Global Scientific JOURNALS

GSJ: Volume 7, Issue 8, August 2019, Online: ISSN 2320-9186
www.globalscientificjournal.com

A PROPOSED APPROACH FOR PREVENTING CROSS-SITE SCRIPTING (XSS)

Twana Asaad TAHA
Department of Software Engineering
University of Firat
twana.assad@gmail.com

Abstract— **In this paper the great threat XSS (Cross-Site Scripting) is introduced, that faced with the web pages. Because of the impacts of such web threats during design and developing web pages, web developers must be aware and have adequate knowledge about varies type of web attacks and how to prevent or mitigate them. Web developers should have knowledge about how attackers attack websites and exploit weak points on websites during filling forms, registering and opening suspicious links or attachments in emails. The important of this subject is to provide great details and information about identifying impacting and protecting from these types of web threats. It aims to provide both web developers and users with enough knowledge while developing and using websites to prevent from such attacks and reduce them impacting and protecting from these types of web threats. It aims to provide both web developers and users with enough knowledge while developing and using websites to prevent from such attacks and reduce them. In this paper use PHP's functions to evaluate the efficiency of web pages for implementing it and to prevent XSS attack.**

Keywords—component: Web application Security, Injection threats, input validation.

I. INTRODUCTION

In 1989 the term "Web" that stands for World Wide Web (shortened to www) developed by Tim Berners, www is the service that the internet provides users to visit or browse documents (web pages) that linked by hypertext links. The main purpose of web is to share and participate data like, text, image, video etc. between web users. HTML (Hyper Text Markup Language) is a basic file of all web page, Web hangs on using hyperlinks to visit between web pages with a program named a web browser like Internet Explorer, Mozilla Firefox and Google Chrome. At early websites composed with only HTML (Static websites) and are restricted to exchange data. A few years latter world wide web became commercialized, a lot of server-side languages like PHP, ASP, JSP, Java Script, and VB Script as client-side language is invented and made the web most more interactive (Dynamic websites). However, now web site and web application after invented those languages and techniques are becoming more usable, but the web threats and web attackers have also become the biggest issue for web server, so web designers need to create these components, incorporating the applicable criteria that follow.

Cross-Site Scripting (XSS) threat is a code injection attack that lets an attacker to implement or execute harmful Java Script or other web script in browser of users. The gap (weak point) lets injection of inputs comprising HTML tags and client-side scripts code [1]. The code of XSS can be composed in any client-side scripting language. But JavaScript is used widely than other web scripts. This attack can likewise be positioned through a link in an Email or on a web page that seems to be invented from the hacker's web page [2].

Among those attacks, XSS is the most well – know security issues in web application in the present and it's dangerous because it gives the opportunity to other kind of attacks. Due to healthcare institute statistics on 2017 the XSS threat got third place of most common and dangerous web security threats (see figure 1).

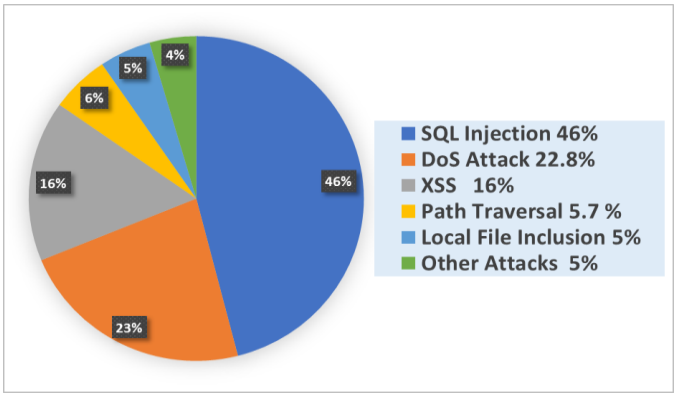

Figure 1. Top five Attack on Web Application of healthcare institute 2017.

II. BACKGROUND

The Open Web Application Security Project (OWASP) is a noncommercial group that helping organizations to maintain, purchase, and develops trusted software applications. In the first time, it was built as a venture to assign a methodology of standard testing for industry of application security through web [3].

The security technician can mix and use OWASP recommendations within their work. The (table 1) demonstrates top 10 web security risks, Therefore, users can use those standards as a criterion to test services or applications they use [4].

TABLE I. OWASP WEB SECURITY RISKS LIST

Order	Threat Type
1	A1-Injection
2	A2-Broken Authentication and Session Management
3	A3-Cross-Site Scripting (XSS)
4	A4-Broken Access Control
5	A5-Security Misconfiguration
6	A6-Sensitive Data Exposure
7	A7-Insufficient Attack Protection
8	A8- Cross-Site Request Forgery (CSRF)
9	A9- Using Components with Known Vulnerabilities
10	A10 -Under protected APIs

A. XSS Examples

In the following simple example of XSS malicious code, we assume that the hackers intend is to steal the cookies information of victim's user via manipulating an XSS vulnerability in the webpage. This can be accomplished by having HTML code of victim's browser parser.

<script>

window.location='http://www.hackerwebsite/?cookie='+document.cookie

</script>

The above code of JavaScript directs the user's browser to a dissimilar URL, starting an HTTP request to the hacker's web server. The URL comprises the victim's cookies like a query parameter, which the hacker can get from the request after it reaches to his/her web server. As soon as the hacker has obtained the cookie information, he can utilize them to illegally access the victim and manipulate additional attacks.

B. XSS Types

Overall, XSS attacks classified into two classes: stored (or persistent) and reflected (or non-persistent). In addition to these types, there is another non-famous type of XSS attack called DOM based XSS [5]. The following sub-sections are explaining different classes of XSS, (See table 2) shows common type of XSS and characters of each type.

TABLE II. XSS TYPES

XSS Type	Server	Client
Stored	Stored Server	Stored Client
Reflected	Reflected Server	Reflected Server
DOM-Based		Subset of Client

- Stored or Persistent XSS

In stored XSS attacks, the injected malicious code is stored forever on the target servers. In this situation, first the attacker attempts to find weak points in the web application. If any weak points have been located, then the attacker injects a malicious script that has ability to take classified information of the user or might cause other damages [6].

- Reflected XSS

The reflected XSS attacks are opposed to stored XSS attacks, reflected XSS attacks attach malicious code that doesn't locate on the web server. This type of XSS attacks send malicious or injected links to target (victims) via email, or place the links in a web page or to redirect to another web server [6].

- DOM Based XSS

One of the Cross-Site Scripting attack is DOM XSS that depends on inappropriate processing of data form its related DOM objects in the web page. The attacker can manipulate a lot of DOM objects to produce the XSS condition [6].

III. RELATED WORK

There are numerous studies, investigations and researches have done in the last years about detecting and preventing Cross-Site Scripting issues associated with web sites and web applications. Many solutions of XSS vulnerability has invented by researchers, unfortunately XSS threat still happens in both web sites and web applications. Web pages still face various XSS attacks and the most significant is session hijacking, hackers steal the cookie data of victim's user and they may use the same sessions. Tools and instruments are designed to protect the web application security; it characterizes the arrangement that helps developers for building up a secure web application and web site. It says XSS is a result of despicable filtering of user input and proposed cautious expulsion of undesirable scripts or HTML tags that can be executed on the web pages.

A strategy is utilized which cover security all through the application life cycle. The application life-cycle stages like web page design, web development, and run time execution ought to be taken after the OWSAP security rule in an exact way. Sometimes developer absence of learning can be the reason for security disappointment. The approach of Saxena et al. [7] mitigate Cross-Site Scripting attack by using Microsoft ASP.Net platform that permit both dynamic and static web page contents in defined HTML, they control the recognized context of generation. Their method finds proper sanitization routine for code that dynamically produces contents for instance JavaScript and web page links (URLs).

The hacker can attack the web server by executing the malicious script, therefore content filtering is used as input filtering technique. In Java Script sometimes filter method fails to identify the unwanted input contents [8].

IV. MITIGATION AND PREVENTING AND OF XSS

For avoiding Cross-Site Scripting risk, the first is to place to apply a context dependent output encoding. In some cases, it might be enough to encode the HTML special characters, such as opening and closing tags (See table 3) [9].

TABLE III. COMMON CHARACTERS REPLACING TO DEFEAT XSS

Replace	With
<	&lt
>	&gt
(	&#40
)	&#41
#	&#35

XSS threats can be avoided by validating and checking data that are provided by users to ensure consistence with the required format for web applications, there are four suggested mechanisms for user input validation [10,11].

- **Replacement** is a way to search for dangerous user inputs then substitutes those dangerous codes with correct and true characters.
- **Removal** is a way also to find dangerous inputs but opposed to replacement by removing them.
- **Escaping** way changes (or marks) key characters of the data to avoid it from being interpreted in a dangerous code.
- **Restriction** way checks the user inputs to limited non-malicious.

OWASP's guide to secure development gives three rules for dealing with user data [12]:

- Accept only known valid data
- Reject recognized harmful data
- Clean harmful data

V. APPROACHED SYSTEM TO PREVENT XSS

XSS is to be stated how to happen. it takes place when web forms receive malicious scripting code that has been injected to the victim computer then the web browser will execute. In this approached system, secure code PHP functions are proposed to detect and prevent form XSS attack by using two methods, the first one is to use regular expression to validate data from web forms that has been entered by the user, and the second one is another regular expression to check and protect every input entry that has a possibility to face a malicious script in it so even if the hacker inject XSS script code in the input field , this malicious code will not be allowed to be executed and immediately will be removed.in this work, vulnerable PHP web sites has used to assess the efficiency of the proposed system before and after applying it. (See figure 2).

For preventing XSS attack such htmlEntities()and htmlspecialchar(), PHP web programming language provides built-in functions that adapt characters to HTML entities, by using regular expressions that can be found it easier, also replace and work with string as shown in algorithm (1), the primary strategy is AllowList regular expression, which is by tolerating just expected and trusted user inputs data it will make a validation while DenyList regular expression includes checking if the information contains unsuitable data and evacuate all conceivable suspicious characters, for example, as HTML starting tags and ending tags <> with any text inside

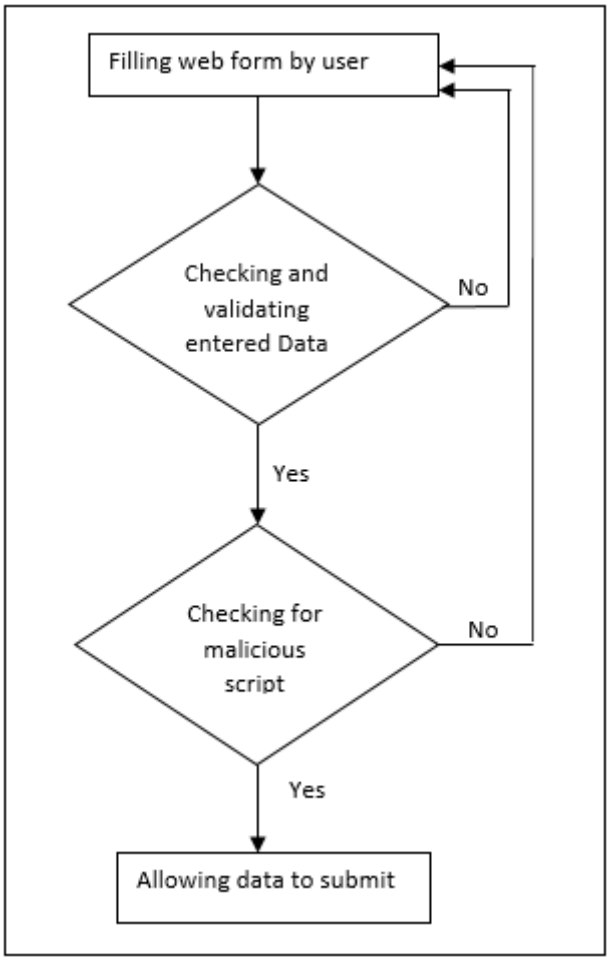

Figure 2. approached system to prevent XSS flowchart

Algorithm (1): The Proposed System to prevent XSS Attack Process

```
Initialize AllowList_RegExp
[
Hold only alphabetic latter lower and upper case
Email address reg_exp
URL: Protocol, domain name, page reg_exp
MasterCard reg_exp
Credit card numbers reg_exp
Phone number regex
Currency amount reg_exp
]
Initialize DenyList_RegExp
[
Band JavaScript regex
Band out VBscript regex
Band out HTML tags regex
Band style tags property regex
Clear away ASCII code characters excel regex
Clear away any Unicode code point that is unused in the
current Unicode regex
Clear away quotes and backslashes regex
Clear away hex tag regex
Clear away <img src> regex
]
While user input not empty do
For each item of AllowList _RegExp do
If user input match item
Return user_input
End if
Next
For each item of DenyList _RegExp do
If user input match item
Remove user_input
Return empty string
End if
Next
Next
End
End
```

VI. CONCLUSIONS AND FUTURE WORK

In this paper, XSS threats, its principles and aspects, dangers types are presented in this work. Nowadays, XSS counts as one of the prime threat for web applications. There are three types of XSS, stored XSS, reflected XSS and DOM based XSS. XSS is simply known as "script injection attacks". Web developers must be aware to face attackers and how they attack websites and taking advantages out from weak points of websites during form filling, logging in and opening fishy links or attachments in emails. This paper presented several types of (RG) regular expression that can be used to find threats. In the first step the (XSS) Cross Site-Scripting introduced and explained the level of risky and the main aspects for preventing and detecting from this common threats and approached a system to protecting web page while any attacker tries to run malicious code in victim's browser, for the next step, we will try to develop a tool or extension to browser to automatically detect and prevent running malicious code in the web form.

VII. REFERENCES

[1] H. Shahriar and M. Zulkernine, "S2XS2: A Server-Side Approach to Automatically Detect XSS Attacks", 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, Sydney, NSW, 2011, pp.7-14

[2] V. K. Malviya, S. Saurav and A. Gupta, "On Security Issues in Web Applications through Cross Site-Scripting (XSS)," 2013 20th Asia-Pacific Software Engineering Conference (APSEC), Bangkok, 2013, pp. 583-588

[3] M. Rouse, "OWASP (Open Web Application Security Project)" Retrieved form http://searchsoftwarequality.techtarget.com/definition/OWASP last accessed 25/11/2017

[4] "About The Open Web Application Security Project ", Retrieved form https://www.owasp.org/index.php/About_The_Open_Web_Application_Security_Project, last accessed 12/12/2017

[5] A. Wiegenstein, and F. Weidemann, "White paper on the cross-site scripting threat," Virtual Forge, version 1.2, 2007

[6] A. Shrivastava, S. Choudhary and A. Kumar, "XSS vulnerability assessment and prevention in web application," 2016 2nd International Conference on Next Generation Computing Technologies (NGCT), Dehradun, 2016, pp. 850-853

[7] P. Saxena, D. Molnar, and B. Livshits, SCRIPTGARD: Preventing Script Injection Attacks in Legacy Web Applications with Automatic Sanitization, MSR-TR-2010- 128, Sept. 2010

[8] D. Patil and K. Patil, "Client-side automated sanitizer for cross-site scripting vulnerabilities," International Journal of Computer Applications, vol. 121, no. 20, 2015

[9] L. K. Shar, and H. B. K. Tan, "Defending against cross-site scripting attacks", Computer, vol. 45, issue. 3, pp. 55-62, March 2012

[10] G. Bayse ," A Security Checklist for Web Application Design" , 2004 SANS Institute InfoSec Reading Room, P.6

[11] S.Cook , "A Web Developer's Guide to Cross-Site Scripting " SANS Institute 2003 , P.11

[12] R. Mohammed , "Protecting Website from Cross Site Script Attack" , 2015 Iraqi Journal of Information Technology , P.95