International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 05 Issue: 03 | March - 2021 ISSN: 2582-3930

Cross Site Scripting

Donal babu, Kevin Chacko, Arun Padmanabhan

Department of Computer Applications

Saintgits College Of Applied Sciences, Kottayam, India

Abstract -As the internet becomes more complex and one of the most common malicious attacks is code injection. Many malicious web users that exploit these vulnerabilities is cross-site scripting (XSS). XSS is defined as the number one and utmost prevalent website vulnerability on the internet. XSS is long and arduous to repair. Based Cyber-attacks the security of web applications is at a risk of information stealing or tampering. Cross-site scripting are vulnerabilities that targets web applications by embedding scripts in a web page that will be executed at client side or server side. This paper gives a 'positive security model' to prevent web applications and its users from XSS.

Key Words:Cross-site scripting,Cyber Crime

1.INTRODUCTION

Most websites today contain dynamic content which gives its viewers a more interactive. A dynamic website is generated by two different types of interactivities: Client-side scripting and Server-side scripting. Moreover, in a dynamic website, we can make ourself susceptible to a popular and very powerful security vulnerability that plain static websites. This threat is called cross-site scripting (XSS).
Attackers direct their attention towards these vulnerabilities and insert malicious content into the client-side browser without the user's knowledge which allows an attacker to gain access to the user's personal information. As new XSS threats develop, malicious attackers will continue to discover innovative ways to exploit this weakness. It is necessary to have a comprehensive understanding of cross site scripting due to its significant impact in today's internet security worldwide.

Cross site scripting

Cross-site scripting is a type of computer security vulnerability that is found in web-based applications which allows code injection by malicious web users into any webpage that is viewed by other users. The term "Cross site scripting", originated when a malicious website could potentially load a website onto another window and then use JavaScript to read or write information on the other website, which was later redefined as injection. The vulnerabilities are exploited by attackers to bypass access controls such as the same origin policy. The most frequent kinds of web applications that are victimized by XSS attacks are search engine, discussion boards, web-based emails and posts.

Programming languages utilized in XSS attacks are Sun Microsystem's java, JavaScript, Action Script, VB Script, Microsoft's Active X, Adobe's Flash, HTML or XHTML, RSS and Atom Feeds

TYPES OF ATTACKS

1.DOM-Based Attack

The problem exists within the client-side script. If an attacker hosts a malicious site, which contains a vulnerable website on a client's local system, a script can be injected. Now the attacker can run the privileges of that user's browser on their system "Local Zone".

2.Persistent Attack

This vulnerability is susceptible to the most powerful kinds of attacks. First, the data is stored on the server provided by a web application. Then it is later reopened and shown to other users on a webpage without any html encoding. An attacker can exploit this vulnerability and affect a large magnitude of users, this web application

can also be infected by a cross-site scripting virus or worm.

3.Non-Persistent Attack

It is also referred to as a Type 1 or a stored vulnerability. It is by far the most common type of the three. If a web user provided data to a server-side script, it instantly generates a resulting page back, a resulting page without html encoding can be intercepted by an invalidated user. This maliciousclient-side code can then be injected into the dynamic page. The attacker can apply a little social engineering so as to persuade a user to follow a malicious URL that will inject code into the resulting page. After the attacker has accomplished that, he now has full access to that web pages content.

SOLUTIONS FOR USERS

The user has two main possibilities himself from a XSS attack. The most effective solution is to disable all scripting languages in his browser and e-mail reader. Alternatively, the user should visit and follow only trusted sources. Even if all scripting languages are disabled, attackers may still be able to influence the content of a webpage by inserting only HTML tags. So, this first advice only helps decreasing the possibility to be victim of a code insertion attack. With scripting enabled, an attacker can hide a malicious link by altering the representation of the link in the bottom bar of the client browser.

Another possible attack is through an insecure integrated application, like flash, Windows Media player etc. To prevent these types of attacks, user is advised to uninstall these integrated applications or use additional software tools, like an antivirus program. In the end, it is up to web page developers to modify their pages in order to eliminate these types of problems

SOLUTIONS FOR DEVELOPERS

It is very hard if not impossible for the end user to solve the XSS problem. So, web page developers it is very hard if not impossible for the end user to solve the XSS problem. So, web page developers are responsible to implement their web applications in a way that these types of attacks don't have any effect. As two web applications are never the same, the developer must adjust his web application to their special requirements.

Web developers must evaluate whether their sites will send untrusted data as part of an output stream. The most relevant procedure to make a web application resistant to XSS attacks is to validate and filter any input a server-side script receives. The complexity of the filtering depends on the requirements of the web applications and the chosen server-side scripting language. Most special characters have a special meaning to the syntax of the malicious code when inserted into a webpage or URL.

To prevent a XSS attack the developer must filter out a series of characters in any user input received. Every web page has a range of inputs which can be checked sequentially for every input field.

THE PROPOSED SOLUTION

The recent Cyber-attacks have shown the loop holes in the implementation of security Mechanism in the Web Applications and the network. This leads to the needs for improvisation in the implementation of the existing standards and establishing more secure web applications. The existing XSS defense mechanism have following limitations:

- ✓ Whenever a new threat is detected and the researches find preventive solution for the same, an update in the form of releases or versions is mandatory for the client.
- ✓ The existing server-side solutions are better than client solution as they do not increase the burden on clients to upgrade every time.
- ✓ Security Testing solutions are language dependent. Thus, a complete solution
  That is capable of handling the latest XSS attacks. This research aims at providing security mechanism for these web applications.
- ✓ The proposed solution is a 'Server-side solution' based on 'Positive Security Model' which by default rejects or blocks all the html tags, scripts, programming language constructs, insecure keywords. This is better than the existing defenses as it does not increase the processing time in matching the attack vectors from blacklist. In the existing solutions the inputs from user are taken from the web browser as untrusted data, which go

through a filtering process to get a “clean” status. This clean data is stored in the database to generate clean output from that after output sanitization

The main components of the proposed solution are
1. Sanitizer
2. XSS Filter
3. Whitelist

3. CONCLUSIONS

Cross-site scripting has always remained one of the major concerns for the web applications. No single solution was available that could prove to be effective in mitigating XSS attacks. A complete solution for vulnerability removal from the source code of applications before deployment was needed.

This research presented a server-side solution that has successfully removed the limitations of the existing solutions as the proposed solution can be integrated to any web application at any stage and any kind of modification is not needed in the already existing code of the web application to implement this solution. The database will be safe from all the attacks. The proposed solution provides a complete solution that is flexible in nature. Any new attack will not be able to harm the security of the system.

Future research in this solution aims at implementing either Genetic Algorithm or Machine learning for making the system self-adaptive.

REFERENCES

1. Maurya, S. Singhrova, A., "Cross Site Scripting Vulnerabilities and Defenses: A Review," International Journal of Computer Technology Applications, Vol.6, Issue 3, pp.478-482, June 2015

2. Shanmugam, J.; Ponnavaikko, M., "A solution to block Cross Site Scripting Vulnerabilities based on Service Oriented Architecture," 6th IEEE/ACIS International Conference on Computer and Information Science, ICIS 2007, pp.861-866, July 2007

3.Kieyzun, A.; Guo, P.J.; Jayaraman, K.; Ernst, M.D., "Automatic creation a. of SQL Injection and cross-site scripting attacks," IEEE 31st International Conference on Software Engineering, 2009. pp. 199-209, May 2009

4. Kirda, E.; Jovanovic, N. Kruegel, C. Vigna, G.;"Client-side cross-site scripting protection", Computers & Security, vol. 28, Issue 7, pp.592- 604, Oct. 2009