through a literature review conducted by Hendrix et al [10].

To better leverage the potential of serious games as a vector of developer empowerment, we must first understand the difficulties and requirements that industrial code review falls under.

B. Challenges associated with Code Review

Automation provided through DevSecOps can help to lessen human tasks in software testing, as noted by Sánchez-Gordón et al. [2]. Nevertheless, tools like Static Code Analysis (SAST) often produce false positives and thus require human filtration, as per Mao et al. [11].

Ami et al. [12] investigate practitioners' assumptions and behaviors of SAST tools and highlight an industrial tendency of tool over-reliance and lack of awareness on *false negatives*. Rindell et al. [13] explore how enriching the results from SAST tools alone can be insufficient and propose manual code review as one of the ways to tackle security-related tech debt. Tackling false negatives from SAST tools can be tedious, as the tools are unfit to catch, e.g., business logic or architectural security flaws within source code.

Kupsch et al. [14] emphasize that, despite the benefits of automation, manual reviews are still required to ensure good coverage to find and address vulnerabilities.

Bosu et al. [15] suggest that industry professionals find reviews valuable mainly when they enhance code quality. MacLeod et al. [16] explored what makes a code review useful to code contributors, emphasizing challenges like misguided review focus, process ambiguity, and insufficient training.

Recent developments in artificial intelligence (AI) through breakthroughs in Large-Language Models (LLMs) and their application show promise in offering more robust decision-supporting tooling, as highlighted by Fu et al. [17]. Nonetheless, given the nascent stage of the LLM field, it would be premature to *exclusively* commit to a tool-centric approach without also empowering with other means the developers who employ an AI-assisted workflow.

Challenges of code review highlighted by MacLeod et al. [16] pertain wholly to the human aspect of code review (e.g. scope ambiguity, timeliness) which further underscores the industrial significance of our work.

C. Compliance in the Industry

Ramaj et al. [18] highlight that compliance is currently a blind spot within the DevSecOps paradigm. Regarding standardization within the industry, Moyon et al. [19] have highlighted that integrating security into agile software development presents practical challenges. This is especially the case in heavily regulated industries, where scaling agile practices and achieving compliance with security standards leads to enhanced complexity.

Relevant to our work, the IEC 62443-4-1 [20] and IEC 62443-4-2 [21] standards highlight the importance of secure code review in the development lifecycle and the need for specialized knowledge on secure code review among reviewers. ISO/IEC 20246 [22] offers a generic framework for work product reviews applicable across various organizational roles.

Additionally, the ISO/IEC TR 24772-1 [23] provides guidance on avoiding programming vulnerabilities, while standards like MITRE's Common Weakness Enumeration (CWE) Top 25 [24] and OWASP Top 10 [25] offer insights into secure coding practices.

The role played by standards in our work is two-fold. Firstly, they provide a foundation for emphasizing the significance of cybersecurity in software, as standard-compliant products experience enhanced value. Secondly, standards serve as a fundamental resource from which requirements can be extracted and transformed into practical guidelines for reviewers, thereby offering guidance in the pursuit of secure code review within an industrial environment.

III. Methodology

The present work offers a preliminary set of findings collected from crafting and assessing a serious game software artifact focused on empowering software developers on secure code review. Starting from a problem formulation, i.e. *"How to empower developers towards performing secure code review?"*, our goal is to iteratively gain knowledge into this industrial challenge while refining our proposed solution – a software artifact in the form of a serious game.

A. Artifact Design

Our research approach is anchored in the Action Design Research (ADR) method by Sein et al. [26], which focuses on creating and assessing IT artifacts within an organizational context to address specific problems. The authors contend that design science, through the creation of innovative artifacts, aims to enhance human capabilities.

ADR emphasizes iterative collaboration between researchers and practitioners, intertwining problem understanding, action, and evaluation. It operationalizes the principles of Design Science Research (DSR) by embedding the artifact development process within organizational contexts, thus bridging the gap between theory and practice by iteratively designing and evaluating artifacts in situ.

Vom Brocke et al. build upon DSR notions laid out by Hevner et al. and formalize the six dimensions of DSR [27]. Next, we will delineate these dimensions as they apply to the development of our serious game artifact.

Problem Description: Identifying the gap in current code review training methods – Developers are not effectively equipped with the necessary skills and best practices for security-oriented code review.

Input Knowledge: Existing literature on code review processes, design theories, and development practices.

Research Process: Designing and developing the software artifact.

Key Concepts: Code quality standards, review techniques.

Solution Description: Features and functionalities of the software artifact – interactive code review exercises, feedback mechanisms, and tools integration.

Output Knowledge: Evaluating the effectiveness of the serious game in improving code review skills among developers.

Situated in an industrial software setting where cybersecurity is vital, our goal is to enhance code review effectiveness and heighten security awareness. Key to our approach is the design and assessment of a serious game for software developers. With two researchers possessing extensive experience in industrial software development and secure coding, we collaborate with the industry to introduce fresh academic perspectives and rigorously evaluate existing practices. Previous work performed by the authors has already shown that embedding a serious game within workshops performed in the industry was well-received [28]. While gamification has been applied to static code analysis and secure coding in previous works, our approach uniquely integrates these elements into a comprehensive serious game designed to enhance secure review practices in an industrial context.

Based on the positive findings of the first ADR iteration, we consider further refining the artifact to better understand the problem of secure code review and address the practitioners' needs on the topic. We are also looking into the *resource* problem - code review is a demanding process on teams, and therefore we need an approach to optimize the necessary resources. Collaborating with industry practitioners is, therefore, essential for collecting insight into this matter.

Our methodology adopts an iterative stance specific to ADR, with the iteration continuing until a "satisficing" threshold is attained [29]. The iteration process persists until the developed solution adequately addresses the key issues within the problem context, meeting the essential needs and expectations of stakeholders to a degree considered satisfactory and functional – i.e. when participants consistently report that they draw knowledge and practical applicability from participating in the proposed serious game, essentially heightening their empowerment in performing secure code review.

In this first iteration, we aim to evaluate the design in terms of providing developers with the awareness and necessary resources for performing a proper code review for security features. Our artifact aims at improving secure code review, by creating awareness and individual empowerment.

Up to and including this section of the paper, the present work has encompassed discussions that introduced aspects related to the first four dimensions of DSR. In the following subsection, we bring the serious game artifact into the spotlight and present a high-level implementation overview, thus touching on the fifth dimension, the *Solution Description*.

B. Implementation

We propose a serious game targeted at developers in the industry to serve as an educational tool geared towards empowering practitioners in performing secure code reviews.

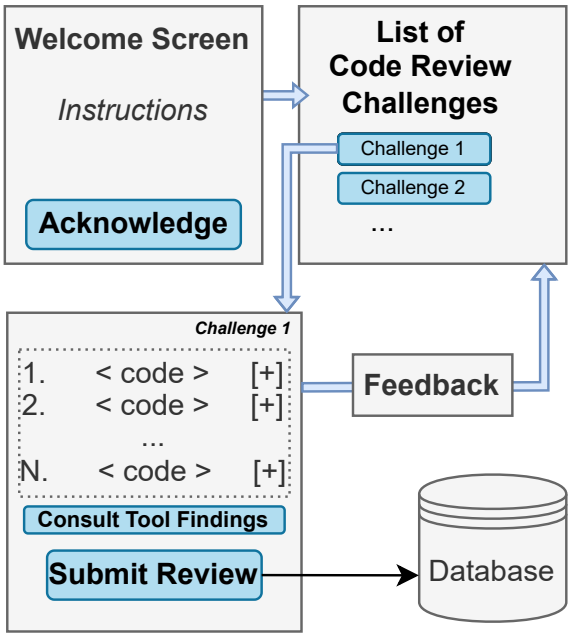


Fig. 1. Artifact Overview

Figure 1 presents our first approach to the game's design. Instructions for the platform are provided to the user on the welcome page. After acknowledging the instructions, they are presented with a list of code review challenges.Upon selecting a challenge, the user receives a source code snippet containing software vulnerabilities and bad practices. The user can select parts of the code and annotate them with comments.

Our serious game encompasses 28 exercises, covering 4 programming languages – Java, C#, Python, and JavaScript. The vulnerabilities included in the game's snippets cover the CWE Top25 [24] and OWASP Top10 [25], to address the most prevalent vulnerabilities encountered in software.

Figure 2 presents the main interface of the serious game. The interface is structured in tabular form, over four columns: **`Comment`**, **`Line Number (#)`**, **`Code`** and **`Solution`**.

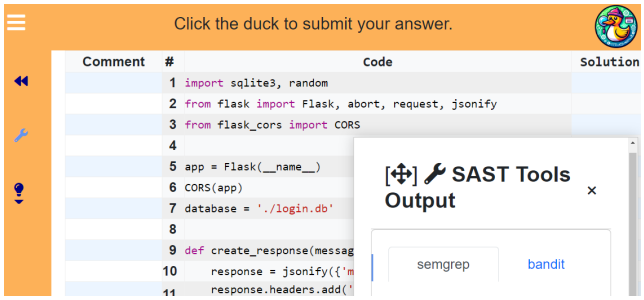


Fig. 2. Artifact Main Interface

The user enters his review comments in the first column. Findings from SAST tools can be consulted in parallel to the source code. By including SAST output in the serious game, we aim to achieve two objectives: firstly, the users are expected to gain familiarity and expertise with using SAST findings in their code review (emulating a real-world industrial tooling environment); secondly, the user is educated on the notions of *false positives/negatives*. By observing the tools' shortcomings or overzealousness, developers can understand where human rationalizing is needed.

When done, the user submits their review, and all their interactions are pushed to a database for later inspection.

Upon finding at least 50% of the vulnerabilities in the snippet, the full intended solution is displayed in the third column for the user to reflect upon, which serves as a direct means of feedback. This percentage stems from a previous work where the authors conducted an experiment consisting of three separate industry interventions [28] in which a simplified version of the proposed artifact was trialed –we observe that developers who undergo security training are able, on average, to identify approximately half of the vulnerabilities that an expert might detect.

The players' progress and ranking is tracked with a dashboard, also accessible to the players, to enhance the game aspect and foster competitivity.

IV. Experiment

This section describes the experimental setup through which we evaluate the artifact.

A. Industrial Intervention

We deployed our serious game artifact throughout a series of industrial interventions. In the context of our study, an *intervention* refers to a workshop on security topics in which the proposed serious game has been embedded for experiential learning.

Across 11 events, 175 participants were presented with the serious game, of which 47 provided feedback about the artifact. Events consisted of secure coding workshops (online, in Germany, and in China), from February 2023 to January 2024. Participants were aged 24-65 and 94.12% graduated from higher education. The vast majority (88.12%) reported an agile working style.

The content was tailored to the participants' areas of expertise. The first two days are spent educating participants on common security antipatterns and their respective mitigations, through a combination of presentations, live demonstrations, and hands-on exercises.

The final day consists of a capture-the-flag-style event where the participants compete against each other in solving various types of challenges – offensive (exploiting binaries or websites), defensive (patching vulnerable programs), and challenges on secure code review, where the participants are presented with the aforementioned artifact and are expected to be able to spot vulnerabilities. The participants could freely choose what types of challenges they wanted to solve.

B. Evaluation and Validation

One of our goals is to foster, through our SG, an environment conducive to enhanced autonomy and effectiveness within the workplace, both being features of empowerment. By assessing usability and perceptions of specific review challenges, we aim to understand how these factors contribute to the empowerment of developers in performing secure code reviews.

The developed artifact is evaluated in terms of usefulness and enjoyability (Graziotin et al. observed that *"happy programmers are good programmers"* [30]).

Our study uses a semi-structured interview method, following Wilson et al. [31], which delivers questions aimed at individually assessing the platform's success in empowering the user to perform secure code reviews. Participants are asked to fill out an anonymous survey. The survey is conducted voluntarily, via Microsoft Forms, to share how they perceive the serious game.

The collected data is devoid of any personally identifiable information. and as such, our analysis focuses on aggregated results from a diverse pool of participants rather than individual-level data. This approach aligns with ethical research practices and industry standards.

V. Results and Discussion

Figure 3 illustrates levels of agreement on game aspects, following a Likert-like scale [32], obtained from the survey responses. **Q1** and **Q3** give insight into how respondents assess their skill in identifying vulnerabilities, within the game's context and professionally. **Q2**, **Q4** and **Q5** explore the participants' reception of the game itself on three implementation aspects - length adequacy, usability, and clarity of instructions.

A substantial majority of respondents concurred that the length of the code snippets was adequate (86.4%, Q2), with a similarly high percentage agreeing that the game's design was usable (81.2%, Q4) and that its instructions were clear (83.3%, Q5).

Conversely, a significant portion of the participants expressed difficulty in identifying vulnerabilities within the code snippets (57.7% combined for "Disagree" and "Neutral," Q3).

Addressing the familiarity with the vulnerabilities presented, the majority stance was neutral, with 60% of participants

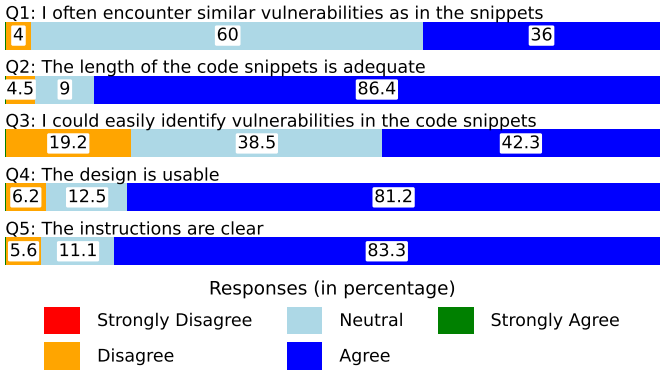


Fig. 3. Agreement Levels towards the Serious Game

neither agreeing nor disagreeing that they often encounter similar vulnerabilities in their work (Q1). Similarly, more than a third of the respondents are neutral in saying that they could easily identify the vulnerabilities presented in the game (Q3, 38.5%). Furthermore, nearly 1 in 5 of the respondents express difficulty in identifying vulnerabilities within the game (Q3, 19.2%). Together, these percentages may hint towards a lack of security awareness and experience of the participants, which may be the scope of further investigative work.

Our design shows that the chosen length of the snippets is positively regarded (86.4%, Q2). The median length of the exercises in the platform is 44 lines of code, with the longest snippet spanning across 116 lines and the shortest across 12. More than 4 out of 5 survey participants report agreement toward designing exercises of this length, this being the highest percentage of agreement recorded in our survey.

It is interesting to note that 19.2% of respondents said that disagreed about being able to easily identify vulnerabilities within the serious game (Q3), while only 4% expressed disagreement towards encountering similar vulnerabilities in their profession (Q1). Based on this difference, we have indicators to believe that this may be due to a lack of awareness of cybersecurity and how to perform reviews for it. Nonetheless, combining the disagreement and neutrality percentages, we reach more similar figures for the two statements – 64% for Q1 and 57.7% for Q3. This is in line with expectations, as the presented vulnerabilities were purposefully selected to cover the most prevalent security defects, as covered in Section III. Nonetheless, further work would be required to explore how the participants' perception differs between the production code and the serious game's snippets.

Concerning feedback on the platform itself, including challenges, we have collected positive feedback (Q2, Q4, Q5). The descriptions of the tasks for each challenge are perceived as being clear, the design is deemed to be usable, and the length of the code snippets is considered to be adequate. These collected responses serve to enhance the trust in the platform and prompt us to contemplate expanding it with additional challenges and further refining the feedback mechanism.

Based on data collected from the industrial interventions, we extend findings from our previous study [28] across a wider range of programming languages and tools. Our present work demonstrates the broader applicability of the following statement: participants receiving minimal security training consistently outperform SAST tools. This is measured through the solve rate of the challenges. By design, the exercises showcase the dangers of tool over-reliance via *false-negatives*. Observing successful solutions, i.e. users finding 50% or more defects, we can imply that our serious game is useful for helping developers be self-reliant and perform effective code reviews, as solving the challenges requires being able to outperform the tools.

Our collected findings from the first set of experiments, together with the feedback from informal discussions with participants, provide a good indicator of the suitability of our design as a means to address secure code review in an industrial context. This aligns with our experience in the field.

Research in the industry comes with limiting factors. Designing a game can be characterized as a *wicked problem*, as per Baskerville [33] and Hevner [29]. The implication is that there is not one optimal solution, and the usefulness of the resulting artifact is influenced by the surrounding social environment as well as social and cognitive capacities. Ensuring a *control group* to evaluate the artifact is also unfeasible in industrial settings, due to these same factors.

The drawn conclusions may be influenced by a restricted number of participants, their backgrounds, functions, and experience levels. Predominantly positive findings may be explained by the survey's voluntary nature and the participant count; those with unfavorable views might have opted out, and awareness of the study's intent might incur positive bias.

However, our findings align with prior research with comparable feedback from both industrial and academic sectors. Thus, we believe that our past and future conclusions would remain largely unchanged even with a larger participant pool. While preliminary indicators point towards the game's suitability, our results are still limited, thus requiring further steps in the design and evaluation process.

VI. Conclusion and Further Work

With the industry transitioning towards integrating security into every step of the software lifecycle, challenges arise in efficiently including adequate practices to assure software quality. One way to tackle code quality is to raise awareness and empower developers on the topic of secure code review. In the present work, we propose to achieve this through a serious game.

This work presents a serious game to raise awareness of code review and its design. We present the motivation behind the work, the importance of the topic in the industrial context in which the work is being carried out, as well as preliminary results from a first design cycle iteration.

We detail how we structure our deployment of the game and show that it can be effectively implemented in diverse settings, with participant pools of varying backgrounds. This facilitates the replication of our study through the repeatability

of the experiment and highlights the serious game's broader application for education and learning.

Our research addresses a pressing industrial requirement: empowering developers with the necessary tools and knowledge for secure code review.

The proposed design of the artifact shows promise. Conclusions drawn from the survey show strong indicators that its format, instructions, and snippets are positively perceived.

In the future design cycles of the work, authors would like to further refine the feedback mechanism, assisting information delivery by employing visual cues and enhancements to the format and broadening the application scope by integrating more programming languages and tools into the artifact.

Good code reviews can only be done with an awareness of secure coding practices and with the necessary structures and the determination to achieve quality code reviews. Participants reported, through empirical evaluation, that the experiential learning activity, conveyed through the serious game, improved their awareness of secure code review. The gained knowledge, specifically towards better recognizing hidden faults in source code, is perceived as useful.

Based on the artifact's positive reception within the industrial target audience, we have reasons to believe that the proposed serious game can be used to foster awareness of secure code review and provide the impulse to create conditions conducive to organizational support for meaningful code review practices.

References

[1] C. Brooks, "Cybersecurity trends & statistics for 2023; what you need to know," bit.ly/forbes23cyber, 2023, "Online, Accessed 2023.10.02".

[2] M. Sánchez-Gordón and R. Colomo-Palacios, "Security as culture: A systematic literature review of DevSecOps," in *Proceedings of the IEEE/ACM 42nd International Conference on Software Engineering Workshops*, ser. ICSEW'20. New York, USA: ACM, 2020, p. 266–269.

[3] R. N. Rajapakse, M. Zahedi, M. A. Babar, and H. Shen, "Challenges and solutions when adopting devsecops: A systematic review," *Information and Software Technology*, vol. 141, p. 106700, 2022.

[4] J. Cohen, *Modern code review*. Sebastopol, CA, USA: O'Reilly, 2010.

[5] L. Braz and A. Bacchelli, "Software security during modern code review: the developer's perspective," in *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. Singapore: ACM, Nov. 2022, p. 810–821.

[6] T. Susi, M. Johannesson, and P. Backlund, "Serious games: An overview," IKI Technical Reports, Tech. Rep., 2007.

[7] N. Hänsch and Z. Benenson, "Specifying IT Security Awareness," in *Proceedings - International Workshop on Database and Expert Systems Applications, DEXA*. Munich, Germany: IEEE, 12 2014, pp. 326–330.

[8] B. Tessem, "Individual empowerment of agile and non-agile software developers in small teams," *Information and software technology*, vol. 56, no. 8, pp. 873–889, 2014.

[9] A. Siami Namin, Z. Aguirre-Muñoz, and K. Jones, "Teaching cyber security through competition an experience report about a participatory training workshop," in *7th Annual International Conference on Computer Science Education: Innovation & Technology (CSEIT 2016)*, ser. CSEIT. Global Science & Technology Forum (GSTF), Oct. 2016.

[10] M. Hendrix, A. Al-Sherbaz, and V. Bloom, "Game based cyber security training: are serious games suitable for cyber security training?" *International Journal of Serious Games*, vol. 3, no. 1, Mar. 2016.

[11] R. Mao, H. Zhang, Q. Dai, H. Huang, G. Rong, H. Shen, L. Chen, and K. Lu, "Preliminary Findings about DevSecOps from Grey Literature," in *2020 IEEE 20th International Conference on Software Quality, Reliability and Security*. Macau, China: IEEE, 2020, pp. 450–457.

[12] A. S. Ami, K. Moran, D. Poshyvanyk, and A. Nadkarni, "False negative – that one is going to kill you: Understanding industry perspectives of static analysis based security testing," *preprint arXiv:2307.16325*, 2023.

[13] K. Rindell, K. Bernsmed, and M. G. Jaatun, "Managing security in software: Or: How I learned to stop worrying and manage the security technical debt," in *Proceedings of the 14th International Conference on Availability, Reliability and Security*. New York, NY, USA: Association for Computing Machinery, 2019, pp. 1–8.

[14] J. A. Kupsch and B. P. Miller, "Manual vs. Automated Vulnerability Assessment: A Case Study," in *First International Workshop on Managing Insider Security Threats (MIST)*. New York, NY, USA: Association for Computing Machinery, 06 2009, pp. 83–97. [Online]. Available: pages.cs.wisc.edu/∼kupsch/va/ManVsAutoVulnAssessment.pdf

[15] A. Bosu, M. Greiler, and C. Bird, "Characteristics of Useful Code Reviews: An Empirical Study at Microsoft," in *2015 IEEE/ACM 12th Working Conference on Mining Software Repositories*. Florence, Italy: IEEE, 2015, pp. 146–156.

[16] L. MacLeod, M. Greiler, M.-A. Storey, C. Bird, and J. Czerwonka, "Code reviewing in the trenches: Challenges and best practices," *IEEE Software*, vol. 35, no. 4, pp. 34–42, 2017.

[17] M. Fu and C. Tantithamthavorn, "Linevul: a transformer-based line-level vulnerability prediction," in *Proceedings of the 19th International Conference on Mining Software Repositories*. Pittsburgh PA USA: ACM, May 2022, p. 608–620.

[18] X. Ramaj, M. Sánchez-Gordón, V. Gkioulos, S. Chockalingam, and R. Colomo-Palacios, "Holding on to compliance while adopting devsecops: An slr," *Electronics*, vol. 11, no. 22, pp. 1–19, 2022.

[19] F. Moyon, D. Mendez, K. Beckers, and S. Klepper, "How to integrate security compliance requirements with agile software engineering at scale?" in *Product-Focused Software Process Improvement*, M. Morisio, M. Torchiano, and A. Jedlitschka, Eds. Cham: Springer International Publishing, 2020, pp. 69–87.

[20] ISO/IEC 64223-4-1:2018-1, "ISO/IEC 62443-4-1:2018 Security for industrial automation and control systems - Part 4-1: Secure product development lifecycle requirements," International Organization for Standardization, Geneva, CH, Standard, 1 2018.

[21] ISO/IEC 64223-4-2:2019-12, "Security for Industrial Automation and Control Systems - Part 4-2: Technical Security Requirements for IACS Components," International Electrical Commission, Geneva, CH, Standard, 1 2019, ISBN 978-2-8322-6597-0.

[22] ISO/IEC 20246:2017, "Software and systems engineering – Work product reviews," International Organization for Standardization, Geneva, CH, Standard, 2017.

[23] ISO/IEC TR 24772-1:2019, "Programming languages – Guidance to avoiding vulnerabilities in programming languages – Part 1: Language-independent guidance," International Organization for Standardization, Geneva, CH, Standard, 2019.

[24] "CWE Top 25 Most Dangerous Software Weaknesses," http://bit.ly/mitre25, MITRE, 2023, online, accessed 2023.07.24.

[25] O. Foundation, "OWASP Top 10 - 2021," https://owasp.org/Top10, OWASP Foundation, 2021, online, accessed 2023.07.24.

[26] M. K. Sein, O. Henfridsson, S. Purao, M. Rossi, and R. Lindgren, "Action Design Research," *MIS Quarterly*, vol. 35, pp. 37–56, 2011.

[27] J. Vom Brocke and A. Maedche, "The DSR grid: six core dimensions for effectively planning and communicating design science research projects," *Electronic Markets*, vol. 29, no. 3, p. 379–385, Sep. 2019.

[28] A.-C. Iosif, T. Espinha Gasiba, U. Lechner, and M.-P. Albuquerque, "Raising awareness in the industry on secure code review practices," in *CYBER 2023: The Eighth International Conference on Cyber-Technologies and Cyber-Systems*. IARIA, September 2023, pp. 62–68.

[29] A. Hevner, S. March, J. Park, and S. Ram, "Design Science in Information Systems Research," *Management Information Systems Quarterly*, vol. 28, pp. 75–, 03 2004.

[30] D. Graziotin, F. Fagerholm, X. Wang, and P. Abrahamsson, "What happens when software developers are (un)happy," *Journal of Systems and Software*, vol. 140, pp. 32–47, 2018.

[31] C. Wilson, "Semi-Structured Interviews," in *Interview Techniques for UX Practitioners*. Burlington, MA, USA: Elsevier, 2014, pp. 23–41.

[32] R. Likert, "A Technique for the Measurement of Attitudes." *Archives of psychology*, vol. 22, no. 140, pp. 1–55, 6 1932. [Online]. Available: bit.ly/lkert

[33] R. L. Baskerville and A. T. Wood-Harper, "A critical perspective on action research as a method for information systems research," *Journal of Information Technology*, vol. 11, no. 3, pp. 235–246, Sep. 1996.