# CyberSecurity Challenges: Serious Games for Awareness Training in Industrial Environments

Tiago Gasiba[1,2], Ulrike Lechner[2], and Maria Pinto-Albuquerque[3]

[1] Siemens AG, Munich, Germany
tiago.gasiba@siemens.com
[2] Universität der Bundeswehr München, Munich, Germany
tiago.gasiba@unibw.de ulrike.lechner@unibw.de
[3] Instituto Universitário de Lisboa (ISCTE-IUL), ISTAR, Lisboa, Portugal
maria.albuquerque@iscte-iul.pt

**Abstract.** Awareness of cybersecurity topics, in particular related with secure coding guidelines, enables software developers to write secure code. This awareness is especially important in industrial environments for the products and services in critical infrastructures. In this work, we introduce and discuss a new serious game which is designed for software developers in the industry. This game addresses software developers' needs and is shown to be well suited for its purpose: raise secure coding awareness of software developers in the industry. Our work is the result of the experience of the authors gained in conducting more than ten CyberSecurity Challenges in the industry. The presented game design, which is shown to be well accepted by software developers, is a novel alternative to standard classroom training. We hope to make a positive impact in the industry by improving the cybersecurity of products at their early stages of production.

**Keywords:** IT Security · Cybersecurity · Awareness · Secure Software Development · Industry · Critical Infrastructures · Serious Game

## 1 Introduction

If not addressed during the early stages of software design and implementation, software development errors and security vulnerabilities can end up in a final product or service. Security vulnerabilities can result in serious negative consequences, for society, for the customer, and also for the company that produced the software. Think, e.g., of critical infrastructures as the grid, transportation, or production lines: a security vulnerability in the code may cause interruptions in service quality for individual customers when critical machinery or information system fail or even for society, when critical infrastructure fails. Over the last years, the number of industrial security-related incidents has been increasing, which has resulted in severe incidents, leading to a huge financial impact, reaching up to 1.6% of GDP in some EU countries [7].

To address these issues, products and services provided by the industry, must follow IT security standards. These standards both mandate the implementation

of a secure software development lifecycle, and also provide secure coding guidelines that must be followed to write secure code. Prominent examples of these standards for industrial environments are the IEC 62443 [23], ISO 27001 [24] and the Grundschutzkatalog from the Bundesamt für Sicherheit in der Informationstechnik (BSI) [4]. Examples of secure coding guidelines widely used in the industry are the SEI-CERT Java Secure Coding Guidelines and SEI-CERT C/C++ Secure Coding Guidelines, both from Carnegie Mellon [5]. Secure Coding Guidelines which are specific for web application development and widely used are provided by the Open Web Application Security Project (OWASP, [27]) and from the BSI (BSI 5.21, [3]).

These standards provide a much-needed basis that establishes ground rules required to produce secure products and services. The effectiveness of these standards is related to the level of awareness and understanding of the standards by the persons that are directly affected by them: software developers. However, a recent study by Patel et al. [28] has shown that more than 50% of software developers cannot spot software vulnerabilities in source code. This is a problem that needs to be addressed: the lack of awareness about secure coding.

Among others, a possible way to address this issue is to provide training to software developers on the topic of secure coding. In this work we present a new serious games designed to raise awareness and train software developers in secure coding. The serious game, named CyberSecurity Challenges, is an adaption of the capture-the-flag game genre. Capture-the-flag were originally developed in the penetration testing community as a means to train and exercise offensive IT-security skills. The idea is that by attacking a system, penetration testers can reveal vulnerabilities which can be fixed, after reporting back to the development team. However, these activities take place late in the software development stages. We propose to use an adapted version of the game, which targets software developers, focuses on the defensive perspective and has the main goal to increase awareness of secure coding guidelines and secure coding best practices. Furthermore, we not only show how our concept can be used for on-site IT-Security Awareness Workshops, but also how it can be adapted for online training.

This work is organized as follows: in section 2, the authors briefly discuss previous work which is related to the cybersecurity challenges. Section 3 introduces the CyberSecurity Challenges and discusses challenges based on open-source components and on the Sifu platform. Section 4 discusses evaluation of the games in an industrial context through survey results, participant feedback and lessons learned. Finally, section 5 summarizes and concludes the paper.

## 2 Related Work

Although several methods exist to deal with software vulnerabilities, e.g., requirements engineering and code reviews, we focus on awareness training for software developers. Several previous studies indicate that software developers lack secure programming awareness and skills [1,28,32]. In 2020, Bruce Schneier,

a well-known security researcher, and evangelist stated that *less than 50% of software developers can spot security vulnerabilities in software* [30]. His comment adds to a discussion on secure coding skills: In 2011, Xie et al. [33] did several interviews with 15 senior professional software developers in the industry with an average of 12 years of experience. Their study has shown a disconnect between software security concepts and their role in their jobs. Awareness training on Information security is addressed in McIlwraith [25], which looks at employee behavior and provides a systematic methodology and a baseline on implementing awareness training.

There is a stream of literature on compliance with security policies, which deals with employees in general and not with software developers specifically. This stream of literature explores many reasons why people do not comply with IT-security policies. The unified framework by Moody et al. [26] summarizes the academic discussion on compliance with IT-security policies. Empirical findings conclude that neither deterrence nor punishment such as e.g., public blame, works to increase compliance. However, increasing IT-security awareness increases the level of compliance [31]. In their seminal review article, Hänsch et al. [22] define IT-security awareness in the three dimensions: *Perception* (knowledge of existing software vulnerabilities), *Protection* (knowing the existing mechanisms - best practices - that avoid software vulnerabilities), and *Behavior* (knowledge and intention to write secure code). The concept of IT-security awareness is typically used in IT security management contexts.

Graziotin et al. [21] show that *happy developers are better coders*, i.e., produce higher quality code and software. Their work suggests that by keeping developers happy, we can expect that the code they write has a better quality and, by implication, be more secure. Davis et al. [6] show, in their construct, that cybersecurity games have the potential to increase the overall happiness of software developers. Their conclusions support our approach to use a serious game to train software developers in secure coding. Awareness games are a well-established instrument in information security and are discussed in de-facto standards as the BSI Grundschutz-Katalog [4] (M 3.47, Planspiele) as one means to raise awareness and increase the level of security. Frey et al. [8] show both the potential impact of playing cybersecurity games on the participants and show the importance of playing games as a means of cybersecurity awareness. They conclude that cybersecurity games can be a useful means to build a common understanding of security issues. Rieb et al. [29] provide a review of serious games in cybersecurity and conclude that there are many approaches. The games listed mainly address information security rather than secure coding. Documented and evaluated games are [2] and [29].

Capture-the-flag is one particular genre of serious games in the domain of Cybersecurity [6]. Game participants win flags when they manage to solve a task. Forensics, cryptography, and penetration testings are skills necessary for solving tasks and capturing flags. The present work uses serious games to achieve the goal of *raising secure coding awareness of software developers in the industry.*

Previous work on selected design aspects and a smaller empirical basis on the CSC includes [10, 13–15, 18–20].

## 3 CyberSecurity Challenges

In this section we introduce the CyberSecurity Challenges (CSC), which were developed in the industry as a means to raise awareness on secure coding. We also present a detailed discussion on how to create these games based on two different methods: (1) by using existing open-source components, and (2) by using an open source platform developed by the authors - the Sifu platform.

### 3.1 What are CyberSecurity Challenges

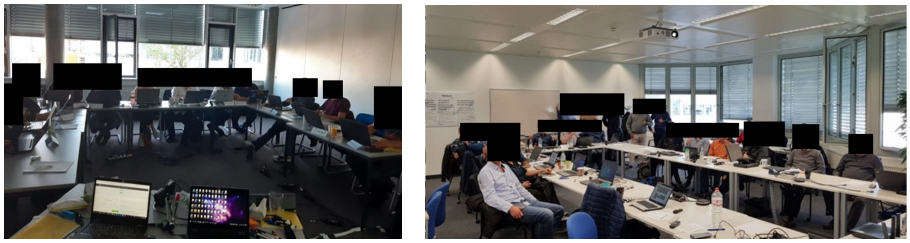

**Fig. 1.** CyberSecurity Events - On-site Events

CyberSecurity Challenges (CSC) are a genre of serious games developed with the purpose to raise awareness of industrial software developers in the topic of secure coding and secure coding guidelines. Figure 1 shows two examples of CSC events that took place in the industry.

The game consists of a platform where several participants (i.e. software developers) form teams that compete against each other in solving secure coding challenges. The challenges consist of exercises which are developed specially to address software development vulnerabilities. Solving the challenges requires the participants to know and follow secure coding guidelines. Figure 2 shows the general architecture of CyberSecurity Challenges (CSC), which consists of the following components: Challenges, Dashboard, and Countdown.

The challenges represent the individual exercises that the participants must solve to gain points. The dashboard displays the available challenges and is used to control the current status of each team in terms of number of gathered points Figure 3 shows an example of a dashboard, based on the open-source CTFd platform. Upon solving a challenge, the participants receive a flag. This flag consists of a random-like string that can be redeemed for points in the dashboard. The reward on the amount of points is related to the difficulty level of the challenge. The countdown component consists of a timer that, when expired,

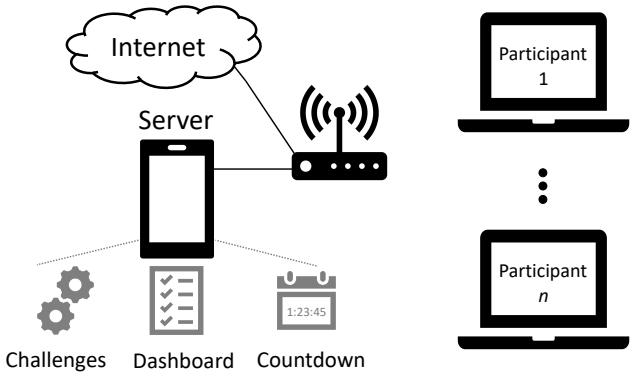


**Fig. 2.** Architecture of CyberSecurity Challenges infrastructure

automatically locks the dashboard preventing further submission of flags. The countdown timer is also used to incentivize the competitiveness of the players on solving the challenges. Coaches aid every team and every participant during the gameplay, such that no one gets stuck or lost while solving the exercises. The coaches also supervise the gameplay to ensure that the desired game objectives, e.g. in terms of learning goals, are achieved. At the end the team with the highest amount of points wins the challenge. Nevertheless, all teams and players are winners since, by participating in the game, the awareness on the topic of secure coding of every participant is stimulated. The competitive nature of the game increases the fun, contributes to the overall awareness level of every player and ensures a memorable event that can have long-lasting impressions.

The different CSC challenges can be implemented in two different ways: 1) using open-source components or 2) using self-developed components. In the first case, the challenges are implemented through adaptation, re-use and re-purpose of existing open-source projects and components. The main advantage of this method is the reduced cost of implementation of individual challenges, while outsourcing their maintenance. In the second case, the challenges can be better adapted to company internal policies while also focusing more on the defensive perspective. The architecture shown in Figure 2 was initially developed for on-site events. A recent installment of the game [15] allows the game not only to be played remotely, but also to include an intelligent coach based on artificial intelligence techniques. In the following we present a more detailed introduction of the CSC game implementation based on open-source components and on the Sifu platform.

### 3.2 CyberSecurity Game

The CSC game was developed in the industry focusing on Web and C/C++ developers. In contrast to C/C++, for the web challenges, it was decided not to focus on a single programming language or framework since many of these programming languages and frameworks are in everyday use in the company

where the CSC game was developed. In this case, we chose a generic approach based on the Open Web Application Security Project - OWASP [27]. The challenges' design took two approaches: 1) based on open-source components and 2) design of own challenges. A common approach to the design of the challenges is given in [19]. Each challenge is presented to the participants according to the following phases: *Phase 1* - introduction, *Phase 2* - challenge, and *Phase 3* - conclusion. Phase 1 presents an introduction to the challenge and sets up the scenario; the main part of the challenge is phase 2; phase 3 concludes the challenge by adding additional text related to secure coding guidelines or additional questions related to phase 2. The types of challenges are: Single-Choice Questions, Multiple-Choice Questions, Text-Entry Questions, Associate-Left-Right, Code-Snippet Challenge, and Code-Entry Challenge.

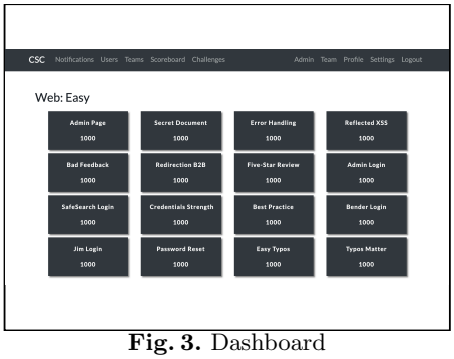


**Fig. 3.** Dashboard

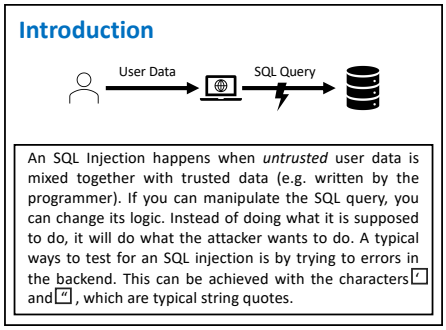


**Fig. 4.** Web Challenge: Phase 1

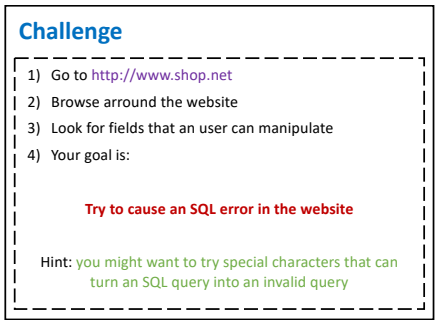


**Fig. 5.** Web Challenge: Phase 2

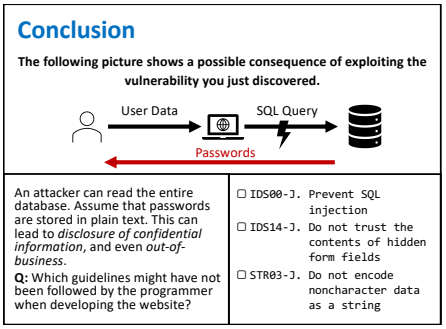


**Fig. 6.** Web Challenge: Phase 3

**Challenges using Open-Source Components** Challenges on secure coding for software developers can be implemented by using and adapting existing open source components. Since most of the available projects focus on the offensive perspective, the following adaptations are suggested: 1) include an incomplete description on how to solve the challenge, and 2) provide follow-up questions

related to secure coding guidelines. Fig. 4-6 shows an example of a challenge for Web developers using OWASP JuiceShop. The challenge's learning goal is to understand what SQL injections are and how to identify an SQL injection quickly. Phase 1 sets the stage for the challenge (Fig. 4). In Phase 2, the player is assisted with how to find the vulnerability, through the textual description, as in Fig 5, or also directed by the game coaches. The last phase consists of an additional question related to the exercise, as shown in Fig 6, which enquires and directs the player to corresponding secure coding guidelines.

Table 1 shows the open-source projects and components which have been used to design CSC challenges for Web and for C/C++, along with the expected effort required to modify them. Note that the design of these challenges is based on open source components that include an offensive perspective. Therefore, after the components' adaptation, it is more natural and more accurate to describe these types of challenges as being *defensive/offensive* (D/O).

**Table 1.** Open-Source Tools used for Cybersecurity Challenges

| Type | Project | Effort | Description |
|---|---|---|---|
| Web/Java | Juice Shop | Minimal | Insecure web application for training purposes from the OWASP project. |
| Web/Java | Java SEI-CERT | Medium | Secure coding guidelines dedicated to Java from Carnegie Mellon University |
| Web | Vulnerable API | Medium | REST API containing several vulnerabilities |
| C/C++ | MBE | Small | Vulnerable code from RPISEC course at Rensselaer Polytechnic Institute |
| C/C++ | C/C++ SEI-CERT | Medium | Secure coding guidelines dedicated to C/C++ from Carnegie Mellon University |
| C/C++ | Vulnerable code snippets | High | Vulnerable C/C++ code from NIST (Juliet Set) |

**Defensive Challenges using Sifu Platform** The Sifu platform hosts code projects containing vulnerabilities in a web application. A web interface is chosen to avoid the players' need to install software on their machines, as this might be difficult or impossible in an industrial setting. The players' task is to fix the project's source code to bring it to an acceptable solution (therefore focusing on the defensive perspective). An acceptable solution is a solution where the source code is compliant to secure coding guidelines and does not have known vulnerabilities. The Sifu platform contains two main components: 1) challenge assessment and 2) an automatic coach. The challenge assessment component analyses the proposed solution submitted by a player and determines if it is acceptable. Analysis is based on several tools, e.g., compiler output, static code analysis, and dynamic code analysis. The automatic coach component is implemented through an artificial intelligence technique that provides hints to the participant when the solution is not acceptable, with the intent to guide the participant to an acceptable solution. Figure 7 shows the web user interface of the Sifu platform. Note that only phase 2 is shown in the figure. The player can browse the different files of the project. All the hints issued by the automatic

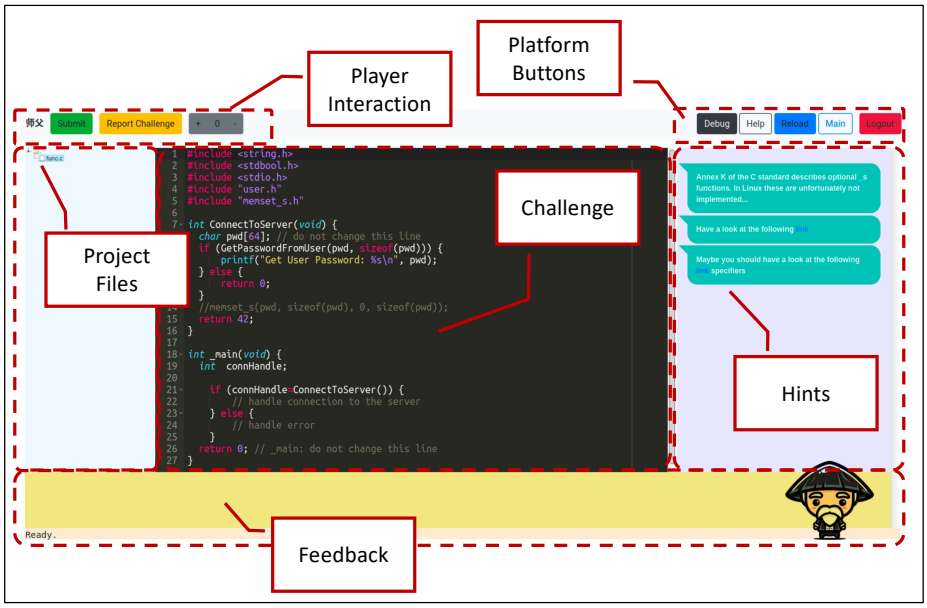


**Fig. 7.** Sifu Platform - User Interface

coach are available on the right-hand side. If the player experiences errors when using the platform, these can be reported for later analysis and improvement. Note that, since untrusted and potentially malicious code will be executed in the platform during the analysis stage, several security mechanisms need to be implemented to guarantee that the players cannot hack it. Further detailed information on the implementation is available in [15, 18]. The open-source Sifu platform can be downloaded from Github [9].

## 4 Evaluation of CyberSecurity Challenges

**Table 2.** CyberSecurity Challenge Events

| **No.** | ***1*** | ***2*** | ***3*** | ***4*** | ***5*** | ***6*** | ***7*** | ***8*** | ***9*** | ***10*** | ***11*** | ***12*** | ***13*** | ***14*** | ***15*** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Type** | D/O | D/O | D/O | D/O | D/O | D/O | D/O | D/O | D/O | D | D | D | D | A | A |
| **Date** | 11/17 | 5/18 | 7/18 | 7/18 | 9/18 | 7/19 | 7/19 | 9/19 | 10/19 | 6/20 | 7/20 | 7/20 | 7/20 | 11/20 | 11/20 |
| **NP** | 11 | 12 | 6 | 30 | 16 | 14 | 15 | 7 | 23 | 15 | 21 | 20 | 15 | 12 | 4 |
| **Where** | DE | DE | DE | DE | DE | CH | CH | DE | TK | OL | OL | OL | OL | OL | OL |

**D/O**: Defensive/Offensive, **D**: Defensive, **NP**: Number of participants, **DE**: Germany, **CH**: China, **TK**: Turkey, **OL**: Online

The authors have implemented the CSC game and have held a total of thirteen CSC events in the industry: nine on-site events (from November 2017 to October 2019) and four CSC online events (from June 2020 to July 2020). Furthermore, two events in November 2020 were held in the academia. Table 2

summarizes all the events. To evaluate and refine the CSC game, we have performed empirical studies together the CSC events. The results presented in this work summarize the overall studies by focusing on the following six dimensions:

- **Know-how** - evaluate if the CSC game contributes to learning new techniques and principles to be used during software development
- **Significance** - evaluate if the CSC game contributes to understanding the importance of secure coding guidelines
- **Skills** - evaluate if the CSC game contributes to improve the participants' secure coding skills
- **Clarity** - evaluate if the challenges in the CSC game are clearly presented
- **Coaching** - evaluate if the help provided by coaches is adequate during gameplay
- **Behavior** - evaluate if the participants, after playing the CSC game, feel prepared to write secure code

The answers to the survey questions were based on a 5-point Likert scale on agreement and are summarized through negative (-) answers (strongly disagree and disagree), neutral (N), and positive (+) answers (agree and strongly agree). Answering the survey was not mandatory and the participants that took part in the study have given their consent; aditionally their answers were anonimized. Although the total number of participants to the CSC events exceeded 200, the total number of participants that answered the survey were: 56 - for defensive/offensive (D/O) events 1-9, 25 - for defensive (D) events 10-13, and 14 - for defensive challenges in the academia (A) in events 14-15. Additional results were captured through open feedback, questions and discussions with the participants. The main positive and negative quotes from the participants were also collected. In the following sub.sections we present a brief overview and discussion of the main results of the survey, participant feedback, and also an overview of the lessons-learned on the design of CSC games and events. For a more in-depth overview of the empirical studies, we refer the reader to the work published by the same authors in [10–20].

### 4.1 Results

Table 3 shows a summary of the results for the different six questions, both for the industry (81 participants) and for the academia (14 participants). The two highest-ranked questions, are the following: Defensive/Offensive Challenges - Q2, Q5; Offensive Challenges - Q2+Q3+Q5, Q1; Offensive Challenges - Q3, Q4+Q5. The results in this table leads to the following conclusions: (1) defensive challenges have a higher level of agreement than defensive/offensive challenges, (2) there is a higher amount of neutral answers in defensive/offensive than in purely defensive challenges, (3) nevertheless both defensive/offensive and defensive challenges show a high level of agreement on the suitability as an method to increase awareness. This means that, while there are good indicators that both challenge types be suitable to raise secure coding awareness on software

**Table 3.** CyberSecurity Challenge - Empirical Results

| Question | Industry | | | | | | Academia | | | Description |
|---|---|---|---|---|---|---|---|---|---|---|
| | D/O | | | D | | | | | | |
| | - | N | + | - | N | + | - | N | + | |
| *Q1* | 12.5 | 7.1 | 80.4 | 0.0 | 10.0 | 90.0 | 6.2 | 12.5 | 81.3 | I learned new techniques and principles of secure software development |
| *Q2* | 0.0 | 5.3 | 94.7 | 0.0 | 0.0 | 100.0 | 18.7 | 12.5 | 68.8 | I understand the importance of secure coding guidelines |
| *Q3* | 3.6 | 14.3 | 82.1 | 0.0 | 0.0 | 100.0 | 0.0 | 6.2 | 93.8 | Focusing on the challenges improves my practical secure coding skills |
| *Q4* | 8.9 | 8.9 | 82.2 | 8.0 | 8.0 | 84.0 | 0.0 | 12.5 | 87.5 | The learning goals of the challenges were clearly explained |
| *Q5* | 1.8 | 12.5 | 85.7 | 0.0 | 0.0 | 100.0 | 12.5 | 0.0 | 87.5 | The help from the coaches was adequate |
| *Q6* | 8.9 | 26.8 | 64.3 | 0.0 | 20.0 | 80.0 | | | | I feel that I am prepared to handle issues related to secure coding at work |

**-**: Negative agreement, **N**: Neutral answers, **+**: Positive agreement
**D/O**: Defensive/Offensive, **D**: Defensive

developers, the indicators for defensive challenges show a higher adequacy. The presented results also show good results for the three awareness constructs as introduced by Hänsch et al [22] - perception (Q2), protection (Q1), and behavior (Q3). In an extended study, using the same artifact in the academia, shows also good indicators of its suitability as a means to train future generation of junior industrial software developers, while still in an academic setting. For a more in-depth discussion on the presented results, we refer the reader to the literature by the same authors [10–20].

### 4.2 Participant Feedback

**Table 4.** Quotes from CSC Participants

| | Quotes from Participants |
|---|---|
| Positive | I really enjoyed participating in the challenges. |
| | I am well excited in trying to crack the answers to the challenges |
| | Enjoyed the challenges, different topics and how competitive we became |
| | It was lots of fun. Questions inbetween were nice. |
| | Enjoyed and lots of fun. I've learned many interesting things |
| | Quite fun and nice to work, especially work in team |
| | Enjoyed and learned very much |
| | It was really funny and I leaned a lot |
| | Funny and interesting; learned a lot - hope to remember and use in practice |
| | Really liked and enjoyed the exercises |
| | Enjoyable to try everything and very fun |
| Negative | Hints not always accurate or precisely leading to the problem in the code |
| | We do not perform attacks on systems |
| | Could not understand what to do in the challenge |
| | Some hints are very generic |
| | The user interface is very minimalist |
| | User interface could be improved |

Table 4 shows the main positive and negative quotes from participants to the CSC games. Most of the collected feedback was positive and gives a good indicator that the CSC game is suitable for raising secure coding awareness. The feedback obtained by the authors, during all the events that took place in the industry, has also shown that the software developers highly appreciate playing the CSC game. For one of the groups that participated in the CSC event, the players have joined force after the event, and searched the internet for further similar games, thus giving a good indicator of possible long-term effects. Another success factor was the positive feedback from management, that lead to recurring CSC events and the establishment of a good impression managers. Nevertheless we collected some negative feedback related with the user interface, and the precision of the hints. Additional negative feedback is related with the fact that defensive/offensive challenges still include an offensive part and that this can lead to difficulty in understanding on what to do in the challenge. In a separate discussion, we could conclude that the help from coaches can improve the game experience positively.

### 4.3 Evaluation of the Design

Figure 8 shows an overview of the lessons learned on the different aspects related with the design, deployment and refinement of CyberSecurity Challenges. These have resulted from all the thirteen deployments that were performed in the industry. The five top-level design aspects are: 1 - learning goals, 2 - time management, 3 - game roles, 4 - game components, and 5 - challenges. Learning goals (L) is related with the content of the game and its adaptation to the target group of software developers and considers factors such as programming language, secure coding guidelines, alignment with management, and current status quo of know-how. Time management is an important aspect for games in the industry. This aspect includes the agenda of the event and the temporal dimensioning of the challenges. Clear definition of roles in a serious game is also a critical aspect of the design of such a game. The CyberSecurity Challenges game defines three roles: individual player, team, and coach. Since these games are deployed in a computer network, the different components in present in the network and their management is also an important aspect of the game. Finally, the aspect challenges (CH) looks at the different categories of challenges (as introduced before), challenge types suitable for the industry, the different phases of a challenge and tools to create the challenges. Detailed discussions on each of these aspects can be found in [10–20].

## 5 Conclusions

If not addressed appropriately, software vulnerabilities can result in serious negative consequences. A good time to address these issues is in the early stages of software development by raising the awareness of software developers on the topic of secure coding. This paper presents CyberSecurity Challenges (CSC) as

a possible solution. CyberSecurity Challenges is a genre of serious games developed with the purpose to raise awareness of industrial software developers on the topic of secure coding and secure coding guidelines. CSC games have been developed since 2017 in the industry, and have been extensively studied as part of the PhD research by the first author, resulting in more than ten publications. The CSC game can be used both for onsite training and for remote training, thus easily adapting to possible travel restrictions as imposed by the current COVID-19 situation.

Our results through empirical studies show that this game is adequate as a method to raise secure coding awareness, both when using defensive/offensive challenges and purely defensive challenges. Furthermore, preliminary results indicate that the same artifact could be used in the academia to prepare the future industry workforce. Feedback obtained from software developers in the industry also indicates that this game is well accepted and welcome by this community. During gameplay, not only do software developers have fun but also practice the usage secure coding guidelines for secure software development. Although the authors did not conduct a long-term study on the effects of playing such game, positive results are expected according to well established policy compliance theories. Furthermore, CSC games found not only success in the software development community, but was also well accepted by management. Therefore, we think that this type of game is a viable approach to tackle possible software vulnerabilities due to bad code quality in terms of security.

## Acknowledgements

The authors would like to thank the participants of the CyberSecurity Challenges for their time and their valuable answers and comments. Also, the authors would also like to thank Kristian Beckers and Thomas Diefenbach for their helpful, insightful, and constructive comments and discussions. This work is financed by national funds through FCT - Fundação para a Ciência e Tecnologia, I.P., under the projects FCT UIDB/04466/2020 and UIDP/04466/2020. Furthermore, the third author thanks the Instituto Universitário de Lisboa and ISTAR, for their support.

## References

1. Assal, H., Chiasson, S.: 'Think secure from the beginning' A Survey with Software Developers. In: Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. pp. 1–13. CHI '19, Association for Computing Machinery, New York, NY, USA (2019)
2. Beckers, K., Pape, S.: A Serious Game for Eliciting Social Engineering Security Requirements. In: 2016 IEEE 24th International Requirements Engineering Conference (RE). IEEE (08 2016)
3. Bundesamt für Sicherheit in der Informationstechnik: Baustein B 5.21 - Webanwendungen (2014), https://tinyurl.com/y25m2kxl

4. Bundesamt für Sicherheit in der Informationstechnik: BSI IT-Grundschutz-Katalog, 2016, 15. ed. (2016), https://download.gsb.bund.de/BSI/ITGSK/IT-Grundschutz-Kataloge_2016_EL15_DE.pdf
5. Carnegie Mellon University: Secure Coding Standards (2019), https://tinyurl.com/y29mwsyj, online
6. Davis, A., Leek, T., Zhivich, M., Gwinnup, K., Leonard, W.: The fun and future of CTF. 2014 USENIX Summit on Gaming, Games, and Gamification in Security Education (3GSE 14) pp. 1–9 (2014), https://www.usenix.org/conference/3gse14/summit-program/presentation/davis
7. ENISA: The cost of incidents affecting CIIs (8 2016), https://tinyurl.com/y3v4rv8x
8. Frey, S., Rashid, A., Anthonysamy, P., Pinto-Albuquerque, M., Naqvi, S.A.: The Good, the Bad and the Ugly: A Study of Security Decisions in a Cyber-Physical Systems Game. IEEE Transactions on Software Engineering **45**(5), 521–536 (2019)
9. Gasiba, T.: Sifu Platform (12 2020), https://github.com/saucec0de/sifu, Siemens AG, MIT License
10. Gasiba, T., Beckers, K., Suppan, S., Rezabek, F.: On the Requirements for Serious Games Geared Towards Software Developers in the Industry. In: Damian, D.E., Perini, A., Lee, S. (eds.) Conference on Requirements Engineering Conference. pp. 286–296. IEEE, Jeju, South Korea (09 2019). https://doi.org/10.1109/re.2019.00038
11. Gasiba, T., Hodzic, S., Lechner, U., Pinto-Albuquerque, M.: Raising Security Awareness using CybersecurityChallenges in Embedded Programming Courses. In: forthcoming (2021), in preparation
12. Gasiba, T., Lechner, U.: Raising secure coding awareness for software developers in the industry. In: 2019 IEEE 27th International Requirements Engineering Conference Workshops (REW). pp. 141–143. IEEE, Jeju, South Korea (09 2019). https://doi.org/10.1109/REW.2019.00030
13. Gasiba, T., Lechner, U., Cuellar, J., Zouitni, A.: Ranking Secure Coding Guidelines for Software Developer Awareness Training in the Industry. In: Queirós, R., Portela, F., Pinto, M., Simões, A. (eds.) First International Computer Programming Education Conference (ICPEC 2020). OpenAccess Series in Informatics (OASIcs), vol. 81, pp. 11:1–11:11. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2020)
14. Gasiba, T., Lechner, U., Pinto-Albuquerque, M.: Awareness of Secure Coding Guidelines in the Industry - A first data analysis. In: The 19th IEEE International Conference on Trust, Security and Privacy in Computing and Communications. IEEE, Online (12 2020), to appear
15. Gasiba, T., Lechner, U., Pinto-Albuquerque, M.: Sifu - A CyberSecurity Awareness Platform with Challenge Assessment and Intelligent Coach. In: Journal, Special Issue on Cyber-Physical System Security. SpringerOpen (12 2020). https://doi.org/10.1186/s42400-020-00064-4
16. Gasiba, T., Lechner, U., Pinto-Albuquerque, M.: CyberSecurity Challenges for Software Developer Awareness Training in Industrial Environments. In: 16th International Conference on Wirtschaftsinformatik (2021), to appear
17. Gasiba, T., Lechner, U., Pinto-Albuquerque, M.: Is Secure Coding Education in the Industry Needed? An Investigation Through a Large Scale Survey. In: 43rd International Conference on Software Engineering (2021), to appear
18. Gasiba, T., Lechner, U., Pinto-Albuquerque, M., Porwal, A.: Cybersecurity Awareness Platform with Virtual Coach and Automated Challenge Assessment. In:

6th Workshop On The Security Of Industrial Control Systems & Of Cyber-Physical Systems (CyberICPS). pp. 67–83. Springer, Cham, Online (12 2020). https://doi.org/978-3-030-64330-0_5
19. Gasiba, T., Lechner, U., Pinto-Albuquerque, M., Zouitni, A.: Design of Secure Coding Challenges for Cybersecurity Education in the Industry. In: 13th International Conference on the Quality of Information and Communications Technology. pp. 223–237. Springer, Online (09 2020). https://doi.org/978-3-030-58793-2_18
20. Gasiba, T., Lechner, U., Rezabek, F., Pinto-Albuquerque, M.: Cybersecurity Games for Secure Programming Education in the Industry: Gameplay Analysis. In: Queirós, R., Portela, F., Pinto, M., Simões, A. (eds.) First International Computer Programming Education Conference (ICPEC 2020). OpenAccess Series in Informatics (OASIcs), vol. 81, pp. 10:1–10:11. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany (2020)
21. Graziotin, D., Fagerholm, F., Wang, X., Abrahamsson, P.: What happens when software developers are (un)happy. Journal of Systems and Software **140**, 32–47 (2018)
22. Hänsch, N., Benenson, Z.: Specifying IT security awareness. In: 25th International Workshop on Database and Expert Systems Applications, Munich, Germany. pp. 326–330. IEEE, Munich, Germany (Sep 2014). https://doi.org/10.1109/DEXA.2014.71
23. IEC 62443-4-1: Security for industrial automation and control systems - part 4-1: Secure product development lifecycle requirements. Standard, International Electrotechnical Commission (01 2018)
24. ISO 27001: Information technology – Security techniques – Information security management systems – Requirements. Standard, International Standard Organization, Geneva, CH (10 2013)
25. McIlwraith, A.: Information Security and Employee Behaviour: How to Reduce Risk Through Employee Education, Training and Awareness. Gower Publishing, Ltd. (2006)
26. Moody, G.D., Siponen, M., Pahnila, S.: Toward a Unified Model of Information Security Policy Compliance. MIS quarterly **42**(1), 1–50 (2018)
27. OWASP Foundation: Open Web Application Security Project, https://owasp.org/
28. Patel, S.: 2019 Global Developer Report: DevSecOps finds security roadblocks divide teams (July 2020), https://about.gitlab.com/blog/2019/07/15/global-developer-report/, [Online; posted on July 15, 2019]
29. Rieb, A.: IT-Security Awareness mit Operation Digitales Chamäleon. Ph.D. thesis, Universität der Bundeswehr München, Neubiberg (2018)
30. Schneier, B.: Software Developers and Security. Online (July 2020), https://www.schneier.com/blog/archives/2019/07/software_develo.html
31. Stewart, G., Lacey, D.: Death by a Thousand Facts: Criticizing the Technocratic Approach to Information Security Awareness. Information Management & Computer Security **20**(1), 29–38 (2012)
32. Tahaei, M., Vaniea, K.: A Survey on Developer-Centred Security. In: 2019 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). pp. 129–138. IEEE (2019)
33. Xie, J., Lipford, H.R., Chu, B.: Why do Programmers Make Security Errors? 2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC) pp. 161–164 (09 2011). https://doi.org/10.1109/VLHCC.2011.6070393

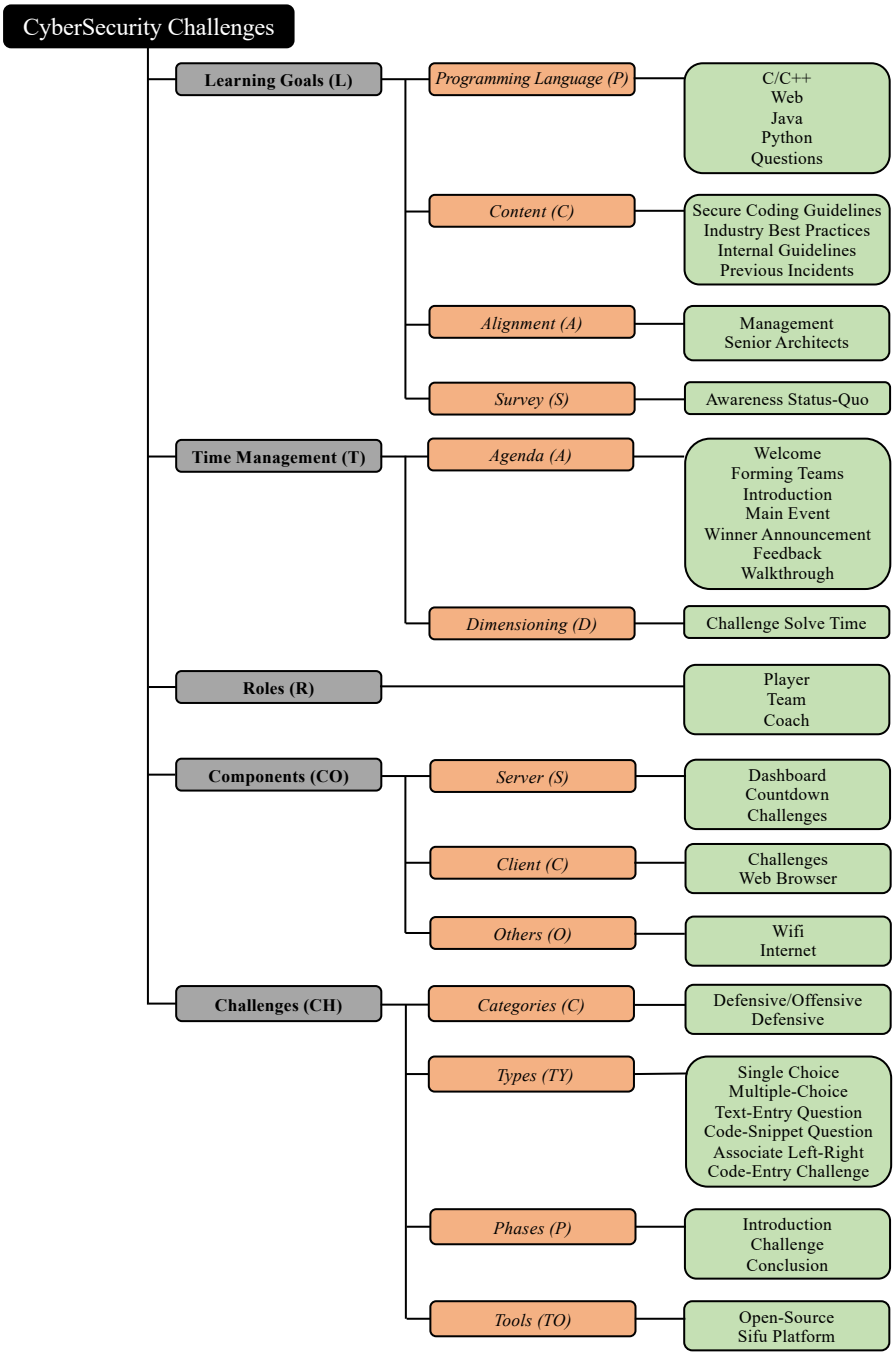


**Fig. 8.** CyberSecurity Challenges