Computer and Information Science
www.ccsenet.org/journal.html
Vol. 2, No. 4
November 2009

Developing a Secure Web Application Using OWASP Guidelines

Khairul Anwar Sedek
Faculty of Computer Science and Mathematics, Universiti Teknologi MARA (UiTM)
Kampus Arau, 02600 Arau, Perlis, Malaysia
Tel: 60-19-474-6960 E-mail: khairulanwar@perlis.uitm.edu.my

Norlis Osman
Faculty of Computer Science and Mathematics, Universiti Teknologi MARA (UiTM)
Kampus Arau, 02600 Arau, Perlis, Malaysia
Tel: 60-19-477-2808 E-mail: norlis@perlis.uitm.edu.my

Mohd Nizam Osman
Faculty of Computer Science and Mathematics, Universiti Teknologi MARA (UiTM)
Kampus Arau, 02600 Arau, Perlis, Malaysia
Tel: 60-19-413-5362 E-mail: nizamos@perlis.uitm.edu.my

Hj. Kamaruzaman Jusoff (Corresponding author)
Faculty of Forestry, Universiti Putra Malaysia, 43400 Serdang, Selangor, Malaysia
Tel: 60-3-8946-7176 E-mail: kjusoff@yahoo.com

The research is financed by Research Management Institute (RMI) of UiTM.

Abstract

Developing a secure Web application is very difficult task. Therefore developers need a guideline to help them to develop a secure Web application. Guideline can be used as a checklist for developer to achieve minimum standard of secure Web application. This study evaluates how good is OWASP guideline in helping developer to build secure Web application. The developed system is then tested using code auditing and penetration testing to identify the achievement of the system security for the application. After applying the testing techniques from Open Source Security Testing Methodology (OSSTMM) on the Top Ten Critical vulnerabilities as defined by OWASP, a standard measure score are calculated. The score is used to decide on the level of security of the developed web application. A high percentage score would indicate that the guideline helps in building a secured web application. Hence, the result proved that OWASP guideline is effective in ensuring the trustworthiness of the system and can be used as referral by other web developer especially in developing applications for a university.

Keywords: Web Application, Security

1. Introduction

In the age of Internet and World Wide Web, system security has become an important issue in any global web based information systems. This can be seen from the strong commitments of system security professionals, the research community, and major application software vendors. Recently web technology has developed rapidly and affected people in many aspects of lives and working. Many daily activities, which required face-to-face interaction, can now be conducted over the World Wide Web. Web applications are crucial components of our life. They cover critical activities such as economic transactions, e-commerce, e-government, e-business, e-procurement, e-education and many more.

The processes of building a secure web application need one or more guidelines to make it a secure system. Without guideline, it is impossible to develop a secure system. Gritzales & Spinelis (1997) provide the best practice for addressing security issues and threats, which can be prevented using security services. Stuart et al. (2001) has been addressing a very comprehensive guideline including system, network, and software security. Ed (2002) provided a step-by-step guide to computer attacks and effective defences including web application. Darothy (1998) assert that the best defence against security breaches is to make use of the tools and knowledge of good software engineering practice to prevent security attacks by developing and evolving secure system. This means that the requirements related to security issues must be identified and included early in the development and evolution of systems. Care must be taken to ensure that the security requirements are correct and complete.

The first OWASP (2003) issued the top 10 most critical web application security vulnerabilities to be considered in building secure web application with an update on the latest vulnerabilities in 2004. OWASP issued the latest Top 10 vulnerabilities (2007) which show that A1-Cross Site Scripting (XSS) has moved to the top of the list from 4th place and A2-Injection Flaws from 6th place to 2nd place. While A7-Broken Authentication and Session Management vulnerability has moved down from 3rd place in the list to the 7th placement. Several new vulnerabilities have been identified in the Top 10 2007 list such as A3-Malicious File Execution, A4-Insecure Direct Object Reference, A5-Cross Site Request Forgery (CSRF) and A9-Insecure Communications. Some vulnerability evolved from the old vulnerability into its own vulnerability definition, for example A10-Failure to Restrict URL Access is redefined from the previous A2 2004-Broken Access Control and A8-Insecure Cryptographic Storage is redefined from A8 2004 Insecure Storage. Meanwhile vulnerabilities such as unvalidated input, buffer overflow and denial of service have been taken out from the top 10 2004 list.

Mark et al. (2002) provided an open source document of guideline to building secure web application. These documents are intended to help developer to design, building and maintain a secure web application.

In the article "Buzzing About Security", Sandra (2002) explained why developer should have a framework as guideline to building secure web application. She recommends OWASP because it is an open source document where everybody can use it for developing, building, and testing secure system.

Pete (2002) produced an Open-Source Security Testing Methodology (OSTMM). It created an accepted method for performing a thorough security test including Internet presence points, information security, social engineering, networking, and physical security. Mark Curphey (2007) has produced a draft of OWASP Web Security Certification Criteria document to be used to test and certify the security of Web application. It can be a framework of Web application security certification. OSTMM has more comprehensive testing methodology to measure the result of security testing.

Andrew et al. (2003) have evaluated security features of Microsoft Windows Server 2003 with .Net Framework and IBM WebSphere. The study evaluated the level of effort required for developers and administrator to create and deploy secure web application. Both platforms provide infrastructure and effective tools for building secure application but the .Net platform scored higher than WebSphere.

Most of the studies discussed about security tools and how to build secure web application and also on current web threats. However, there is no evidence that the security tools or recommended security practice is adequate to build a secure web application. To conclude, there is no study on evaluation of security guideline or standard that can be followed by a developer in building a secure web application.

In this study, we evaluate how effective is OWASP guideline to help developer to develop secure Web application. To evaluate, OWASP guideline is used to develop secure Web application.

For this purpose, the required activities are integrated tightly into the development process. The security measures are carried out during the entire process, as early as possible when they become relevant. This ensures that security problem are discovered when they are still easy to counter. The process we used improves the quality (by its requirement on design, implementation and testing) and trustworthiness of the system and reduces evaluation time and cost.

2. Methodology

OWASP guideline is applied throughout the software development life cycle (SDLC) phases in application development which are system planning, system analysis, system design, implementation, and testing as shown in Figure 1. Security requirement defined in the OWASP such as authentication and authorization, input validation, and session handling is applied to ensure the system being developed is secure from security risks.

To have a standard measurement, score value for the vulnerabilities mentioned above is defined in Table 1 below. The table has been modified from the simplified web application framework to evaluate the guideline.

At the end of testing, all score will be summed up and the percentage will be calculated. This percentage will be analysed to determine whether the web application is secure or not. The following Table 2 represent the meaning of percentage in order to get the result or conclusion of the research for the guideline provided by OWASP.

3. Results and discussion

The study has successfully done 35 securities testing in the area of re-engineering, authentication, session management, input manipulation, output manipulation and information leakage testing. The test found 8 possible vulnerabilities out of thirty five possible testing (22.86%). The testing result is shown inTable 3.

The study has successfully done 35 securities testing in the area of re-engineering, authentication, session management, input manipulation, output manipulation and information leakage testing. The test found 8 possible vulnerabilities out of 35 possible testing (22.86%).

Based on the testing that we have done, for the area of re-engineering and information leakage security testing, with a result of 100%, we found that the guidelines helps immensely in building a secured web based application at least from the top 10 most critical vulnerabilities. Meanwhile testing the security in the area of authentication and session management, with a result of 78% and 76% respectively, shown the usage of guideline in this area gave adequate contribution to building a secured application. While in the area of input manipulation and output manipulation security, the above 85% result proved the guidelines to be considerable help in building a secured web application at least from the top 10 most critical vulnerabilities. Overall, the results of the security testing on ITMS yield the average security percentage of 86.27%.

Our study has demonstrated how we evaluated a Web application by using OWASP Guideline to building a secured Web Application. The guideline was evaluated using OSSTMM proposed by Pete Herzog, with the development for Industrial Training Management System (ITMS) Web application as a case study. This study has successfully applied the OWASP guideline to ITMS Web application. The result of all criteria that was evaluated indicated that OWASP contributed significantly in developing a secured Web application at least in reducing the number of security vulnerabilities especially for Web based university application.

4. Conclusion

The guideline was evaluated using OSSTMM proposed by Pete Herzog, with the development for Industrial Training Management System (ITMS) Web application as a case study. This study has successfully applied the OWASP guideline to ITMS Web application. The result of all criteria that was evaluated indicated that OWASP contributed significantly in developing a secured Web application at least in reducing the number of security vulnerabilities especially for Web based university application.

Overall, taking into account security does not make web design more complicated; it should be one of many natural elements of web design nowadays. It is not hard to consider if it is included into the process of web design right from the beginning.

Incomplete development processes leave the applications at risk, no matter how structured the company's development process may be. To achieve a greater level of application security, mature development practices that focus specifically on Web application security need to be implemented.

References

Andrew Jaquith, Frank Heidt, & Chris Wysopal, (2003). Security Evaluation: Microsoft Windows Server 2003 with .NET Framework and IBM WebSphere. Retrieved from http://www.atstake.com/research/reports/eval_ms_ibm/acrobat/atstake_eval_ms_ibm.pdf.

Andrew Jaquith. (2002). The Security of Appplications: Not All Are Created Equal. Retrieved from http://www.atstake.com/research/reports/acrobat/atstake_app_unequal.pdf.

Boiler. (2003). Hacking Techniques: Issue#2 – Boucing Atttacks. Retrieved from http://www.governmentsecurity.org/articles /HackingTechniquesIssue2-BouncingAttacks.php.

David Endler, Brute-Force Exploitation of Web Application Session IDs. Retrieved from http://www.cgisecurity.com /lib/SessionIDs.pdf

Dorothy E. Denning. (1998). Cyberspace Attacks and countermeasures". In Internet Besieged Countering Cyber Scofflaws, ACM Press, New York, N.Y.

Ed Skoudis. (2002). Counter Hack: A Step-by-Step Guide to Computer Attacks and Effective Defenses. Upper Sadle River, NJ: Prentice Hall PTR.

Grizalis, Stefanos & Spinellis, Diomidis. (1997). Addressing Threats and Security Issues in World Wide Web Technology. In Proceeding CMS '97 3rd IFIP TC6/TC11 ,33-46. IFIP, Chapman & Hall. Retrieved from http://www.dmst.aueb.gr/dds/pubs/conf/1997-CMS-WebSec/html/w3sec.html.

Mark Curphey. (2004). The OWASP Testing Project, (2004, December). Retrieved from http://www. owasp.org/index.php/OWASP_Testing_Project

Mark Curphey. (2004). The Ten Most Critical Web Application Security Vulnerabilities. Retrieved from http://www.owasp.org.

Mark Curphey. (2007). SpoC 007 - The OWASP Web Security Certification Framework. Retrieved from http://www.owasp.org/index.php/SpoC_007_-_The_OWASP_Web_Security_Certification_Framework.

OWASP Top Ten Web Application Vulnerabilities Version 1.0. (2003). Retrieved from http://prdownloads.sourceforge.net/OWASP/OWASPWebApplicationSecurityTopTen-Version1.pdf?download.

OWASP Top 10 2004. (2004). Retrieved from http://www.owasp.org/index.php/Top_10_2004.

OWASP Top 10 2007 (2007). Retrieved from http://www.owasp.org/index.php/Top_10_2007.

Pete Herzog. (2002), Open Source Security Testing Methodology Manual (OSSTMM). Retrieved from http://isecom.securenetltd.com/osstmm.en.2.1.pdf.

Sandra Kay Miller. (2002, January). Buzzing About Security. InfoSecurity. Retrieved from http://infosecuritymag.techtarget.com/2002/jan/departments_news.shtml.

Scott Berinato. (n.d.). The Bugs Stop Here. Retrieved from http://cio.idg.com.au/index.php?id=597539172.

Shynlie Simmons, Hacking Techniques: Web Application Security. (2005, November) East Caroline University. Retrieved from http://www.infosecwriters.com/text_resources/pdf/HackingTechniques_WebApplicationSecurity.pdf.

Stuart McClure, Joel Scambray, & George Kurtz, (2001). Hacking Exposed: Network Security Secrets and Solutions, Third Edition (3). : Osborn/McGraw Hill.

Table 1. Score value for the guideline evaluation

Score Value	Score Meaning
1	Not secure
2	Partly not secure
3	Fully secure.

Table 2. The definition of the security percentage calculated

Total Score	**Definition**
Less 25%	The guideline failed to help building a secure web application
26% - 50%	The guideline help eliminate some vulnerabilities but not enough to have secure application
51% - 79%	The usage of the guideline is adequate to build secure application
80% - 100%	The guideline helps building secure web application at least from the top 10 most critical vulnerabilities.

Table 3. Security Testing Result

No.	**Items**	**Score (1-3)**
	Re-Engineering	

1	Decompose or deconstruct the binary codes, if accessible	3
2	Determines the protocol specification of the server/client application	3
3	Guess program logic from the error/debug messages in the application output program behaviours/performance	3
	TOTAL	**9/9**
	Authentication	
4	Find possible brute force password guessing access points in the application	3
5	Find a valid login credentials with password grinding, if possible	1
6	By pass authentication system with spoofed tokens	3
7	By pass authentication system with replay authentication information.	1
8	Determine the application logic to maintain the authentication session – number of (consecutive) failure logins allowed, login timeout etc.	3
9	Determine the limitations of access control in the application – access permissions, login session duration, idle duration	3
	TOTAL	**14/18**
	Session Management	
10	Determine the session management information – number of concurrent session, IP-based, authentication, role-based authentication, identity based authentication, cookies usage, session ID in URL encoding string, session ID in hidden field variables, etc	1
11	Guess the session ID sequence and format	3
12	Determine the session ID is maintained with IP address information; check if the same session information can be retried & reused in another machine	1
13	Determine the session management limitations – bandwidth usages, file download/upload limitations, transaction limitation, etc	3
14	Gather excessive information with direct URL, direct instruction, action sequence jumping and/or pages skipping	3
15	Gather sensitive information with Man-in-the-Middle attacks	1

16	Inject excess/bogus information with Session-Hijacking techniques	3
17	Replay gathered information to fool the applications.	1
	TOTAL	**16/21**
	Input Manipulation	
18	Find the limitations of the defined variables and protocol payload – data length, data type, construct format, etc.	3
19	Use exceptionally long character-strings to find buffer overflow vulnerability in the applications	3
20	Concatenate commands in the input strings of the applications	2
21	Inject SQL language in the input strings of database-tired web applications	3
22	Examine "Cross-Site Scripting" in the web applications of the system	1
23	Examine unauthorized directory/file access with path/directory traversal in the input strings of the applications	3
24	Use specific URL-encoded string and/or Unicode-encode strings to bypass input validation mechanism of the applications	3
25	Execute remote commands through "Server Side Include"	3
26	Manipulate the session/persistent cookies to fool or modify the logic in the server-side web application.	2
27	Manipulate the (hidden) field variable in the HTML forms to fool or modify the logic in the server-side web application	3
28	Manipulate the "Referrer". "Host", etc. HTTP Protocol variables to fool or modify the logic in the server-side web applications.	3
29	Use illogical/illegal input to test the application error-handling routines and to find useful debug/error message from the applications.	3
	TOTAL	**32/36**
	Output Manipulation	
30	Retrieve valuable information stored in the cookies	2
31	Retrieve valuable information from the client application cache	3

32	Retrieve valuable information stored in the serialized objects.	3
33	Retrieve valuable information stored in the temporary files and objects	3
	TOTAL	**11/12**
	Information Leakage	
34	Find useful information in hidden field variables of the HTML forms and comments in the HTML documents	3
35	Examine the information contained in the application banners, usage instructions, welcome messages, farewell messages, application help messages, debug/error message, etc.	3
	TOTAL	**6/6**

Table 4. Summary of the Result Security Testing

Security Testing Category	**Marks**	**Percentage (%)**
Re-engineering	9/9	100%
Authentication	14/18	78%
Session Management	16/21	76%
Input Manipulation	32/36	89%
Output Manipulation	11/12	92%
Information Leakage	6/6	100%

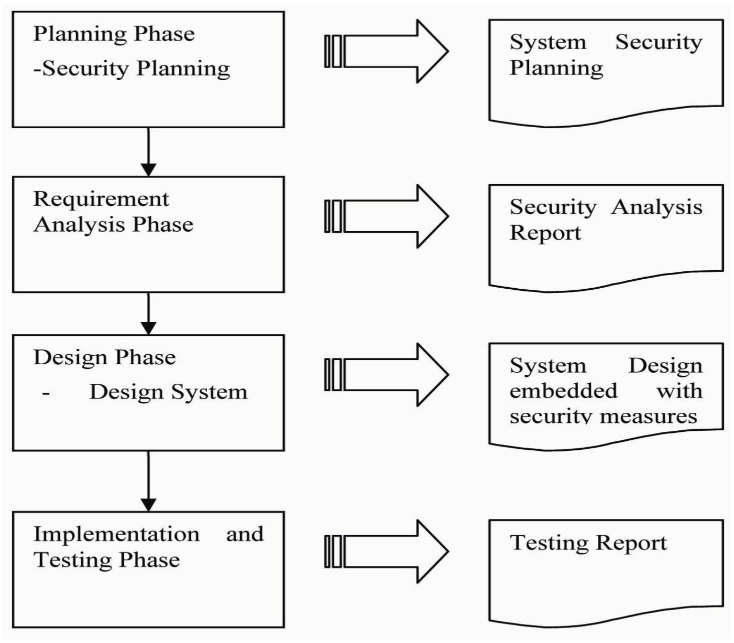

Figure 1. Security in Software Development Life Cycle