Volume 5, Issue 12, December 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Investigation and Analysis of XSS Attacks on Web Applications: Survey

Shweta Sharma* M.Tech student, Dept. of IT AIET, Lko, India

Madhulika Sharma HOD of Dept. of CSE & IT AIET, Lko, India

Usama Husain Asst. Prof., Dept. Of CSE AIET, Lko, India

Abstract— Now a day, the use of web based applications has become increasingly popular in our day to day life as making online payments for shopping, reading news paper, making various types of transactions etc. At this time when we use web applications there is an increase in number of various types of attacks that target them. To secure a web application against hacking is a big challenge today. Cross-Site Scripting (XSS) technique to hack a web application is very harmful. It is one of the most common types of hacking technique to hack the web application. Cross-Site Scripting vulnerabilities are being exploited by the hackers to steal web browser's resources such as credentials, cookies etc. by injecting the malicious JavaScript code on the victim's web applications. In this paper we have presented the analysis of detection and prevention of Cross-Site Scripting help to avoid this type of attack. We also describe a technique to detect and prevent this kind of manipulation and hence eliminate Cross-Site Scripting attack.

Keywords— Cross-Site Scripting (XSS) attack, detection, prevention, Web Application

I. INTRODUCTION

The full-fledged popularity of World Wide Web (www) has paved way for the nimbus of web applications in every field especially social networking, banking, health services, finance etc. At the starting, there were only static web pages .That provides static information expressed in graphics and text. Now a day's internet is growing, the website become more popular, dynamic and professional. Current generation want to use web applications instead of static web sites. Most systems such as health care, Social Networks, blogs, banking, are relying on these web applications. We can use these web applications for communicating with other users via instant messaging, for editing and viewing video or even creating spreadsheets, for reading email and for managing their photographs and other files. Hence we can say that internet plays a vital role in our daily life. So security must be included to use these web applications. To provide a beneficial and safe networking environment is very necessary. If vulnerability occurs in these famous web applications, a lot of users will be attacked and the result can be very crucial and cannot be imagined. There are many attacks to hack web applications like SQLIAs, XSS attacks.

XSS attacks are one of the most common vulnerabilities in web based applications. Statistical reports [1] state that 63% of web applications accessed poses an average of 6 unsolved defects. According to Web Application Security Project (OWASP)[2] and Common Vulnerabilities and Exposures (CWE) [3] in the year 2013 projected that Cross – Site Scripting attack as the most crucial threat to web applicants.

Cenzic Application Vulnerability Trends Report (2013) XSS represents 26 percent of the total population respectively [4] and considers this attack as top most first attack.

Two recent incidents of XSS vulnerability [5] are highlighted as follows:

(i) Apple Developer Site (July 18, 2013)
(ii) Ubuntu Forums (July 14 and July 20, 2013)

These attacks carried out using HTML, VBScript, ActiveX, Flash, and other client – side languages. A weak input validation on the user interface leads Cross Site Scripting attacks to gather important information from account hijacking, changing of user settings, cookie theft. Prevention or detection of XSS is a topic of active research in the academia and industry. To gain those purposes, automatic tools and security system have been implemented, but none of them are accurate or complete enough to guarantee an absolute level of security on web application.

One of the important reasons of this perfection is that there is lack of complete and common methodology for the evaluation either in terms of needed source code modification or performance which in an overhead for an existing system. A system which will easily deployable and provide a good performance to prevent and detect the Cross-site scripting (XSS) attack is essential one.

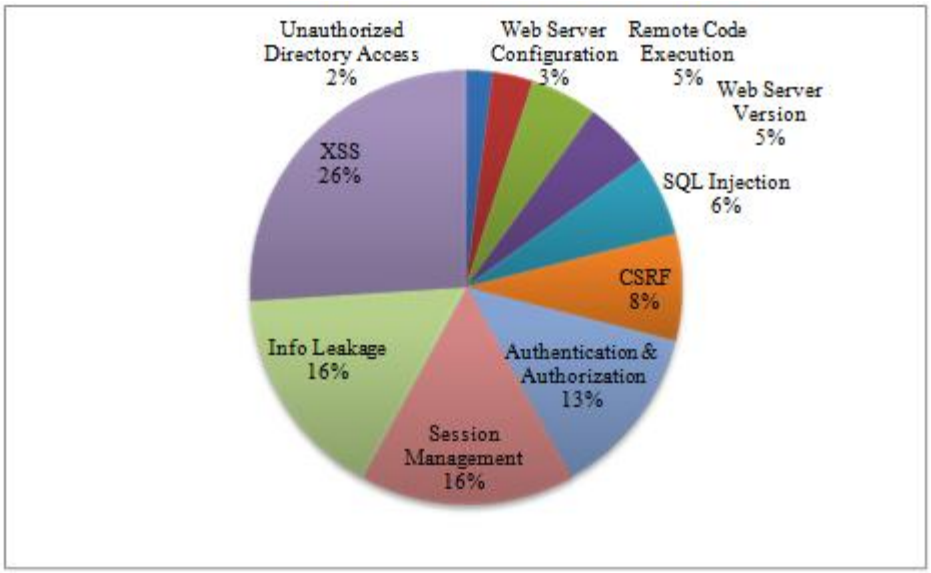

Fig 1 Web Application Security Vulnerability Population (2013)

II. OVERVIEW OF XSS ATTACK

We study in this section three main types of XSS attacks: persistent, non- persistent XSS attacks and DOM based attack.

What is Cross-Site Scripting (XSS)?

XSS also known as Cross Site Scripting .Still it is one of the most common web based vulnerability that allow hackers to run his/her their client side scripts (especially JavaScript code) into web pages viewed by other users.

In this mechanism, a hacker injects his malicious JavaScript code in the legitimate web application. When a user visit specially – crafted link, it'll execute the malicious JavaScript. Successfully exploited Cross – Site Scripting vulnerability will allow hackers to do phishing attacks, steal accounts and worms.

Example:

Let's imagine, a attacker has discovered XSS vulnerability in Gmail account and inject malicious script. When a user visits the site, it will execute the malicious script. The malicious script can be used to redirect users to fake Gmail page or can capture cookies. This stolen cookie can be used to login into your account and change password.

It will be easy to understand this attack, if you have the strong knowledge about following things:

- ✓ Strong Knowledge in HTML, JavaScript.
- ✓ Basic Knowledge in HTTP Client-Server Architecture.
- ✓ Basic Knowledge about server side programming (asp,jsp,php)

We study in this section three main types of Cross –Site Scripting (XSS) attacks:

- ✓ Stored XSS (AKA Persistent or Type I)
- ✓ Reflected XSS (AKA Non-Persistent or Type II)
- ✓ DOM Based XSS (AKA Type-0)

2.1. Threats of Cross – Site Scripting (XSS)

XSS poses severe application risks that include are as follows:

(i) Malicious Scripts can spy on what you do such as web history of sites visited and Track information you posted to a website and Access to personal data such as (Bank Account, Credit card).
(ii) By this attack hacker can access business data of user such as (construction details, bid details).
(iii) Hacker can perform phishing attack by XSS. Attacker can add login form posts to third party sites.
(iv) Attacker can add malicious JavaScript code to redirect the user.
(v) Attacker can also perform Pop-Up-Flooding by this attack. Pop-Up-Flooding means malicious JavaScript can make your website inaccessible, become inoperable, also can make browser crash.

2.2. Identifying Cross-Site Scripting vulnerabilities

Cross – Site Scripting (XSS) vulnerabilities may occur if:

- ➢ Input which is coming from the attacker into web application is not validated.
- ➢ Output which is generated to the browser is not html encoded.

2.3. How Cross-Site Scripting (XSS) attack works

For running malicious JavaScript code in a victim's browser, hacker must first find a way to inject a payload into a web based application that the victim visits.

To perform XSS attack to take place the vulnerable web application needs to directly include user input in its pages. A hacker can then insert a string that will be used within the web application and treated as code by the victim's browser.
The following server – side code can be used to display the most recent comment on a web page:

```
Print "<html>"
Print  "<h2>Most recent comment</h2>"
Print  database.latestComment
Print "</html>"
```

Above script is used to print the latest comment from a comments database and printing contents out to html page, assuming that comment printed out only consists of text.
Above html page is vulnerable to XSS attack because hacker could submit a comment that contains malicious payload such as **<script>doSomethingEvil ();</script>.**
Hacker's malicious code is as follows:-

```
<html>
<h2>Most recent comment!!!</h2>
<script>doSomethingEvil();</script>
</html>
```

When the web page loads in the victim's browser, the hacker's malicious script will execute most often without the user realizing or being able to prevent such an attack.

2.4. The anatomy of a Cross-site Scripting (XSS) attack

Three actors are needed to perform XSS attack-
(i)Website,(ii)Victim and (iii)Attacker
Below example is an example of XSS attack. The main goal of an attacker is to impersonate the victim by stealing the victim's cookie.

```
<script>
  window.location="http://evil2.com/?cookie=" + document.cookie
</script>
```

Below figure illustrates a step-by-step walkthrough of a simple Cross –Site Scripting(XSS) attack.

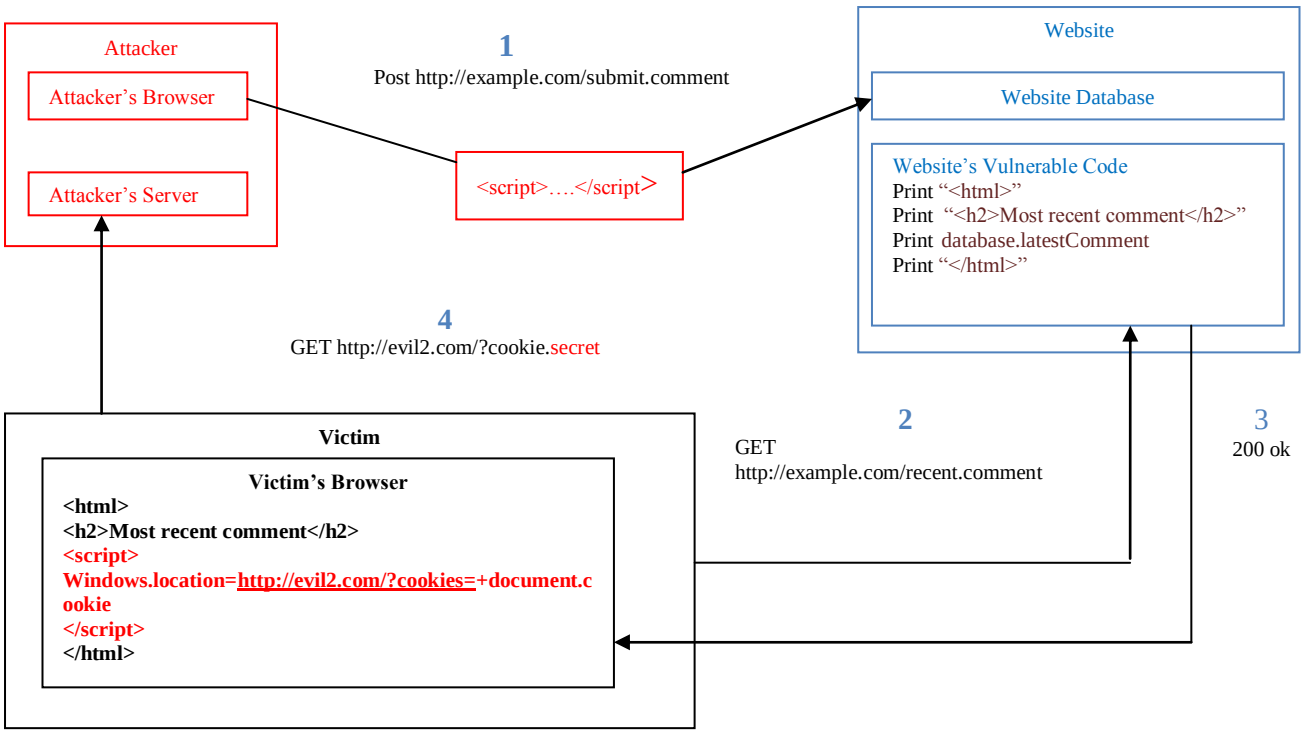

Fig 2 step-by-step walkthrough of a simple Cross –Site Scripting (XSS) attack.

(i) The attacker injects a payload in the website's database by submitting a vulnerable form with some malicious JavaScript.
(ii) The victim requests the web page from the website.
(iii) The website serves the victim's browser the page with the attacker's payload as part of the HTML body.
(iv) The victim's browser will execute the malicious script inside the HTML body. In above case it would send the victim's cookie to the attacker's server.

Attacker now simply needs to extract victim's cookie when hyper text transfer protocol request arrives to the web server. After which the attacker can use the victim's stolen cookie for impersonation.

2.5. Types of Cross-Site Scripting (XSS) Attack

We are going to define different types or categories of cross-site scripting (XSS) vulnerabilities and how they are related to each other. Before 2005, two primary types of Cross – Site Scripting (XSS) were identified, Stored XSS & Reflected XSS. In 2005, third type of XSS was identified by Amit Klein, which he coined DOM based XSS. These 3 types of Cross – Site Scripting (XSS) attacks are defined as follows:

2.5.1 Stored XSS (AKA Persistent or Type I)

In this type of attack injected script is permanently stored on the target servers, such as in a database, in a message forum, comment field, visitor log etc .Victim then retrieves malicious script from the server when it requests the stored information. Sometimes this stored XSS is referred to as Persistent or Type-1 XSS.

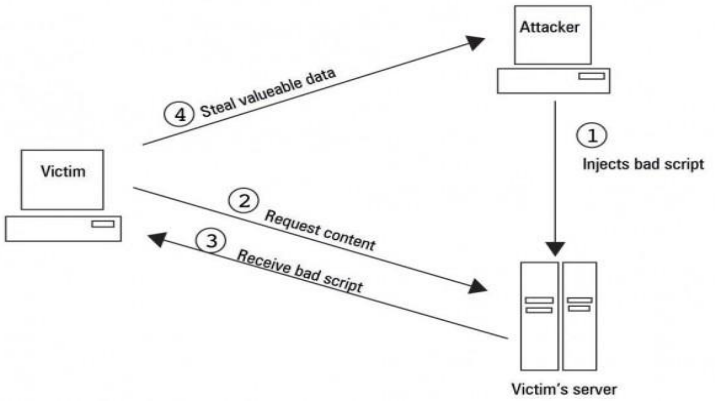

Fig 3 Process of Stored XSS attack

2.5.2 Reflected XSS (AKA Non-Persistent or Type II)

In Reflected XSS attack , which is also known as second type of Cross – Site Scripting attack, an attacker sends the injected script to the site which is vulnerable so that it will be immediately returned back to the user. Common method to perform this type of attack is, target pages where user input becomes part of the output of a page. A search page which is able to display the search terms to the user and could provide an outlet for this type of attack. The injected script or content in the user's input should never be stored by the web application.

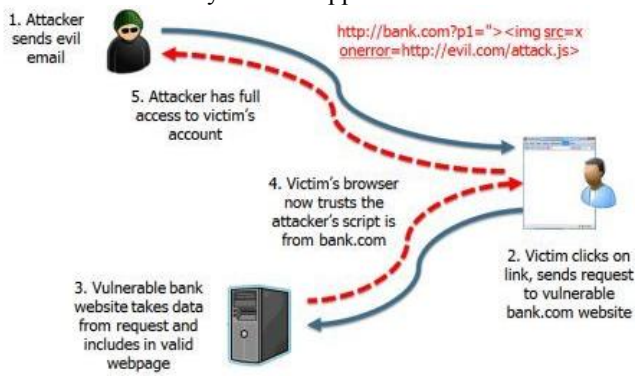

Fig 4 Process of Reflected XSS attack

2.5.3 DOM Based XSS (AKA Type-0)

DOM Based XSS attack which is also known as third Cross- Site Scripting attack. This type of attack occurs entirely in the web browser. An attacker performs the injection of malicious script into the web page; in the other types, the web server performs the injection. This type of attack generally involves server-controlled, trusted script or code that is sent to the client, such as JavaScript that performs sanity checks on a web form before the user submits it. DOM Based XSS is possible when server supplied script or code processes user supplied data and then injects it back into the page (such as with using of dynamic html).

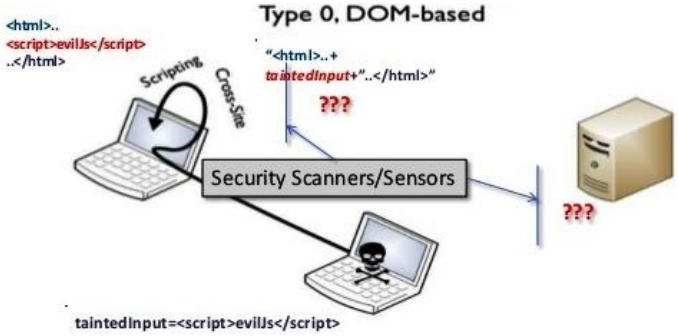

Fig 5 Process of DOM Based XSS attack

III. VIGILANT APPROACHES

There are many steps that users can take to check themselves from Cross-Site Scripting (XSS) and other types of attacks. We should apply our common sense while surfing. We should not click on link sent from unknown sources; close sessions when finished and users should regard the next criteria.

3.1 Restrict Untrusted JavaScript:

The most productive (but not foolproof) method for a utilizer to Prevent Cross-Site Scripting (XSS) attacks is to permit JavaScript to execute only if it emanates from a domain that the utilizer explicitly trusts.

3.2 Content Filterering:

In this type of technique we required to endeavour to detect and abstract all scripts from html (untrusted) afore sending it to the web browser.

3.3 Use Built In Browser Protections:

Many browsers are available now days but some of them have started to incorporate XSS aegis inherently.

Example: windows 8, Internet Explorer. These include an XSS filter as well as a smart screen filter that utilizes reputation to check against maleficent websites.

Maintain Good System Hygiene:

System and application which we are using must be up to date with latest updates and patches, which protected from malicious malware and securely configured.

IV. RELATED WORK

Over the past few years, there has been lot of research going on in both institutes as well as industries to prevent Cross-Site Scripting (XSS) attacks. Many researchers have proposed some detection and prevention mechanisms discussed below:

4.1 Detection and Prevention Techniques

Adam Kiezun et al.[6] they provide a mechanism for creating XSS attacks in web application. It is an automated tool. This mechanism is implemented for PHP. This sensing is predicated along the dynamic taint propagation, input generation, and input mutation to determine a form of user input that exhibits susceptibility. With the avail of this mechanism efficaciously and correctly detect the input predicated web application attacks.

Philip Vogt et al [7].The solution given in this document stops Cross-Site Scripting (XSS) attacks on the client side by passing over the flow of touchy information within the World Wide Web (WWW) browser. If sensitive information is about to be transferred to a third party, the utilizer can determine if this should be permits or not. As a consequence, the user receives an additional protective layer when surfing the www, without solely depending on the protection of the web application. This provider with the help of dynamic data tainting and static analysis.

Adam Kie˙zun et al[8].This mechanism support to identify the XSS attacks as well as SQLIA vulnerabilities. This mechanism is used on existing code which is not modified, produced concrete input that expose attacks malicious and it operate before software sis deployed. It is an automated process for creating attacks. This mechanism requires source code for analysis so that it is belong to the category white box testing. This is based on the input generation, input mutation, taint propagation to find alternative of an execution that exploit vulnerabilities.

William Robertson et al[9] introduced a way that addresses the limitations of anomaly-based intrusion detection schemes by using both characterization and generalization techniques. Inductive reasoning is used to make use of more abstract account in speech of an anomaly that enables one to group kindred attacks. If we talk about characterization, it is used to understand the class of attack that is affiliated with a group of anomalies. By using these two techniques, it's potential to cut time taken by an admin to make conclusions around the nature of the anomalies and their criticality.

V. CONCLUSIONS

In this paper we have represent the deep survey of the Cross-Site Scripting (XSS) attacks. In addition we have also described the various types of XSS attacks detection and prevention techniques known to date. This paper helps to many researchers to create more effective detection and prevention system with the help of our survey paper.

REFERENCES

[1] www.whitehatsec.com/home/resource/stats.html.

[2] Common Vulnerabilities and Exposures (The Standard for Information Security Vulnerability Names) http://cwe.mitre.org/.

[3] https://www.owasp.org/index.php/.

[4] H. Chen and M. V. Gundy, "Using randomization to enforce information flow tracking and thwart cross site scripting attacks," In Proceedings of the 16th Annual Network and Distributed System Security Symposium (NDSS), (2009.

[5] https://isis.poly.edu/~jcappos/papers/tr-cse-2013-02.pdf.

[6] Adam Kie˙zun,Philip J. Guo,Karthick Jayaraman,Michael D.Ernst:"Automatic Creation of SQL Injection and Cross-Site Scripting AttackS", ICSE˝ 09, May 16-24, 2009, Vancouver, Canada,978-1- 4244-3452-7/09/$25.00 © 2009 IEEE.

[7] Vogt, P., Nentwich, F., Jovanovic, N., Kirda, E., Kruegel, C.,Vigna, G. (2007, February). Cross Site Scripting Prevention with Dynamic Data Tainting and Static Analysis. In *NDSS*.

[8] Z. Su and G. Wassermann "The essence of command injection attacks in web applications". In ACM Symposium on Principles of Programming Languages (POPL'2006), January 2006.

[9] Robertson, W., Vigna, G., Kruegel, C., & Kemmerer, R. A. (2006, February). Using generalization and characterization techniques in the anomaly-based detection of web attacks. In NDSS.

