Theme Article: Secure Software Before Code/ing

Empower New Code Citizens to Reflect and Communicate on Code Security

Manuel Maarek, *Heriot-Watt University, Edinburgh, Scotland, UK*

Daisy Abbott, *The Glasgow School of Art, Glasgow, Scotland, UK*

Olga Chatzifoti, *University of Athens, Athens, Greece*

Ryan Shah, *Heriot-Watt University, Edinburgh, Scotland, UK*

Sheun Chi Chan, *Heriot-Watt University, Edinburgh, Scotland, UK*

Hans-Wolfgang Loidl, *Heriot-Watt University, Edinburgh, Scotland, UK*

Robert J. Stewart, *Heriot-Watt University, Edinburgh, Scotland, UK*

Jamie Ferguson, *The Glasgow School of Art, Glasgow, Scotland, UK*

Sandy Louchart, *The University of Edinburgh, Edinburgh, Scotland, UK*

Shenando Stals, *Heriot-Watt University, Edinburgh, Scotland, UK*

Lynne Baillie, *Heriot-Watt University, Edinburgh, Scotland, UK*

Abstract—Democratized access to coding means software is often built by developers with neither formal training nor security knowledge, which could make systems vulnerable. We present an approach based on short games and knowledge cards designed to help these non-experts gain the knowledge and ability to communicate on code security.

Keywords—Developer-Centered Security, Code Security, Software Security Training, Serious Games, Knowledge Cards, Provoking Games

Software is developed in a variety of ways by people with disparate skills and experience: from expert software engineers with extensive knowledge of development methodologies, tools, and verification techniques to hobby developers without any formal training. This diversity, the increased accessibility of code development and deployment, and the availability of generative artificial intelligence (GenAI) for code, all are positive steps. However, they raise issues when considering the security of software codes and systems. For instance, adoption of GenAI with no critical understanding of the code raises security risks as not every coder has the skills, or even awareness to question the use of unvetted code. Even if their code is hosted on development-operations (DevOps) platforms (which offer automated lightweight testing and verifications) the feedback provided might not be understood and consequently bypassed. Therefore, it is important to consider how to best help the wider population of non-expert coders to engage in secure coding.

In this article, we present an *Empowering Code Security Workshop* to help novices and experts alike reflect and communicate on code security. The main audience is coders with limited development experience, no formal training, and who are ignorant (or with limited knowledge) of security issues and their implications. The immediate goal of the workshop is not to teach concrete security skills or techniques but instead to help participants realize that, although security is a complex matter, it can be addressed by every coder regardless of their abilities. The intention in building the participants confidence and ownership of these issues is to foster discussion on security and to provide an entry for further information-seeking activities. The workshop toolkit comprises a deck of knowledge cards and small thought-provoking games. We developed and evaluated this approach through a Slow Game Jam (SGJ) context (explained later in this article) where

Digital Object Identifier 10.1109/XXX.0000.0000000

non-expert participants were invited to design games for code security. This groundwork[1,2,3,4,5] showed evidence the approach can increase confidence in engaging with cybersecurity, and helps non-experts to communicate with experts. The qualitative evaluations with two different novice audiences have limitations as the number of participants was relatively low, and as observations and data collection were limited to the SGJ events.

This article considers existing research focusing on the developer when considering code security. It then presents the two elements of the developed toolkit and demonstrates how to combine these tools into the empowering workshop. The article finally gives a summary of the methodology and evaluations of SGJs.

The tools were designed and developed within the Secrious Project. They are available on the project website secrious.github.io under open licenses.

DEVELOPER-CENTERED SECURITY

A study on developers' perspectives on code security[6] evidenced that security is not a straightforward issue and that developers vary in how they think about and handle code security. In one study,[6] developers performed code reviews and answered four open-ended questions about the codes and their reviews. The research identified diverse conceptions and approaches and inconsistent terminology, therefore the way we invite developers to consider and communicate about security should accommodate this diversity. The study also highlighted that the most prominent code security awareness campaign run by the Open Worldwide Application Security Project (OWASP), the OWASP Top 10, was not known by half of the participants suggesting a need to disseminate code security information more widely, differently, and holistically.

Another study[7] focused on professional software developers to investigate how secure coding guidelines are followed and what constraints and issues they face in following them. The study recommended a number of steps to improve the integration, relevance, and understanding of secure coding guidelines in the software development life cycle. It also highlighted the need for better communication about security (including with less technical team members) and awareness training. This confirms that technical solutions require effort to be integrated into professional practices. Developers faced with technical security challenges might adopt potentially dangerous behaviors to tackle them. Recent research surveyed[8] these behaviors in the context of security application programming interfaces (APIs) uses. It recommends that API providers collaborate with developers to address API security challenges, highlighting the importance of communication within teams and across the coding interfaces.

Securing the New Code Citizen

When aiming to build awareness of code security, the focus is often on software developer professionals or student software engineers. The focus of the Secrious Project is to consider the broader population of coders. This population is expanding for two main reasons: increased accessibility of software development and deployment platforms, and the introduction of programming from primary school onwards. As a result, "coders" now includes a wide variety of backgrounds and ages. We coined *new code citizens* to refer to this population.They create, edit, reuse, build, and run software code without necessarily having prior training or a background in software engineering. They might publish code, deploy it to app stores, or just run their code on personal devices. These new code citizens might not be professional developers nor enthusiast hobbyists, but anyone who has been writing or executing code. Careless but common code practices such as *Copy & Paste* have been shown to affect the security of software applications.[9] It is therefore crucial to motivate new code citizens to secure their code and to assist them in engaging with the complexities of software security.

Cybersecurity Awareness and Education

Addressing cybersecurity threats needs to combine both technical solutions and training users. Training can take the form of University programs, online courses, commercial gamified approaches such as securecodewarrior.com, or freely available resources such as the OWASP Developer Guide: owasp.org/www-project-developer-guide. These resources are essential for security upskilling but are likely to be harder to reach and grasp by non-experts or developers less engaged with securing their code. A detailed analysis of different multimedia approaches to cybersecurity awareness[10] recommends and classifies design principles for such tools based on surveyed publications. It covers a wide range of users in general cybersecurity but fewer works targeting software security which mainly focus on professional software engineers (with the exception of the *Agile App Security Game*, part of an evaluated[11] package of interventions to help software development teams improve their security maturity.)

ACCESSING COMMON CODE SECURITY KNOWLEDGE

Security Knowledge

Knowledge about security is complex and has been rapidly evolving. A multitude of resources organize facets of cybersecurity knowledge for a particular aim. These range from known techniques such as The MITRE Corporation's Adversarial Tactics, Techniques, and Common Knowledge (MITRE ATT&CK) (attack.mitre.org) to the classification of topics for constructing cybersecurity curricula (e.g. the Cybersecurity Curricular Framework: ccecc.acm.org/guidance/cybersecurity, or SPARTA project curricula designer: sparta.eu/curricula-designer). Going beyond curricula creation, the Cybersecurity Body of Knowledge (CyBOK) is a project led by the University of Bristol and funded by the United Kingdom's National Cyber Security Centre (NCSC). It is designed to be open, complete, and to provide a foundation for education and professional training in all fields of cybersecurity by characterizing the concepts, terminologies, and activities in the cybersecurity domain. It is composed of thematic chapters or *knowledge areas* (KAs). Each KA is associated with a number of *topics* (concepts and terminology) organized into branches of the CyBOK *knowledge tree*. CyBOK also provides a portal of training resources mapped to its knowledge base. CyBOK version 1.1[12] comprises 21 KAs of which 3 are dedicated to *Software and Platform Security*.

- *Software Security*.
- *Web & Mobile Security*.
- *Secure Software Lifecycle*.

These 3 KAs on software security, combined with elements of other KAs, make up an exhaustive repository of information on the topic. However, while this information is rich and instructive, it is not necessarily easy to access and understand by users with limited knowledge of computing and security.

Cybersecurity Cards

To facilitate learning software security knowledge, we designed a deck of cybersecurity memo-cards[3] based on CyBOK. We chose CyBOK as a foundation for these cards because of CyBOK's aim of providing cybersecurity knowledge that is not limited to pure techniques and that has a wider focus than curriculum creation. Learning outcomes are:

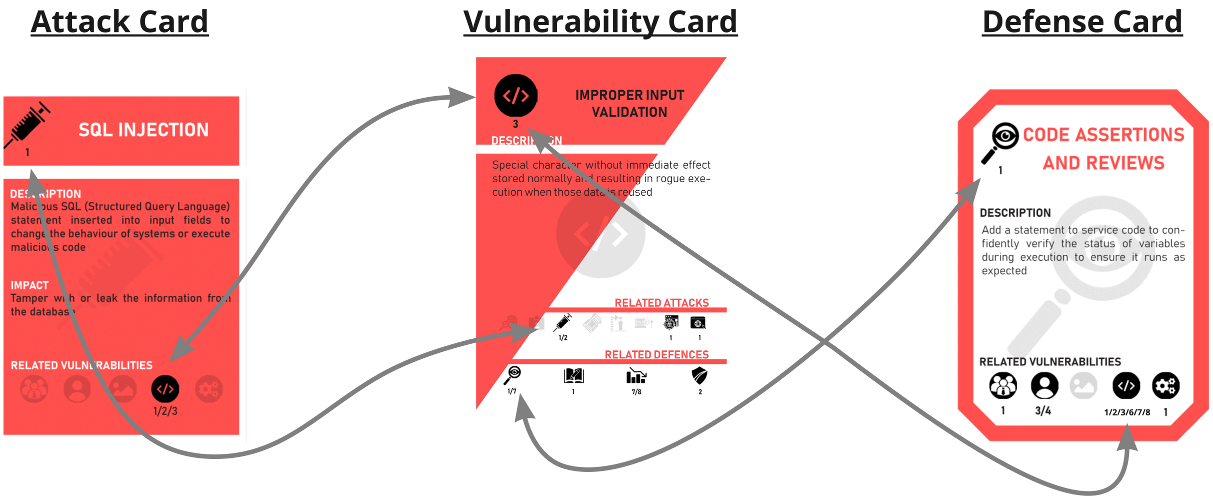
- The cards should provide introductory knowledge on code security.
- The cards should provide a complete overview of code security based on CyBOK.
- The cards should support independent learning.
- The cards should facilitate communication about software security issues and scenarios.

Figure 1 illustrates key design features of the cybersecurity cards. The deck of cards (which is available under open license at github.com/secrious/cybersecurity_cards) comprises 20 attack, 20 vulnerability, and 30 defense cards. We designed the cards in two iterations with associated evaluations[3] which examined how the cards provide a knowledge base to their users, how they foster independent learning, and how they provide an interface for discussion between non-experts and experts. The context of these evaluations was the SGJs which we present in a later section of the paper and in Table 2.

Note that the cybersecurity cards are not a game but rather a deck of knowledge memo-cards. The cards' format helps to structure knowledge into digestible chunks of similar size. It facilitates the manipulation, ownership, and selection of code security concepts, and games can be built using the deck (as demonstrated by the No-Entry game designed by participants of an SGJ[5] which playfully explores the knowledge represented.) Card games for security awareness are not new, other examples include: OWASP Cornucopia: owasp.org/www-project-cornucopia, Control-Alt-Hack,[13] and Riskio.[14] These existing decks for security were developed within games and therefore the information they contain is restricted by, if not oriented to, the mechanics of those games. In contrast, our deck of cybersecurity cards is built from a comprehensive and complete knowledge base: CyBOK.

TRIGGERING REFLECTION ON CODE SECURITY

Motivating coders to consider security or to take up training exercises can be facilitated by using game-based approaches. The range of such approaches in cybersecurity is wide,[10] as illustrated in the remainder of this paragraph. Simple gamification could motivate the developer to engage with code verification using badges or leaderboards based on usage metrics. A role-playing game might put practitioners in an experiential (game) context to prepare them if such a situation was to occur in their organization. A serious game could address a training need by embedding learning objectives supported by appropriate game mechanics into a stand-alone game, potentially more


Attack Card
Vulnerability Card
Defense Card
SQL INJECTION
1
DESCRIPTION
Malicious SQL (Structured Query Language) statement inserted into input fields to change the behaviour of systems or execute malicious code
IMPACT
Tamper with or leak the information from the database
RELATED VULNERABILITIES
1/2/3
IMPROPER INPUT VALIDATION
3
DESCRIPTION
Special character without immediate effect stored normally and resulting in rogue execution when those data is reused
RELATED ATTACKS
1/2
1
1
RELATED DEFENCES
1/7
1
7/8
2
CODE ASSERTIONS AND REVIEWS
1
DESCRIPTION
Add a statement to service code to confidently verify the status of variables during execution to ensure it runs as expected
RELATED VULNERABILITIES
1
3/4
1/2/3/6/7/8
1

- Each card contains brief information on a code security concept with a clear title using common terminology.
- Cards are classified by topics signaled by unique icons, and are identified by a unique number within each topic.
- The deck is decomposed into the attacks-vulnerabilities-defenses trichotomy.
- Links between cards highlight vulnerabilities-attacks, and vulnerabilities-defenses relationships.
- There is no direct link between attacks and defenses, to give vulnerabilities a central position and instill the notion of attack surface.
- There are no one-to-one single links between a vulnerability and either an attack or a defense. This conveys the message that there are no straightforward solutions to a security risk.
- The deck is constructed[3] to cover and rationalize all CyBOK topics of the CyBOK knowledge tree relevant for code security.

FIGURE 1. Example of Cybersecurity Cards and their Relationships

motivating than other training activities.

Provoking Games

The approach we took was to create serious games designed specifically to provoke reflection. These games aim not to convey specific knowledge but rather to make the player think and initiate discussion and reflection, matching our intention to create behavioral change in non-experts.

We designed and developed[2] a set of three small provoking games covering the three main security KAs of CyBOK. The short digital games are available for free at secrious-research-project.itch.io.

- *Protection* focuses on code security.
- *Collaboration* focuses on the security life cycle and human factors.
- *API-ary* focuses on API security.

Although the games can be played independently, they share common elements that relate them to each other. Games are deliberately ambiguous or have unexpected twists, provoking dialogue and encouraging players to think about both security and their own roles within a security context. These games set the scene for the player to develop their understanding, preparing them for further activities.

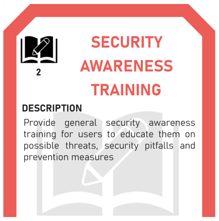
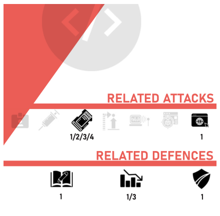
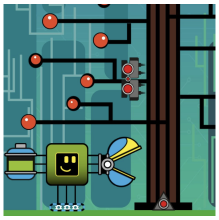
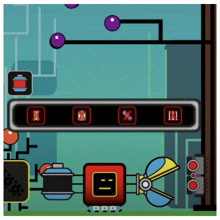
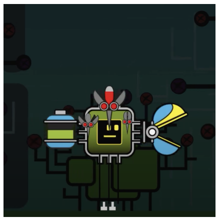
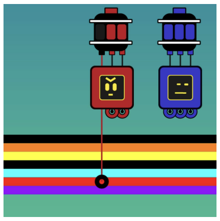
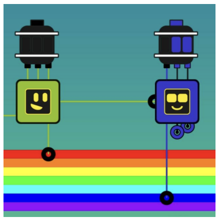
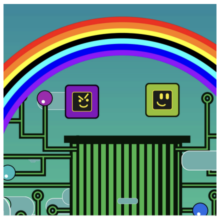
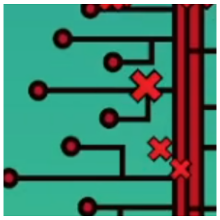
Within the empowering code security workshop, participants engage in group reflection to analyze the meaning(s) of the provoking games. They use the cybersecurity cards as an analysis tool. Because of the ambiguity of the provoking games, this deconstruction phase forces questioning and discussion within each group, setting the stage for participants to then construct their own code security scenario.

The relationship between the games' features and their reflective motives are summarized and illustrated in Table 1.

Protection. In this game, the player explores a digital rainforest environment as an electronic/biological hybrid character, discovering berries to eat and discerning through experimentation which have negative effects. The character meets malicious entities, and can discover and equip different add-ons that extend its capabilities and may offer some protection against the risks from berries and attackers.

The player character is rapidly exposed to risks and threats, showing that the need for security is an immediate, important concern. They must identify counter-

TABLE 1. General Correspondance between Code Security Misconceptions, Reflective Motives, and Cybersecurity Cards/Small Provoking Games Features

Misconceptions	Reflective Motives	Features of Cards and Games
Code security is for experts	Provide comprehensive code security content in digestible format	 SECURITY AWARENESS TRAINING 2 DESCRIPTION Provide general security awareness training for users to educate them on possible threats, security pitfalls and prevention measures  RELATED ATTACKS 1/2/3/4 1 RELATED DEFENCES 1 1/3 1
Security is binary with a single solution per issue	One-to-many links with vulnerabilities, and no one-to-one link between attacks and defenses	
Relations between threats and mitigation are straightforward	Discover relationships between threats and mitigation	 
The more security, the better	Illustrate the impact of excessive security preventing functionality or usability	
There are no impacts to security mitigation	Make security requirements be an integral part of systems requirements	 
Security is only an issue for security experts	Show the impact of disengaging from security matters	
There is a technical solution to all security issues	Illustrate that security technical solutions benefit from being inclusive	 
Security is every team's top priority	Highlight that communication about security requires energy and time which need to be costed	
Using this library works so it can just be added to my code	Show the danger in not vetting library code and external data in the system being developed	 
This piece of code is not important so does not need to be secured	Illustrate the impact of insecure code on the ecosystem to advocate for security by default	

The last column displays illustration details of the cybersecurity cards (first row), *Protection* game (second row and first image of third row), *Collaboration* game (second image of third row and fourth row), and *API-ary* game (last row).

measures but are also exposed to the complex relationships between threats, mitigation strategies, and consequences of mitigation. The rainforest evolves, requiring the player to adapt their strategies. This aims to provoke reflection on the complexity of security, its trade-offs, and the need for constant re-assessment to adapt to a changing landscape. A narrative twist in the game occurs when the player realizes that equipping every upgrade at once impedes the movement of their creature so much as to render it dormant. Code security is not simply a layer of upgrades that are added to a system with a preconceived view that *the more security, the better*. Instead, security is an integral part of the system with its requirements needing to be balanced with others such as functionality and usability. The game forces the player to analyze the situations, assets, threats, and capabilities and evaluate appropriate solutions, inviting them to reflect on the security dimension of their own coding practices and to take ownership over security issues.

Collaboration. The game is located above the rainforest where a group of hybrid creatures similar to the previous game are co-working on their infrastructure: seven colored lanes which the team is trying to turn into a rainbow. The rainforest and its trees (representing the wider ecosystem) are affected by the data flow coming from the colored lanes e.g. corrupted lanes produce harmful acidic rain.

At first, the player sees the creatures as independent workers and instructs them to do appropriate tasks. However, it rapidly becomes clear that rainbow production is a group effort requiring communication between members of the team — which uses up energy. The quality and security of the end-product reflect the quality of communication within the team. Each character has a unique expertise and mood which makes them more or less receptive to different forms of communication, only discoverable by experimentation. To solve the different levels of the game, the player must understand both the needs and abilities of members of the team. This aims to convey that, in technical contexts where people have specialized skills, the amount of energy taken by communications can often be underestimated. In teamwork, an overview of security processes and investing effort to build good communication practices is impactful. Soft skills, which are often undervalued, contribute to building more secure systems. The game invites the player to reflect on their own skills and to evaluate the collaborative dimension of security.

API-ary. The third game takes place in a single tree. The player manages an "APIary" where bees enter from the wider rainforest, carrying berries similar to those of the first game. Each berry is dropped and produces tree shoots which form part of the rainforest ecosystem. However, some bees only steal nectar from the tree whilst contributing nothing and some berries create corrupted tree shoots that damage the ecosystem. The player must identify the different effects to understand that it is critical to evaluate resources. They can then configure a mechanism to filter out unwanted bees or damaging berries, whilst still allowing the helpful bees through. This is achieved through experimentation and observation. The goal is to provoke construction of knowledge on the importance of safeguarding data and codes that are making up the system being developed. Security configurations are to be understood to adequately guard the system. A deeper consideration of the game metaphor may lead to the realization that the code one produces could be reused by others or enter other systems in ways that were not necessarily foreseen. It is therefore important to provide quality well-documented contributions as this impacts the health of the ecosystem. The game invites the player to reflect on the role of APIs both from the consumer and producer perspectives, and the role they can play within the broader open-source community.

Transformative Reflection

The small provoking games are designed to provoke dialogue, knowledge exchange, and self-reflection. They aim to produce transformative reflection through subverting expectations to trigger inquiry and analysis. By using cognitive and affective challenges they invite the player to actively construct meaning from the game and their own context.[2] The achievements within the game increase a sense of purpose and should empower the player to question their role and gain knowledge and confidence.

CODE SECURITY EMPOWERING WORKSHOP

We now present our approach combining cybersecurity cards and provoking games as part of a workshop to empower novices and experts to address code security issues. The workshop consists of the following steps.

1) Cybersecurity cards introduction.
2) Deconstructing provoking games.
3) Building code security interventions.

This workshop is a subset of the SGJ, a methodology developed[1] and evaluated[4,5] within the Secrious project, see below for more details. The three steps of the workshop correspond to activities taking place on the first day of the 6-day SGJ[4] and 5-day SGJ.[5]

Cybersecurity Cards Introduction

In step one, workshop participants familiarize themselves with the cybersecurity card deck. Participants receive the deck in either a physical or digital format. The structure and scope of the deck are presented with an accompanying activity that involves navigating through the deck and identifying the cards' relationships.

Deconstructing Provoking Games

The second step uses provoking games to spark reflection and interest in exploring the topic of code

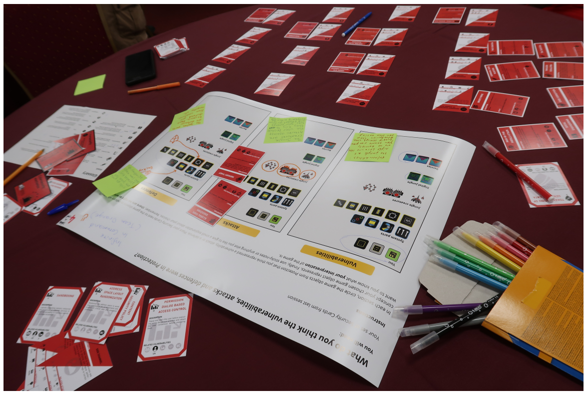


FIGURE 2. Example of Small Provoking Game Deconstruction by SGJ Participants

security. Participants play (in groups) one of the three provoking games presented earlier. They are then invited to deconstruct the game by identifying the security aspects covered by different elements of the game. They use cybersecurity cards and images of game assets to draw parallels and relationships between game elements and security concepts, as illustrated in Figure 2. There is no single correct answer to this exercise as the fundamental goal is for the participants to continue and consolidate the reflections triggered by the game and to concretely explore the concepts presented in the cybersecurity cards. Depending on the time available, groups can play more than one game. This step could see a group of participants presenting and discussing their findings with another group for them to cement and expand their grasp of the games and their associated concepts.

Build Code Security Interventions

In the last step of the workshop, participants are invited to articulate a code security intervention based on their interests and experiences in code security, or based on what they believe needs addressing in this domain. Each group uses the cards to navigate through concepts, helping them to reinforce understanding, and select or discard concepts as appropriate to their task. They derive an intervention by combining cards, researching the topic further, and inquiring with available expertise. Each intervention should describe a code security problem, identify factors related to the problem, and define an intervention to address the security issue. The intervention produced by each group could then be presented to other groups to exchange feedback which could also be provided by experts attending or delivering the workshop.

Requirements and Uses of the Empowering Code Security Workshop

Running the empowering workshop requires participants to have access to: a device to play the provoking games; printed or digital copies of the cards; and a physical or digital space to draw their deconstruction and intervention. The workshop does not actually require any access to a code development facility. The workshop requires a facilitator with knowledge of code security to provide expert guidance or perspectives. The workshop can be run in a half-day or a day depending on the number of provoking games to be played. In organizations where security is not the core focus, such expertise might exist as a single role or single person with security responsibility, in this case, the workshop could serve as a medium to engage the rest of the organization to consider code security. As the workshop is an entry point for further resources, it is best suited as a digital skills awareness event or an introduction to more technical training.

The template of the workshop can be adapted and expanded to fit specific goals or needs. As indicated above, these steps are a subset of the SGJ methodology[1,4,5] developed to co-design serious games for code security with new code citizens. The code security interventions in this context are implemented into a serious game. The additional steps in the SGJ consist of identifying the learning mechanics and game mechanics to be deployed to implement the code security intervention through a serious game.

SLOW GAME JAM EVALUATION FINDINGS

SGJ[1] is a methodology developed to engage non-expert participants in the co-design and development of serious games.

The methodology is grounded on the Triadic Game Design (TGD) approach[15] which emphasizes finding the right balance between the three constituents of a serious game: Reality (the context the game addresses), Meaning (the learning purpose the game has), and Play (the motivating and entertaining aspect). The cybersecurity cards and provoking games presented in this paper were developed to match the Secrious context of code security.

> *"I feel I have to learn a lot on cybersecurity. But the workshop has more than enough knowledge to get you started and understand cybersecurity."* (Participant in the SGJ with students[4])

The details of the design of the cards,[3] provoking games,[2] SGJ methodology,[1] and the evaluations of the methodology,[4,5] are outputs of the Secrious project of which Table 2 provides a comprehensive summary. Although these findings and their limitations are in the context of the SGJs, they provide meaningful insights into the relevance of the workshop presented here as a method to engage non-experts with code security matters, and provide evidence of the suitability of cybersecurity cards to be a medium of security information and for communication between non-experts and experts.

CONCLUSION

Code security should be considered by anyone who codes since software-based systems are ubiquitous in most activities of our everyday lives. Addressing code security issues has to include all stakeholders regardless of their security expertise or software development proficiency. We presented an approach that includes the wider population to empower coders to tackle code security and to be able to constructively communicate with their peers and experts. The evaluation of this approach in the SGJ context showed, although with limitations, that it provides non-experts with confidence in engaging with cybersecurity, and helps them to communicate with experts about code security. Such approaches that lower the barriers to entry to be part of security solutions are essential for building inclusive and resilient communities of code citizens. Serious games and user-centered approaches combined offer the appropriate tools to foster reflection and communication that include novices and experts alike.

ACKNOWLEDGMENTS

This work contributes to the SECRIOUS project and is supported by the Engineering and Physical Research Council (Grant ID: EP/T017511/1).

The authors thank our collaborators at Civic Digits, a company that works at the intersection of games, performance and technology. Civic Digits' director, Dr Clare Duffy is a playwright and theatre director and Rupert Goodwins is a technical journalist with a specialism in cybersecurity.

REFERENCES

1. D. Abbott, O. Chatzifoti, J. Ferguson, S. Louchart, and S. Stals, "Serious 'Slow' Game Jam - A Game Jam Model for Serious Game Design," in *ACM Proceedings of the 7th International Conference on Game Jams, Hackathons and Game Creation Events*, ser. ICGJ '23, 2023, pp. 28–36. [Online]. Available: https://dl.acm.org/doi/10.1145/3610602.3610604
2. D. Abbott, S. Louchart, and O. Chatzifoti, "Provocative Games to Encourage Critical Reflection," *European Conference on Games Based Learning*, vol. 16, no. 1, pp. 1–10, Sep. 2022, number: 1. [Online]. Available: https://papers.academic-conferences.org/index.php/ecgbl/article/view/486
3. R. Shah, M. Maarek, S. Stals, L. Baillie, S. C. Chan, R. Stewart, H.-W. Loidl, and O. Chatzifoti, "Introducing and Interfacing with Cybersecurity – A Cards Approach," 2023, presented as part of the activity at the Deconstructing Gamified Approaches to Security and Privacy DGASP Workshop at SOUPS'23. [Online]. Available: http://arxiv.org/abs/2307.16535
4. S. Stals, L. Baillie, J. Ferguson, D. Abbott, M. Maarek, R. Shah, and S. Louchart, "Evaluating slow game jams as a mechanism for co-designing serious games to enhance understanding of cybersecurity," *ACM Games: Research and Practice*, 2025.
5. S. Stals, L. Baillie, R. Shah, J. I. Ferguson, and M. Maarek, "Evaluating and validating the serious slow game jam methodology as a mechanism for co-designing serious games to improve understanding of cybersecurity for different demographics," *Computer Standards & Interfaces*, vol. 92, p. 103924, 2025. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S092054892400093X
6. I. Rauf, T. Lopez, H. Sharp, M. Petre, T. Tun, M. Levine, J. Towse, D. van der Linden, A. Rashid, and B. Nuseibeh, "Influences of developers' perspectives on their engagement with security in code," in *ACM Proceedings of the 15th International Conference on Cooperative and Human Aspects of Software Engineering*, ser. CHASE '22, 2022, pp. 86–95. [Online]. Available: https://doi.org/10.1145/3528579.3529180
7. T. Espinha Gasiba, U. Lechner, M. Pinto-Albuquerque, and D. Méndez, "Is Secure Coding Education in the Industry Needed? An Investigation Through a Large Scale Survey," in *IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, 2021, pp. 241–252. [Online]. Available: https://ieeexplore.ieee.org/document/9402184
8. P. D. Chowdhury, J. Hallett, N. Patnaik, M. Tahaei, and A. Rashid, "Developers Are Neither Enemies Nor Users: They Are Collaborators," in *IEEE Secure Development Conference (SecDev)*, 2021, pp. 47–55. [Online]. Available: https://ieeexplore.ieee.org/

TABLE 2. Summary of the SGJs and their Evaluations

	SGJ with Students[4]	Summer School SGJ[5]
Theme	*Secure Software Lifecycle*	*Software Security*
Cybersecurity cards version[3]	Version 1. 124-card deck composed of 30 vulnerability, 32 attack and, 47 defense cards, each of which are categorized under one of the 15 general cards	Version 2. 70-card deck composed of 20 vulnerability, 20 attack, and 30 defense cards, with a glossary replacing general cards, with improved linkage between cards (changes informed by feedback from the SGJ with Students)
Provoking game[2]	*Collaboration*	*Protection*
Methodology[1]	3 phases: design, development, development & prototyping, supported by experts in security, learning, and game; deliverables at each phase	
	Each phase composed of 2 consecutive days, spread over 5 weeks	5 consecutive days (1 week), phases of 2, 1, 2 days respectively
Evaluation methods[4,5,3]	Pre/post Likert questionnaires to measure knowledge; two index-questionnaires at multiple points of the event to measure workload, motivation, and engagement; peer feedback; questionnaire to evaluate how the three decks of cards used (cybersecurity, learning mechanics, game mechanics) provided a knowledge base, supported independent learning and self-efficacy, provided an interface for discussion	
Participants	13 participants split into 3 teams; students in first year of conversion MSc programs in computer science (not in cyber security); 11 out of the 13 answered the cybersecurity cards questionnaire	23 participants split into 6 teams; 11–16 years old in late primary or secondary school
Findings[4,5,3]	The students' confidence in their knowledge and understanding rose from 12.5% to 62.5%; 82% of the students agreed that the cybersecurity cards provide introductory cybersecurity knowledge to novice users; in terms of the cards' self-efficacy, 82% of the students agreed that even without a cybersecurity expert present in the team, they were able to access cybersecurity knowledge solely using our cybersecurity cards; students self-reported in free text answers (coded by the researchers) that what they had learned was the different types of security vulnerabilities, attacks and defenses for three quarters of the students; factors influencing cybersecurity (e.g. human factors) for half of the students; and the vulnerabilities-attacks-defenses relationships as well as terminology for almost a third of the students	The pupils' confidence in their knowledge and understanding of cybersecurity rose from 41.2% to 76.5%; 70% of the pupils agreed that the cybersecurity cards provide introductory cybersecurity knowledge to novice users; in terms of the cards' self-efficacy, 57% of the pupils agreed on this compared to understanding topics independently; pupils self-reported in free text answers (coded by the researchers) that the main thing they had learned was a greater general awareness of cybersecurity for a third of the pupils; learning more about one specific attack, defense, or vulnerability for a quarter of the pupils
Limitations	The relatively low number of participants, the measure of confidence in knowledge rather than assessed knowledge, the evaluation with pre-/post-tests (there is a possibility the difference in confidence could partially be by factors external to the SGJ) are limitations which are mitigated by the qualitative evaluation of participants' knowledge and understanding (including the identification of cybersecurity metaphors in the provoking game using cards, the selection of cards for their own game, the assessment, discussion, and peer assessment of the selection of cards for their own serious game); The evaluation is only based on data collection at and immediately after the SGJ, further experiments would need to be conducted to evaluate behavior change or longitudinal impact.	

abstract/document/9652651

9. F. Fischer, K. Böttinger, H. Xiao, C. Stransky, Y. Acar, M. Backes, and S. Fahl, "Stack Overflow Considered Harmful? The Impact of Copy&Paste on Android Application Security," in *IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 121–136. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/7958574
10. L. Zhang-Kennedy and S. Chiasson, "A Systematic Review of Multimedia Tools for Cybersecurity Awareness and Education," *ACM Computing Surveys*, vol. 54, no. 1, pp. 12:1–12:39, 2021. [Online]. Available: https://dl.acm.org/doi/10.1145/3427920
11. C. Weir, I. Becker, and L. Blair, "A Passion for Security: Intervening to Help Software Developers," in *IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, 2021, pp. 21–30. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9402146
12. CyBOK, "The Cyber Security Body of Knowledge, Knowledgebase — CyBOK v1.1," 2021, accessed 19/07/2023. [Online]. Available: https://www.cybok.org/knowledgebase1_1/
13. T. Denning, A. Lerner, A. Shostack, and T. Kohno, "Control-Alt-Hack: the design and evaluation of a card game for computer security awareness and education," in *Proceedings of the ACM SIGSAC conference on Computer & communications security*, ser. CCS '13, 2013, pp. 915–928. [Online]. Available: https://dl.acm.org/doi/10.1145/2508859.2516753
14. S. Hart, A. Margheri, F. Paci, and V. Sassone, "Riskio: A Serious Game for Cyber Security Awareness and Education," *Computers & Security*, vol. 95, p. 101827, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404820301012
15. C. Harteveld, *Triadic Game Design: Balancing Reality, Meaning and Play*, 1st ed. Springer, 2011. [Online]. Available: https://doi.org/10.1007/978-1-84996-157-8

Manuel Maarek is an associate professor at Heriot-Watt University. His research interests are on serious games for cybersecurity, cybersecurity knowledge and education, developer-centred security, software engineering, formal methods, programming languages, software safety and security. He received his PhD degree in computer science from Heriot-Watt University. Contact him at M.Maarek@hw.ac.uk

Daisy Abbott is a Game-Based Learning (GBL) researcher and designer based at The Glasgow School of Art. She specialises in playful approaches to learning within Higher Education. She received her MSc degree in Information Technology (Humanities) from the University of Glasgow. Contact her at D.Abbott@gsa.ac.uk

Olga Chatzifoti was research assistant at The Glasgow School of Art. Her expertise lies in digital and digitised spatiality with a focus in extended reality (XR) environments. She received her MSc degree in serious games & virtual reality from The Glasgow School of Art. She is currently active as an independent XR developer and is a PhD candidate in the University of Athens.

Sheung Chi Chan was a research assistant at Heriot-Watt University. His research interests include data analysis, system and network security, security testing and fuzzing, and security education and awareness training. He received his PhD degree in Computer Science from the University of Edinburgh. Sheung Chi Chan is now researcher at AdaLogics.

Ryan Shah was a research assistant at Heriot-Watt University. His research interests are in cybersecurity and include the security of cyber-physical systems and robotics security. He received his PhD degree in cybersecurity (computer and information sciences) from the University of Strathclyde. He is now senior security consultant at Sapphire.

Hans-Wolfgang Loidl is an associate professor at Heriot-Watt University. His research interests are in distributed and parallel programming, programming languages, foundations of programming languages, and more recently high-performance machine learning and serious games. He received his PhD degree from the University of Glasgow.

Robert J. Stewart is an associate professor at Heriot-Watt University. His research interests are at the intersection between high level programming models and low level system architectures. He received his PhD degree from Heriot-Watt University.

Jamie Ferguson is a lecturer at The Glasgow School of Art. Her research interests are on game design, programming, audio design, haptic interaction design, and human-computer interaction. She received her PhD degree in human computer interaction from the University of Glasgow.

Sandy Louchart is a research associate in Game Design at the University of Edinburgh. He was the head of Undergraduate Programmes at the school of

Innovation and Technology at The Glasgow School of Art. His research investigates the domains of interactive storytelling via the development of the emergent narrative concept and serious games design from the perspective of serious games mechanics. He received his PhD degree in computer sciences and artificial intelligence from the University of Salford.

Shenando Stals is a research associate at Heriot-Watt University. His research interests are on human computer interaction and human robot interaction. He received his PhD degree in computing from Edinburgh Napier University.

Lynne Baillie is a professor at Heriot-Watt University. Her research focuses on the development of novel user-centred methods to facilitate the design and evaluation of new applications and technologies for pervasive and ubiquitous contexts. She received her PhD degree in human computer interaction from Edinburgh Napier University.