SSETGami: Secure Software Education Through Gamification

Hector Suarez
University of Tennessee at Chattanooga, ffq369@mocs.utc.edu

Hooper Kincannon
University of Tennessee at Chattanooga, rrt654@mocs.utc.edu

Li Yang
University of Tennessee at Chattanooga, li-yang@utc.edu

Abstract
Since web browsers have become essential to accomplishing everyday tasks, developing secure web applications has become a priority in order to protect user data, corporate databases and critical infrastructure against cyber-crimes . This research presents a game-like (gamification) approach to teach key concepts and skills on how to develop secure web applications. Gamification draws on motivational models, one of psychological theories. Gamification design has great potential over traditional education where we often find students demotivated and lecturers failing to engage them in learning activities. This research created game-like learning modules to teach top vulnerabilities and countermeasures for these top vulnerabilities in secure web developments including SQL injection, broken authentication and session management, cross site scripting, insecure direct object references, etc. In this paper, each module is self-contained with a module background, sample module questions, and the expected learning outcomes of each module.

Disciplines
Curriculum and Instruction | Engineering Education | Information Security | Management Information Systems | Technology and Innovation

Comments
This material is based upon work supported by the National Science Foundation under Grant No. 1623624 and 1663105. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

1 INTRODUCTION

With the increasing use of web browsers and the rise of The Internet of Things (IoT), developing secure software has become essential in protecting users on the web. Web browsers help users interact with a plethora of web based services. Some of these services include databases and management systems that handle sensitive information, which the integrity of the data and systems can be vulnerable to cyber attacks. A business can no longer risk product delivery turnaround time in exchange for a less secure software product as they risk exposing themselves to corporate espionage, data loss and lawsuits. Developing secure software needs to be considered a requirement and needs to be placed on the same priority level as other system software requirements. Experts claim that the security of browser-based applications is considered less important than speed, functionality and overall experience during developments (Sargent, 2012). Vulnerabilities in HTML 5 make it an emerging threat while SQL injection and XSS remain among the top attacks (OWASP, 2013). Fortunately, researchers have developed a deep understanding of web/browser application threats, and have designed counter measures to mitigate threats, which naturally can become excellent resources to develop valuable education materials. In a study done on ten computer security students it was found that most of them did not do any of the weekly reading assignments and those that did only read about ten minuets(Schreuders & Butterfield, 2016). Therefore, it is imperative to change traditional teaching methods and find ways to engage students in game-like self-paced educational environments that will prepare the future workforce with expertise and skills on web/browser security. This paper presents an interactive game-like approach to Secure Software Education Through Gamification called called SSETGami. Gamification is defined as the application of game mechanisms to non-game contexts and is becoming widely used across a range of domains, including within higher education, to increase motivation and engagement (Deterding, Dixon, Khaled, & Nacke, 2011).

The remainder of this paper is structured as follows. Section 2 discusses related work as well as this paper's contribution. Section 3 discusses this research's impact to cyber security education and presents the ten SSETGami modules, section 4 concludes this paper and future work is also discussed.

2 Related Works

Gamification research is relatively a new topic and is used in online marketing, education (Zichermann & Cunningham, 2011). Gamification typically involves applying game mechanics such as presenting tasks as quests to be completed, reward-

ing completion of quests in the form of experience points (XP), and providing a clear path to progression, often in the form of "leveling up" through player levels. Other common aspects include rewarding certain achievements with virtual badges, and leader boards, which can foster competition between users and give an indication of how their progress compares to that of others. Gamification has previously been applied to increase engagement and enjoyment in security education and training (Schreuders & Butterfield, 2016). Capture the Flag (CTF) events are popular amongst security enthusiasts and prevalent at conferences, such as at the annual DEFCON conference (DEF CON Communications, 2013), the online CTF365 platform (CTF365, 2013). These competitions often gamify security tasks by assigning points to defensive and offensive tasks. Researchers have also applied gamification principles to the usability of CAPTCHAs (via quizzes on altered animations (Kani & Nishigaki, 2013), and to evaluate the effectiveness of existing CAPTCHA systems (Saha, Manna, & Geetha, 2012)), for encouraging the use of strong passwords (with competitive avatar development (Kroeze & Olivier, 2012)) and have considered uses of gamification in security training (Amorim, Hendrix, Andler, & Gustavsson, 2013). In (Yuan, Yang, Jones, Yu, & Chu, 2016) the authors aggregate and present the relationship among secure software engineering curricula from the Department of Homeland Security (DHS),Carnegie Mellon University (CMU) and ACM/IEEE 2013. This paper will extend the web security component of the latter mentioned article by creating the web security component by means of gamification.

3 GAMIFICATION MODULES: ASSESSMENT AND EDUCATION SIGNIFICANCE

The ten modules presented in this section are designed to be self paced and are focused on the following web security topics: SQL injection, broken authentication and session management, cross site scripting, insecure direct object references, cross site request forgery, missing function level access, security misconfiguration, sensitive data exposure, unvalidated redirects and forwards, and using components with known vulnerabilities. The first six mentioned topics will be discussed in detail and the rest of the four topics will be briefly described.

The OWASP top ten has been recommended in the list of secure coding guides (MITRE, 2017a), therefore it was used as a main source to validate the gamification assessment questions, due to its credibility and wide industry acceptance. Other sources such as the community developed Common Weakness Enumeration (CWE) list (MITRE, 2017a) are also referred.

Many of the assessment questions were derived directly from content presented in OWASP top 10 descriptions and practice exams presented on the site. The answers to these assessment questions were validated in two steps. We accept that sources provided from OWASP project as source of truth. The assessment questions and answers were further reviewed by one of the authors, who currently works as a security developer at UNUM, an insurance company with largest disability insurance share in the world. He is currently in charge of secure software development and training. Additionally, the ability of this work on improving quality of education content in areas of security software development is scalable. The work is designed in a way that the pool of gamification assessment questions can be easily expanded. For example, we will develop a linker which can dynamically load questions from a pool into the gamified learning solutions.

Gamification, which includes points, levels, and badges, has been effective in motivating and engaging students in education to improve learning outcomes (Schreuders & Butterfield, 2016). Students gain capability in identifying and fixing common vulnerabilities in web-based applications. Gamification can also be easily linked with hands-on exercises, for example, the levels indicate the scope of students' knowledge in secure software developments; the point system tracks the progress and skill of the student, the interaction between student and software, and the status of the player within one level; the badges provides a token of achievement towards the goal indicating and encouraging progress. The contribution of this work lies in using gamification to improve student learning on secure web development. Our approach can be widely adopted in courses such as software development, computer security, information technology security, secure software development, fundamentals of cybersecurity, etc. Our topics can also be mapped to Information Security Assurance (ISA) and Knowledge Areas (KAs) in the Computer Science Curricula 2013 (ACM, 2013) as shown in Table 1. This offers transferability of our teaching materials, and can internationally impact a large number of universities and colleges.

Figure 1 shows the UI of SSETGami, a display pop up window shows up after each answered is submitted, this way the student doesn't have to wait until they respond to all module questions to know if each answer is correct. Upon completing a module, the student receives a badge, see figure 3, this meant to encourage and keep the student engaged in completing more modules. All of the assessment questions were derived from OWASP top 10 threats and mitigation exam (OWASP, 2011) and the correct answers are italicized.

Topics in secure web development	ISA KAs from CS curricula 2013
SQL Injection	Fundamental Concepts
Unvalidated redirects and forwards, missing function level access	Security policy and governance
Cross site scripting, Cross site request forgery, SQL injection	Risk Management
Using components with known vulnerabilities	Secure coding and software engineering
Broken authentication and session management	Network security, Cryptography
Sensitive data exposure	Cryptography
Insecure direct object references, missing function level access, security misconfiguration	Security architecture and systems administration

Table 1: Mapping teaching topics on secure web development to ISA KAs in CS curricula 2013.

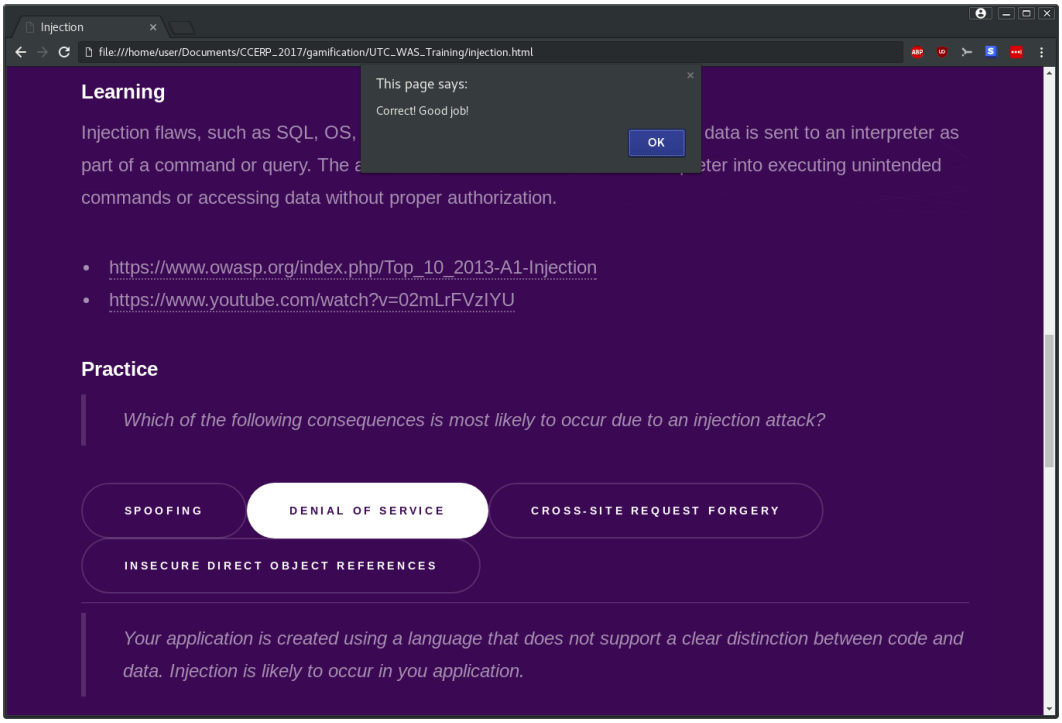

Figure 1: SQL Injection module.

In order to make the modules be self paced, each module consists of a short ten minute video and a reference table containing fundamental module information see figure 2.

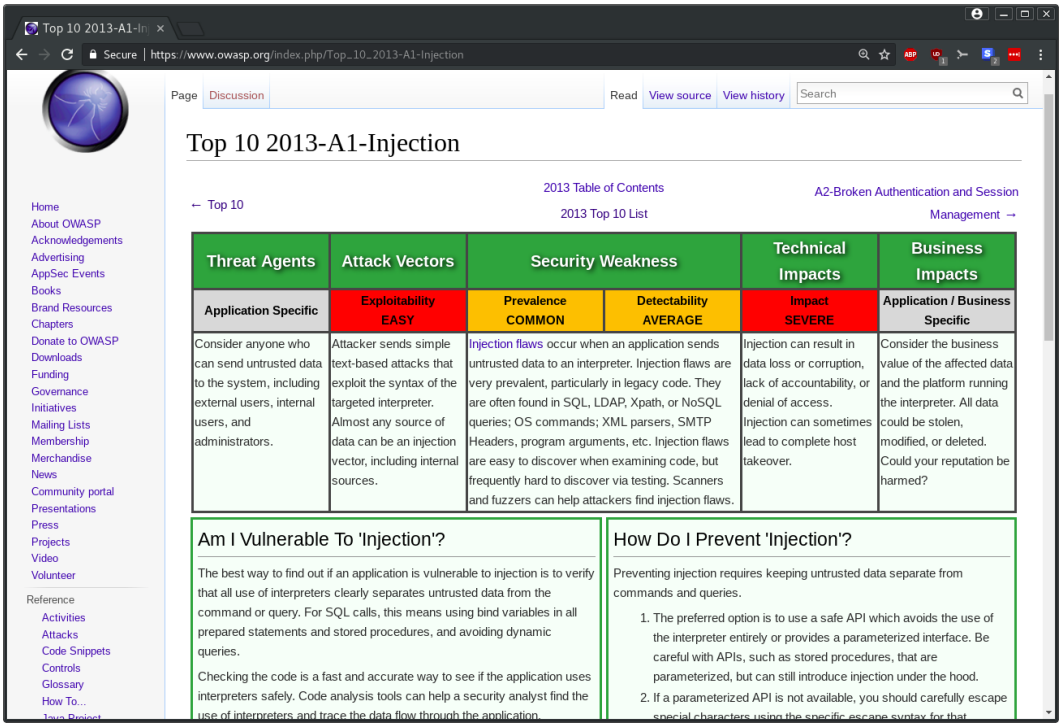

Figure 2: OWASP quick reference to Injection flaws (OWASP, 2013).

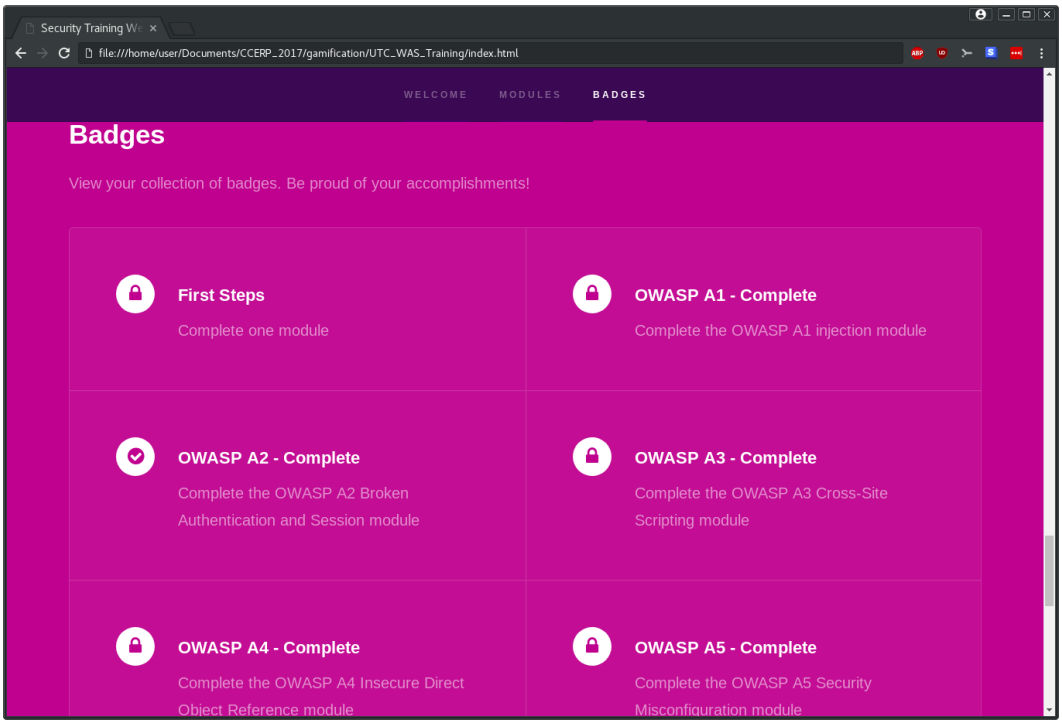

Figure 3: Badges reward the student for their accomplished modules.

3.1 SQL Injection

In web-based applications, an input from a user is interpreted by a web server and then executed by a database server. SQL injection is a technique where malicious users can inject SQL commands into an SQL statement, via web page input. Injected SQL commands can alter SQL statement and compromise the security of a web application. Through SQL Injection, an attacker can bypass authentication checks, making unauthorized changes, or learning table schemes, table names. Strategies used to mitigate SQL Injection attacks include whitelist, blacklist, prepared statement, stored procedures, and escaping user input. (OWASP, 2015).

Sample Assessment Questions

Which of the following consequences is most likely to occur due to an injection attack?

(a) Spoofing
(b) Cross-site request forgery
(c) *Denial of Service*

(d) Insecure direct object references

Which of the following scenarios is most likely to cause an injection attack?

(a) Unvalidated input can be distinguished from valid instructions
(b) *Unvalidated input is embedded in an instruction stream*
(c) A Web action performs an operation on behalf of the user without checking a shared secret
(d) A Web application does not validate a client's access to a resource

Expected Learning Outcomes

By the end of the SQL Injection the student should know how to test for injection vulnerabilities and know how to prevent injection attacks by using the principle of least privilege when creating user accounts on a database.

3.2 Broken Authentication and Session Management

Session management tracks a user's activity across sessions of interaction with a website (Visaggio, 2010). Application functions related to authentication and session management are often not implemented correctly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities.

Sample Assessment Questions

Which of the following scenarios are most likely to result in broken authentication and session management vulnerabilities?

(a) *Unused and unnecessary services, code, and DLLs are disabled*
(b) Poorly implemented custom code is used
(c) Misconfigured off-the-shelf code is used

Which of the following functionalities should you include in an authentication and session management system?

(a) Forwarding system functionality
(b) *Inactivity timeout functionality*
(c) Escaping functionality

Expected Learning Outcomes

The student should have knowledge of what environments are affected, know how to test for session vulnerabilities and be able to prevent session mismanagements by use of a well established authentication and session framework.

3.3 Cross Site Scripting (XSS)

XSS vulnerability makes it possible for attackers to inject malicious code (e.g. JavaScript programs) into the victim's web browser. Using this malicious code, the attackers can hijack the victim's credentials, such as cookies, deface web sites, or redirect the user to malicious sites. An attacker can make a user to unknowingly execute commands from a third-party through a vulnerable website that the user trusts. The access control policies (i.e., the same origin policy) employed by the browser to protect those credentials can be bypassed by exploiting the XSS vulnerability. Vulnerabilities of this kind can potentially lead to large-scale attacks. The XSS usually involves three players: a victim, a vulnerable server, and an attacker.

Sample Assessment Questions

Choose the input sources that can be directly manipulated by a malicious attacker?

(a) SSL Handshake
(b) OPTIONS
(c) Server configuration files
(d) *GET/POST*

When a malicious user convinces a victim to send a request to a server with malicious input and the server echoes the input back to client, what type of XSS attack has occurred?

(a) Failure to restrict URL access
(b) *Reflected XSS*
(c) Insecure direct objecet references
(d) Persistent XSS

Expected Learning Outcomes

Students will know how to test and identify the different types of XSS attacks on the client and server and know that existing auto-sanitization libraries are a good option to prevent this attack.

3.4 Insecure Direct Object References

A direct object reference occurs when a developer exposes a reference to an internal implementation object, such as a file, directory, or database key. Without an access control check or other authorized protection, attackers can manipulate these references to access unauthorized data. An example of this weakness can occur if a system used sequential or other easy to guess session ids that would allow a user to easily switch to another user's session and read or modify their data (MITRE, 2017b).

Sample Assessment Questions

Which is most susceptible to an insecure direct object reference attack?

(a) Conditional constructs
(b) *Registry keys*
(c) Nonpersistent cookies
(d) DELETE parameters

Which of the following vulnerabilities are most likely to occur due to an insecure direct object reference attack?

(a) Impersonating any user on the system
(b) *Accessing a resource without authorization*
(c) Executing commands on the server.
(d) Modifying SQL data pointed to by the query.

Expected Learning Outcomes

By the end of this module students should be able to quickly analyze code and identify reference authorization. They should also know how to protect each user accessible object by checking the object's permissions(OWASP, 2015). Students should also know how to design a secure database and that hashes key values in a database so that the record's integrity can be verified (MITRE, 2017b).

3.5 Cross Site Request Forgery (CSRF)

Cross-Site Request Forgery (CSRF) is an attack that forces an end user to execute unwanted actions on a web application in which they're currently authenticated (OWASP, 2015). This allows an attacker to access functionality in a target web application via the victim's already authenticated browser. Targets include web applications such as social media, in browser email clients, online banking, and

web interfaces for network devices. Using social engineering such as sending a link via email or chat, the attacker may trick victim users of a web application to execute actions of the attacker's choosing. The malicious requests are routed to the target site via the victim's browser, which is authenticated against the target site. The vulnerability lies in the affected web application. In the case of normal user, a successful CSRF exploit can compromise the end user's data and operation. If the targeted end user is the administrator, this can compromise the entire web application.

Sample Assessment Questions

Which threat is most likely to occur when a POST parameter performs an operation on behalf of a user without checking a shared secret?

(a) Cross-site scripting
(b) Injection
(c) Insecure direct object reference
(d) *Cross-site request forgery*

What is the most common result of a cross-site request forgery?

(a) Enabling of IPSec
(b) *Elevation of privilege*
(c) Misconfigured security features
(d) Disabled security features

Expected Learning Outcomes

For this topic, students will need to have a good understanding of their web architecture because protecting against CSRF attacks involves deep knowledge implementing cookies with the same origin policies. Students should be able to review code and identify CSRF vulnerabilities and know which frameworks have built in CRSRF built in support. XSS can bypass CSRF defenses, so students should make sure that their designed applications are not vulnerable to XSS (MITRE, 2017c).

3.6 Missing Function Level Access

Most web applications verify function level access rights before making that functionality visible in the UI. However, applications need to perform the same access control checks on the server when each function is accessed. If requests are not verified, attackers will be able to forge requests in order to access functionality without proper authorization or perform unauthorized functionality like creating an account.

Sample Assessment Questions

Which of the following depict the typical impact of failure to restrict URL access?

(a) Attackers perform all actions that the victims themselves have permission to perform
(b) Attackers perform man-in-the-middle attacks
(c) Attackers impersonate any user on the system
(d) *Attackers invoke functions and services they have no authorization for*

Which two protocols than can be used to protect the connections between the physical tiers of your application?

(a) Kerberos
(b) HTTP
(c) *SSL and IpSec*
(d) FTP

Expected Learning Outcomes

Students should be able to test and verify each function level access an be able to determine if the access role is correct. For preventing this type of attack, students should have an idea of how to encapsulate the authorized user role modules in their web application. Students should also know how to mitigate this type of attacks by using mature libraries and frameworks such as OpenSSL or ESAPI Authenticator (MITRE, 2017d).

3.7 Security Misconfiguration

Good security requires having a secure configuration defined and deployed for the application, frameworks, application server, web server, database server, and platform. Secure settings should be defined, implemented, and maintained, as defaults are often insecure. Additionally, software should be kept up to date to avoid critical vulnerabilities such as *zero day* exploits or remote code execution exploits.

3.8 Sensitive Data Exposure

When sensitive data are not handled correctly such as when transmitting data or when it is at rest in a database. Ways to prevent sensitive data exposure are to encrypt data at rest and during transmission, authenticate users to system resources. Data integrity is important because it can involve a user's protected health care

information, credit card information and other personal information that must be protected. Protected data are targeted by cyber criminals as personal data can be illegally sold for a profit. Corporations are also vulnerable to corporate espionage, so the communication on a corporate network should also be encrypted such as email and other web services that the company and its customers use.

3.9 Unvalidated Redirects and Forwards

This type of attack is also known as an *open redirect*. Web applications frequently redirect and forward users to other pages and websites, and use untrusted data to determine the destination pages. Without proper validation, attackers can redirect victims to phishing or malware sites, or use forwards to access unauthorized pages. Mitigating such an attack can be achieved by creating a white list of allowed URLs or domains to be used for redirection (MITRE, 2017e).

3.10 Using Components With Known Vulnerabilities

Components, such as libraries, frameworks, and other software modules, almost always run with full privileges. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications using components with known vulnerabilities may undermine application defenses and enable a range of possible attacks and impacts. Mitigating such attacks can be done by keeping up to date on new framework/library features and documenting their versions.

4 CONCLUSION & FUTURE WORK

Gamification is relatively a new approach to assisting educators in higher education as well as in the corporate setting. Providing students more opportunities to learn on the go by using their mobile devices, was considered when developing SSETGami by developing it in HTML5, meaning that the code base can be scaled to be used in an online course. This way the students can always have the latest course material and practice it on their mobile devices. Each module content can easily be updated with new questions or video lectures and can be adopted in many other courses such as network security, computer security and other cyber security subjects.

Traditional teaching methods such as white board lectures may not be the most effective in motivating the younger generation of computer science students to learn web security. This has motivated our research in incorporating technology into teaching methods. This research focused on teaching web software security by creating SSETGami a gamification web based platform that engages the student by

allowing them to view their progress in real time and rewarding them with module badges and points. Future improvements to SSETGami would be to add social interaction features, offer analytics to the end user (García, Pedreira, Piattini, Cerdeira-Pena, & Penabad, 2017), and integrate it into an institution's on-line grading system. Another area to study is to gather metrics and evaluate the effectiveness of gamified learning and improve these types of interactive game like teaching methods.

ACKNOWLEDGEMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 1623624 and 1663105. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

REFERENCES

Sargent, M. (2012). Little being done to prevent web application threats, analysts say. Retrieved from http://searchsecurity.techtarget.com/news/2240163546/Little-being-done-to-prevent-Web-application-threats-analysts-say

OWASP. (2013). Top 10 2013. Retrieved from https://www.owasp.org/index.php/Top_10_2013

Schreuders, Z. C. & Butterfield, E. (2016). Gamification for teaching and learning computer security in higher education. In *2016 usenix workshop on advances in security education (ase 16)*. USENIX Association.

Deterding, S., Dixon, D., Khaled, R., & Nacke, L. (Eds.). (2011). From game design elements to gamefulness: Defining gamification, 9–15.

Zichermann, G. & Cunningham, C. (2011). *Gamification by design: Implementing game mechanics in web and mobile apps*. " O'Reilly Media, Inc."

DEF CON Communications, I. (2013). Def con hacking conference capture the flag archive. Retrieved from https://www.defcon.org/html/links/dcctf.html

CTF365. (2013). Ctf365. Retrieved from https://ctf365.com/

Kani, J. & Nishigaki, M. (2013). Gamified captcha. In *International conference on human aspects of information security, privacy, and trust* (pp. 39–48). Springer.

Saha, R., Manna, R., & Geetha, G. (2012). Captchino-a gamification of image-based captchas to evaluate usability issues. In *Computing sciences (iccs), 2012 international conference on* (pp. 95–99). IEEE.

Kroeze, C. & Olivier, M. S. (2012). Gamifying authentication. In *Information security for south africa (issa), 2012* (pp. 1–8). IEEE.

Amorim, J. A., Hendrix, M., Andler, S. F., & Gustavsson, P. M. (2013). Gamified training for cyber defence: Methods and automated tools for situation and threat assessment. In *Nato modelling and simulation group (msg) annual conference 2013 (msg-111), 2013*.

Yuan, X., Yang, L., Jones, B., Yu, H., & Chu, B.-T. (2016). Secure software engineering education: Knowledge area, curriculum and resources. *Journal of Cybersecurity Education, Research and Practice*, *2016*(1), 3.

MITRE. (2017a). Cwe list version 2.11. Retrieved from https://cwe.mitre.org/data/index.html

ACM, I. (2013). Computer science curricula 2013: Curriculum guidlines for undergraduate degree programs in computer science. Retrieved from https://www.acm.org/education/CS2013-final-report.pdf

OWASP. (2011). Owasp top 10 threats and mitigations exam - single select. Retrieved from https://www.owasp.org/index.php/OWASP_Top_10_Threats_and_Mitigations_Exam_-_Single_Select

OWASP. (2015). Injection flaws. Retrieved from https://www.owasp.org/index.php/Injection_Flaws

Visaggio, C. (2010). Session management vulnerabilities in today's web. *IEEE Security & Privacy*, *8*(5), 48–56.

MITRE. (2017b). Cwe-639: Authorization bypass through user-controlled key. Retrieved from http://cwe.mitre.org/data/definitions/639.html

MITRE. (2017c). Cwe-352: Cross-site request forgery (csrf). Retrieved from https://cwe.mitre.org/data/definitions/352.html

MITRE. (2017d). Cwe-306: Missing authentication for critical function. Retrieved from https://cwe.mitre.org/data/definitions/306.html

MITRE. (2017e). Cwe-601: Url redirection to untrusted site ('open redirect'). Retrieved from https://cwe.mitre.org/data/definitions/601.html

García, F., Pedreira, O., Piattini, M., Cerdeira-Pena, A., & Penabad, M. (2017). A framework for gamification in software engineering. *Journal of Systems and Software*.