International Journal of Electrical and Computer Engineering (IJECE)
Vol. 9, No. 2, April 2019, pp. 1393~1398
ISSN: 2088-8708, DOI: 10.11591/ijece.v9i2.pp1393-1398

ImageSubXSS: an image substitute technique to prevent Cross-Site Scripting attacks

P. M. D. Nagarjun[1], Shaik Shakeel Ahamad[2]
[1,2]Department of CSE, K L University, Vijayawada, India
[1]Nagabot Software Development Pvt. Ltd., Nellore, India
[2]CCIS, Majmaah University, Majmaah, Kingdom of Saudi Arabia

Article Info

Article history:

Received Apr 20, 2018
Revised Nov 6, 2018
Accepted Nov 30, 2018

Keywords:

Cross-site scripting
ImageSubXSS
Malicious JavaScript
Web application attacks
XSS attacks

ABSTRACT

Cross-Site Scripting (XSS) is one of serious web application attack. Web applications are involved in every activity of human life. JavaScript plays a major role in these web applications. In XSS attacks hacker inject malicious JavaScript into a trusted web application, execution of that malicious script may steal sensitive information from the user. Previous solutions to prevent XSS attacks require a lot of effort to integrate into existing web applications, some solutions works at client-side and some solutions works based on filter list which needs to be updated regularly. In this paper, we propose an Image Substitute technique (ImageSubXSS) to prevent Cross-Site Scripting attacks which works at the server-side. The proposed solution is implemented and evaluated on a number of XSS attacks. With a single line, developers can integrate ImageSubXSS into their applications and the proposed solution is able to prevent XSS attacks effectively.

Corresponding Author:

P. M. D. Nagarjun,
Department of CSE,
K L University,
Vaddeswaram - 522502, Guntur District, Andhra Pradesh, India.
Email: pmdnr@nagabot.com

1. INTRODUCTION

According to the Open Web Application Security Project (OWASP), Cross-Site Scripting attacks [1] are popular and dangerous web application attacks. Attackers inject malicious JavaScript into vulnerable web applications. When a user opens that infected web application in their browser, malicious JavaScript will be executed and steals cookies and other sensitive information from the user [2].

XSS vulnerabilities are growing every year, XSS attacks increased by 39% in 2017 and almost 79% of web applications having vulnerable JavaScript library [3]. PHP is a popular programming language used to develop web applications. It is easy to learn and with this, it is easy to develop websites. So a lot of developers without proper knowledge of security developing web applications with a lot of vulnerabilities.

Proposed ImageSubXSS will prevent Cross-Site Scripting attacks, this solution works at the server side. ImageSubXSS is implemented and tested with PHP language. The rest of the paper is organized as follows: Section 2 shows different types of XSS attacks. Section 3 discusses related work. In Section 4 we describe the proposed solution to prevent XSS attacks. In Section 5 we evaluated the proposed tool. Section 6 discusses the limitations and future works. Finally, Section 7 concludes briefly.

2. CROSS-SITE SCRIPTING ATTACK TYPES

Two most popular XSS attack types are reflected (Non-persistent) and Stored (Persistent) attacks [4]. The proposed solution in this paper is able to prevent both of these attacks.

2.1. Reflected (non-persistent)

In Reflected XSS attacks, malicious script will be sent to the server through request and server will reflect back that malicious code in response. That malicious code will be executed at user browser and steals user's sensitive information. Normally Reflected attacks reach user through E-mail or malicious websites.
Example malicious link:
http://example.net/searchpage.php?searchkeyword=<script>alert ("XSS Attack"); </script>

2.2. Stored (persistent)

In Stored XSS attacks, malicious scripts will be stored permanently on the web server database [5]. An attacker can inject scripts in a vulnerable web application by writing malicious scripts in the comment section, post message section, etc. Whenever a normal user visits that comment section of the website, then those injected scripts will be executed and may steal user's sensitive information.
Example: The attacker injects malicious code in the comment section in the vulnerable web page.
<script> alert("XSS Attack"); </script>

3. LITERATURE WORK

Hydara *et al.* [6] conducted a literature study on 115 research papers related to XSS attacks from 2004 to 2012. Their study shows that Reflected XSS attacks are popular XSS attacks compared to other types of XSS attacks. Their study shows that most of the work done on detecting and preventing XSS attack vulnerabilities instead of removing vulnerabilities from source code.

Shanmugasundaram, Ravivarman, and Thangavellu [7] stated that websites contain XSS vulnerabilities because developer lack required knowledge on XSS attacks and developers are unable to implement existing solutions in their applications. There are different solutions exist to prevent XSS attacks at client-side and server-side.

Noxes: Client-Side Solution. Developed by Kirda *et al.* [8], it is a client-side web application firewall. Noxes works on a client browser. It acts as a personal firewall in the user browser. If the user requests any URL, checks the filter list to validate URL before sending server request. There will be an alert box for every new URL to validate the URL by the user. To avoid so many alerts there will be a threshold (k), for each page can have k external links and these are considered valid for one click or that session. In this solution, the user needs to have knowledge on deciding which URL is safe and which is not safe.

SWAP: Server-Side Solution. Developed by Wurzinger *et al.* [9], SWAP means Secure Web Application Proxy. It is a reverse proxy technique. Input request is ignored and only output responses are filtered to find XSS attacks. In web application's JavaScript, it will modify code by replacing Script with Script Ids (ex: <script> to <scrip5>). So executable JavaScript will be converted into non-executable JavaScript. While sending a response to the client it checks for JavaScript. If it finds any JavaScript then it considered as an attack and if there is no JavaScript means safe then it will decode all Script Ids and send the response to the client. SWAP may not be suitable for applications with rapid code changes.

BIXSAN: Server-Side Solution. Developed by Chandra and Selvakumar [10], BIXSAN means Browser Independent XSS Sanitizer. BIXSAN only allows static tags and remove all dynamic tags. JavaScript tester in BIXSAN finds existing in-line JavaScript code in static tags, filters scripted tags and finally creates DOM for content. That DOM will be stored in a database or return to the client. BIXSAN can prevent Reflected and Stored XSS attacks. Document DOM will be created at server-side uses this DOM in the client browser.

EWAF: Server-Side Solution. Developed by Kazanavicius *et al.* [11], EWAF means Embedded Web Application Firewall. EWAF works based on the blacklist and whitelist filters. Based on user request EWAF analyze which attacks were possible like XSS, SQL Injection, etc. After analyzing possibilities of attacks, the request sent through corresponding XSS module or SQL Injection or other modules based on attack type. Then take a decision whether it is common request or attack request based on results of corresponding attack module.

4. IMAGESUBXSS: THE PROPOSED TECHNIQUE TO PREVENT XSS ATTACKS

4.1. Overview

ImageSubXSS is Image Substitute technique to prevent Cross-Site Scripting attacks. In this system, characters involved in XSS attacks are replaced with corresponding images. Figure 1 shows the ImageSubXSS system overview.

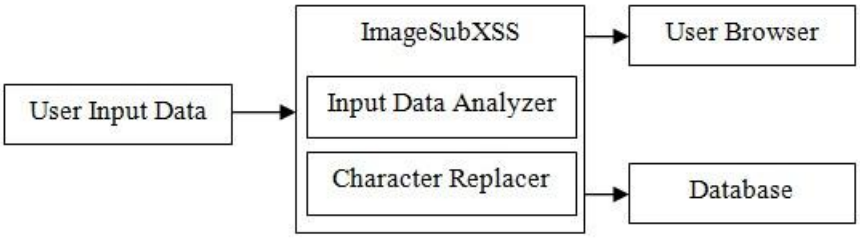

Figure 1. ImageSubXSS system overview

Developers need to include single ImageSubXSS tool file to avoid XSS attacks in their web applications. This tool will analyze every POST and GET data at server-side and checks for malicious characters in user input data [12]. If there are any those characters it will replace those characters with corresponding images. After modifications, the data will proceed to the remaining application. Based on a web application that data may be reflected back to the user's browser or stored in a database. Figure 2 shows how the input data handled by the ImageSubXSS system by simply considering "<" as a malicious character [13] and replacing it with < (lt.png) image.

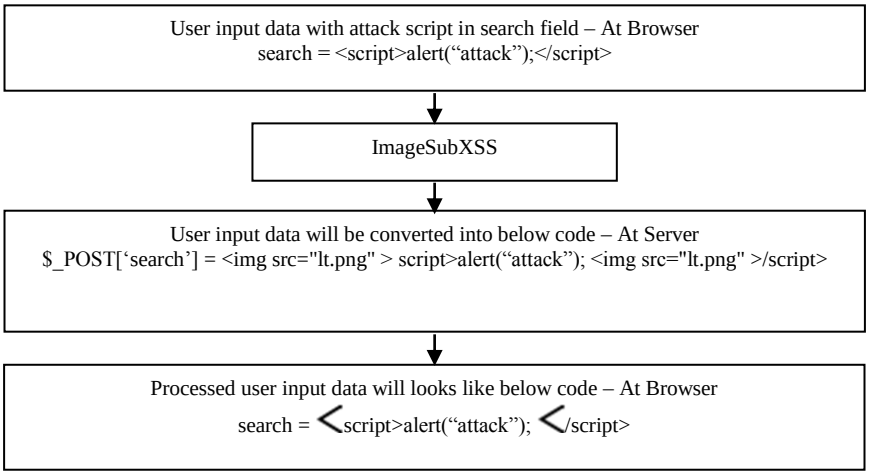

Figure 2. The flow of input data in ImageSubXSS system

4.2. ImageSubXSS images

ImageSubXSS tool uses images to prevent Cross-Site Scripting attacks. This tool will check malicious characters like ", ', (, \, <, and &#. It will replace those malicious characters with corresponding images. All these images are highly compressed to improve performance. Sizes of these images are between 200 to 600 Bytes. Table 1 shows the characters and corresponding images, these characters are the most frequent special characters involve in XSS attacks.

ImageSubXSS tool allows developers to choose different color images based on their web application background. Black Images: These images are clearly visible on web applications with light color related backgrounds, check example shown in Figure 3. White Images: These images can be used in web applications with dark color related backgrounds, check example shown in Figure 4.

Figure 3. Black images on the light color background

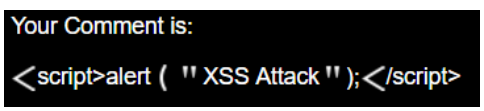

Figure 4. White images on the dark color background

Table 1. Characters and Corresponding Image

Character	Black Image	White Image
“	"	"
‘	'	'
(	(	(
\	\	\
<	<	<
&#	&#	&#

4.3. Testing on real world web application

We tested web application with our proposed system. ImageSubXSS is able to prevent all XSS attacks. We implemented the system in PHP programming language but it can be implemented in any language and can be used on any web application. As a developer, they need to include ImageSubXSS tool file. So the only modification required to implement ImageSubXSS in existing web application is to add below code at beginning of PHP file, which handles user input data.

```
<?php
//Input data in this file is filtered by ImageSubXSS tool include "ImageSubXSS.php"; ...
//Remaining code may contain code to show data to user or
//store data in database
?>
```

From the above code, ImageSubXSS.php file contains functions to filter and replace malicious characters with corresponding images. Figure 5 shows a web page without any XSS security and Figure 6 shows a web page with the ImageSubXSS system to prevent XSS attacks.

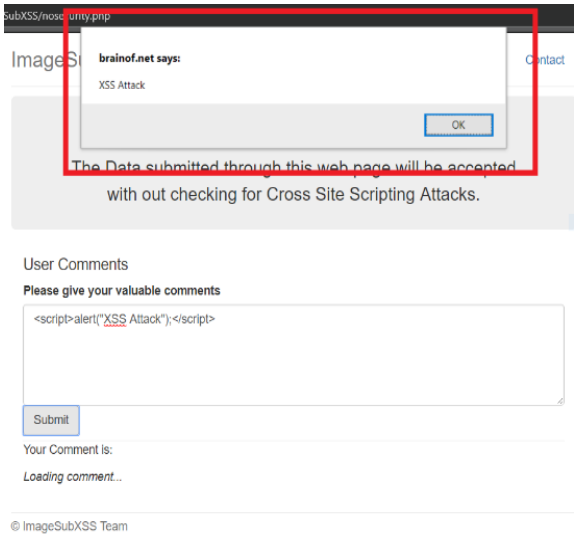

Figure 5. A web page without any XSS security measures

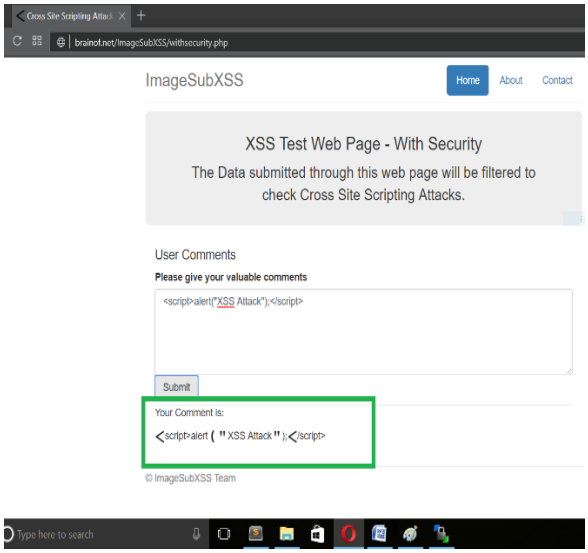

Figure 6. A web page with an ImageSubXSS security system

5. EVALUATION

We manually tested and verified our solution with more than 100 XSS attacks [14] from different sources like OWSAP XSS Filter Evasion Cheat Sheet [15] etc. ImageSubXSS system able to prevent all attack vectors with 100% prevention rate, Figure 7 and Figure 8 shows example XSS attack vectors. ImageSubXSS will not break the web application look even if there is an XSS attack.

Figure 7. XSS attack 1 and corresponding ImageSubXSS output

Figure 8. XSS attack 2 and corresponding ImageSubXSS output

6. LIMITATIONS AND FUTURE WORK

ImageSubXSS system can be implemented into existing web application with a single line of code but sometimes developers need to configure properly to integrate this solution into their web applications. If developers have their own XSS prevention functions those need to be executed first otherwise ImageSubXSS may cause problems while handling user data. Compared to large and complex web applications proposed solution works most effectively on small and simple web applications.

Currently, Image Substitute system to prevent XSS attacks was implemented in PHP, so it only supports PHP web applications. We are working on implementations in other programming languages like Java, Python, ASP, etc. ImageSubXSS system shows negligible performance issues while handling large user input data

7. CONCLUSION

Web applications are growing rapidly. Cross-Site Scripting attacks are popular web application attacks, XSS attacks are difficult to prevent. Previous solutions have difficulties in integrating into existing websites. Image Substitute Technique named ImageSubXSS was proposed to prevent Cross-Site Scripting attacks in web applications. ImageSubXSS can be integrated into existing websites with a single line of code. ImageSubXSS is most effective in simple web applications. Our evaluation shows that ImageSubXSS can prevent every possible XSS attack existed on popular XSS Cheat Sheets.

REFERENCES

[1] S. Gupta and B. B. Gupta, "Cross-Site Scripting (XSS) Attacks and Defense Mechanisms: Classification and State-of-the-art," *International Journal of System Assurance Engineering and Management*, vol. 8(1), pp. 512–530, Jan 2017.

[2] B. Rexha, A. Halili, K. Rrmoku, and D. Imeraj, "Impact of Secure Programming on Web Application Vulnerabilities," in *2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS)*, pp. 61–66, 2015.

[3] G. Podjarny, "XSS Attacks: The Next Wave | Snyk," *Snyk*, 08-Jun-2017. [Online]. Available: https://snyk.io/blog/xss-attacks-the-next-wave.

[4] V. Nithya, S. L. Pandian, and C. Malarvizhi, "A Survey on Detection and Prevention of Cross-site Scripting Attack," *International Journal of Security and Its Applications*, vol. 9(3), pp. 139-152, 2015.

[5] X. Li and Y. Xue, "A Survey on Server-side Approaches to Securing Web Applications," *ACM Comput. Surv.*, vol. 46(4), pp. 54:1–54:29, Mar 2014.

[6] I. Hydara, A. B. Sultan, H. Zulzalil, and N. Admodisastro, "Current State of Research on Cross-site Scripting (XSS) – A Systematic Literature Review," *Information and Software Technology*, vol. 58, pp. 170–186, Feb 2015.

[7] G. Shanmugasundaram, S. Ravivarman, and P. Thangavellu, "A Study on Removal Techniques of Cross-Site Scripting from Web Applications," in *2015 International Conference on Computation of Power, Energy, Information and Communication (ICCPEIC)*, pp. 0436–0442, 2015.

[8] E. Kirda, C. Kruegel, G. Vigna, and N. Jovanovic, "Noxes: A Client-side Solution for Mitigating Cross-site Scripting Attacks," in *Proceedings of the 2006 ACM Symposium on Applied Computing*, Dijon, France, pp. 330–337, 2006.

[9] P. Wurzinger, C. Platzer, C. Ludl, E. Kirda, and C. Kruegel, "SWAP: Mitigating XSS Attacks Using a Reverse Proxy," in *Proceedings of the 2009 ICSE Workshop on Software Engineering for Secure Systems*, pp. 33–39, 2009.

[10] S. C. V. and S. Selvakumar, "BIXSAN: Browser Independent XSS Sanitizer for Prevention of XSS Attacks," *SIGSOFT Softw. Eng. Notes*, vol. 36(5), pp. 1–7, Sep 2011.

[11] E. Kazanavicius, V. Kazanavicius, A. Venckauskas, and R. Paskevicius, "Securing Web Application by Embedded Firewall," *Elektronika ir Elektrotechnika*, vol. 119(3), pp. 65–68, Mar 2012.

[12] T. Scholte, W. Robertson, D. Balzarotti, and E. Kirda, "An Empirical Analysis of Input Validation Mechanisms in Web Applications and Languages," in *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, Trento, Italy, pp. 1419–1426, 2012.

[13] A. Javed and J. Schwenk, "Systematically Breaking Online WYSIWYG Editors," in *Information Security Applications*, pp. 122–133, 2015.

[14] R. Assis, "XSS Cheat Sheet," *Leanpub*, 2018.

[15] OWASP, "XSS Filter Evasion Cheat Sheet-OWASP." [Online]. Available: https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet.