A Comprehensive Inspection Of Cross Site Scripting Attack

Mohit Dayal
Ambedkar Institute of
Advanced Communication
Technologies and Research,
New Delhi, India
mohitdayal.md@gmail.com

Nanhay Singh
Ambedkar Institute of
Advanced Communication
Technologies and Research,
New Delhi, India
nsingh1973@gmail.com

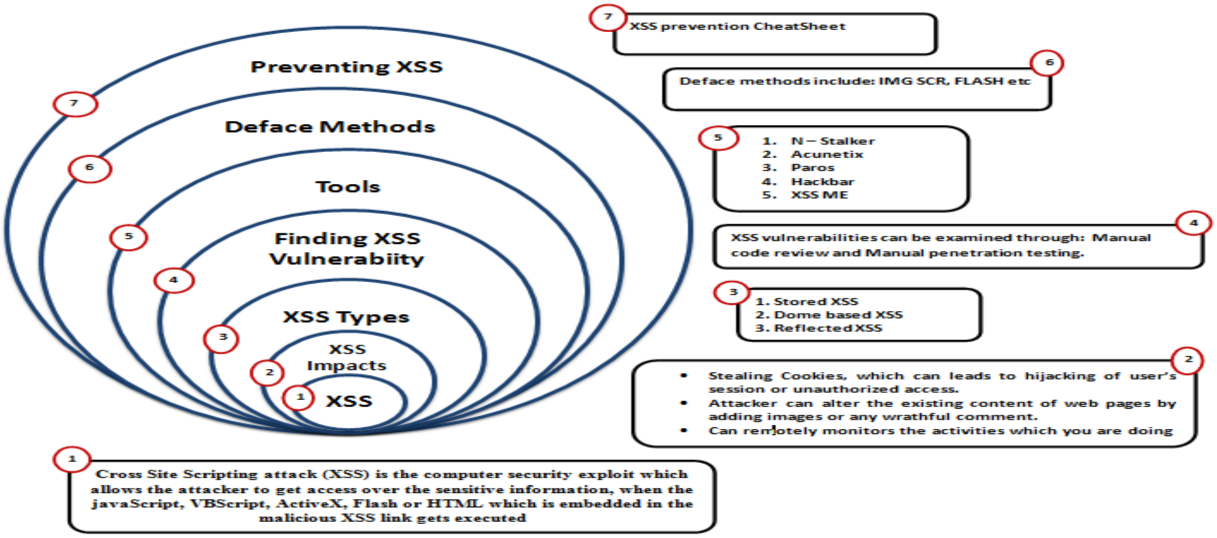
Ram Shringar Raw
Ambedkar Institute of
Advanced Communication
Technologies and Research,
New Delhi, India
rsrao08@yahoo.in

***Abstract:** Cross Site Scripting attack (XSS) is the computer security threat which allows the attacker to get access over the sensitive information, when the javaScript, VBScript, ActiveX, Flash or HTML which is embedded in the malicious XSS link gets executed. In this paper, we authors have discussed about various impacts of XSS, types of XSS, checked whether the site is vulnerable towards the XSS or not, discussed about various tools for examining the XSS vulnerability and summarizes the preventive measures against XSS.*

***Keywords-** Cross site scripting attack (XSS); Stored XSS; Domed based XSS; Reflected XSS*

I. INTRODUCTION

Cross Site Scripting attack (XSS) is the computer security threat which allows the attacker to get access over the sensitive information, when the javaScript, VBScript, ActiveX, Flash or HTML which is embedded in the malicious XSS link gets executed. There could be many ways through which XSS could be launched some of them are URL XSS which contains the malicious code embedded in the URL and gets executed when the user login, or input fields which allows to insert data which will remain on the website. To encode the malicious code attacker uses different techniques so that it appears authentic to the user, while it is not. Session hijacking through the execution of malicious script, stealing data by misusing user's privileges, posting ads in hidden IFRAME and pop-up are some of the ways through which attacker encodes the malicious code to look genuine to the user.



XSS prevention CheatSheet

Deface methods include: IMG SCR, FLASH etc

1. N – Stalker
2. Acunetix
3. Paros
4. Hackbar
5. XSS ME

XSS vulnerabilities can be examined through: Manual code review and Manual penetration testing.

1. Stored XSS
2. Dome based XSS
3. Reflected XSS

- Stealing Cookies, which can leads to hijacking of user's session or unauthorized access.
- Attacker can alter the existing content of web pages by adding images or any wrathful comment.
- Can remotely monitors the activities which you are doing

Cross Site Scripting attack (XSS) is the computer security exploit which allows the attacker to get access over the sensitive information, when the javaScript, VBScript, ActiveX, Flash or HTML which is embedded in the malicious XSS link gets executed

Figure 1 XSS: Analysis overview

ISBN: 978-1-5090-1666-2/16/$31.00 ©2016 IEEE

XSS examples:

1) Attack via Email.

Following steps are used by an attacker to achieve this XSS attack:

1. Attacker first builds the malicious link:
   " *<AHREF = http: // abcbank.com/ registration.cgi?clientprofile=<SCRIPT>malicious code</SCRIPT>>Clickhere</A>* "
2. After creating the malicious link it sends the URL to the user via email and persuades them to click on it.
3. User sends the request to the server to give access to the required page.
4. Genuine server sends the response page containing the malicious code.
5. Malicious XSS link gets executed on the user's browser.

2) Stealing user's cookies.

Following steps are involved in stealing user's cookies

1. Initially attacker hosts a web page containing malicious script.
2. Then the user hits the web page hosted by the attacker.
3. In response attacker's server sends the HTML page embedded with the malicious code.
4. Malicious HTML script gets executed on the user's browser.
5. The cookie logger which is present in the malicious script collects all the information of the user.
6. The user is then directed to the attacker's server.

3) Sending an unauthorized request.

1. Attacker first builds the malicious link.
2. After creating the malicious link it sends the URL to the user via email and persuades them to click on it.
3. User sends the request to the server to give access to the required page.
4. Genuine server sends the response page containing the malicious code.
5. Malicious XSS link gets executed and sends an unauthorized request.

4) XSS attack in comment field

To manipulate the data attacker uses the HTML web page's tag and then launches the attack by modifying the comment field with the malicious script. It then gets executed on the user's browser when the user spots the comment and activates it.

II. XSS IMPACT

Some of the technical and business impact of XSS is discussed below.

TABLE I XSS IMPACTS

Technical Impact	Malicious XSS link gets executed on the user's browser to hijack the user session, to damage websites, to add wrathful comment, to redirect user to the attacker server etc.
Business Impact	It includes the business value affected because of the affected system and data which is affected during the XSS exploit.

III. XSS TYPES

The three types of XSS are:

1) Domed based XSS.

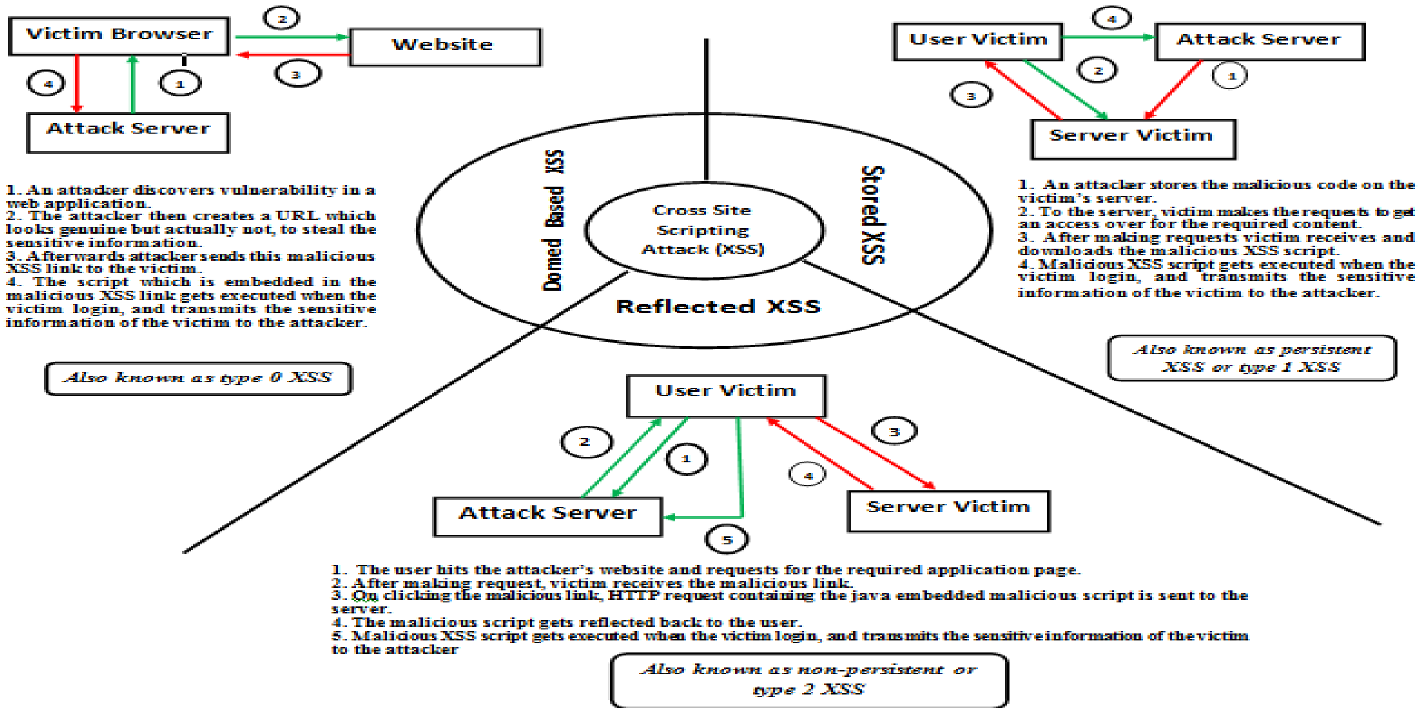
Domed based XSS is also known as type 0 XSS. It is a client side attack, in this attack user provided data is written to the document object model (DOM) by the web application client side script. The web application then simultaneously read data from the DOM. As a part of DOM, an attacker could also insert a payload which is executed when the data is read back from DOM. Attacker's payload is never sent to the server thus making it difficult for web application firewalls and security engineers, who are analyzing the server log to detect this attack.

2) Stored XSS

Stored XSS is also known as persistent XSS or type 1 XSS. In this attack, attacker embedded a malicious script which gets permanently stored on the server. One of the most common examples of this type is a XSS attack in a comment field of a blog or forum post.

3) Reflected XSS

Reflected XSS is also known as non persistent or type 2 XSS. In this attack attacker first builds the malicious link. After creating the malicious link it sends the URL to the user via email and persuades them to click on it. User then sends the request to the server to give access to the required page. Genuine server handles the request made by the user and sends the response page containing the malicious code. Malicious XSS link gets executed inside the user's browser when the user login and sends the sensitive information to the attacker.



1. An attacker discovers vulnerability in a web application.
2. The attacker then creates a URL which looks genuine but actually not, to steal the sensitive information.
3. Afterwards attacker sends this malicious XSS link to the victim.
4. The script which is embedded in the malicious XSS link gets executed when the victim login, and transmits the sensitive information of the victim to the attacker.

Also known as type 0 XSS

1. An attacker stores the malicious code on the victim's server.
2. To the server, victim makes the requests to get an access over for the required content.
3. After making requests victim receives and downloads the malicious XSS script.
4. Malicious XSS script gets executed when the victim login, and transmits the sensitive information of the victim to the attacker.

Also known as persistent XSS or type 1 XSS

1. The user hits the attacker's website and requests for the required application page.
2. After making request, victim receives the malicious link.
3. On clicking the malicious link, HTTP request containing the java embedded malicious script is sent to the server.
4. The malicious script gets reflected back to the user.
5. Malicious XSS script gets executed when the victim login, and transmits the sensitive information of the victim to the attacker

Also known as non-persistent or type 2 XSS

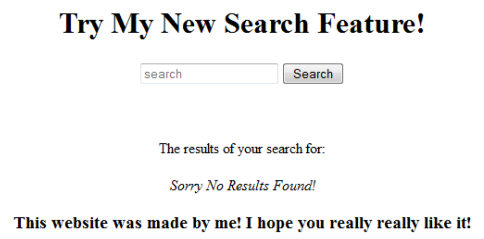
Figure 2 Working of various XSS types

The above figure shows the working of various types of XSS attack.

IV. IMPLEMENTATION RESULTS

Reflected XSS

1: Checking for Vulnerability towards Reflected XSS.
Web server used: XAMPP
Script used: " <script>alert("XSS")</script> "



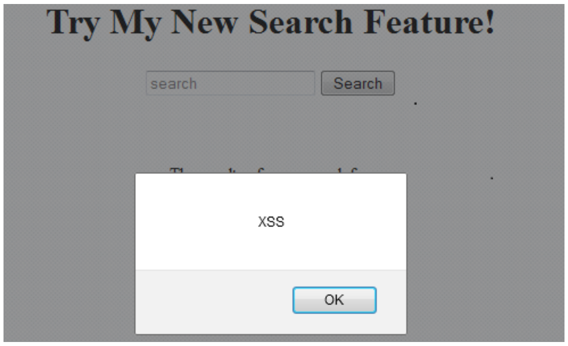
Try My New Search Feature!

The results of your search for:

Sorry No Results Found!

This website was made by me! I hope you really really like it!

Figure 3 Original website



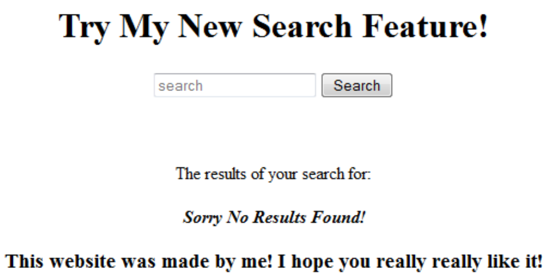
Try My New Search Feature!

XSS

Figure 4 Website showing vulnerability towards reflected XSS

Output: On executing the above mentioned script, the pop up message contains the bold words; therefore the particular website is vulnerable towards the Reflected XSS attack.

2: Modifying content of the website
a) Replacing the content to bold format
Script used**:** <b>



Try My New Search Feature!

The results of your search for:

Sorry No Results Found!

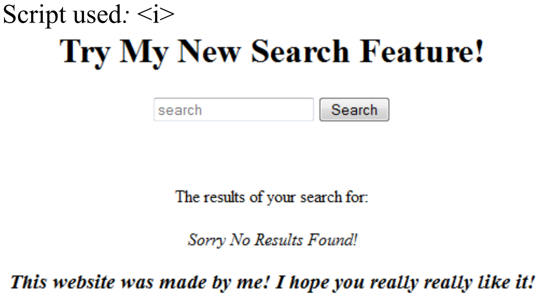
This website was made by me! I hope you really really like it!

Figure 5 Content of the website converted to bold

Output: Comparing it with the original website, the content of the website has been formatted to bold.

b) Replacing the content to italic format

Script used*:* <i>



Try My New Search Feature!

The results of your search for:

Sorry No Results Found!

This website was made by me! I hope you really really like it!

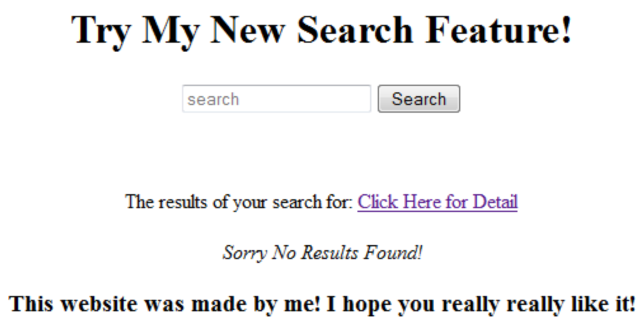
Figure 6 Content of the website converted to italic

Output: Comparing it with the original website, the content of the website has been formatted to italic.

c) Script for redirecting to different page

Script used:

"<aonclick="document.location='http://localhost:82/xss/123.php?cookie='+escape(document.cookie);"href="#">Click Here for Detail</a> "



Try My New Search Feature!

The results of your search for: Click Here for Detail

Sorry No Results Found!

This website was made by me! I hope you really really like it!

Figure 7: link for redirection to different page has been included

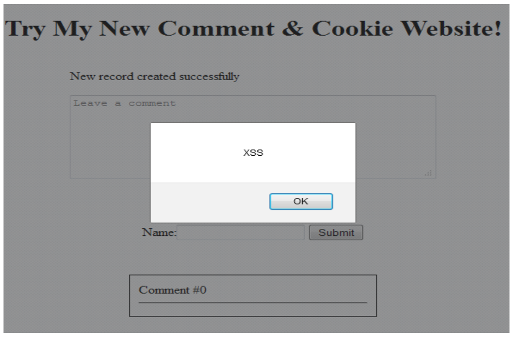
Output: link which redirects to different page has been added on the website.

Stored XSS

1: Checking for Vulnerability towards Stored XSS.

Web server used: XAMPP

Script used: "<script>alert("XSS")</script>



Try My New Comment & Cookie Website!

New record created successfully

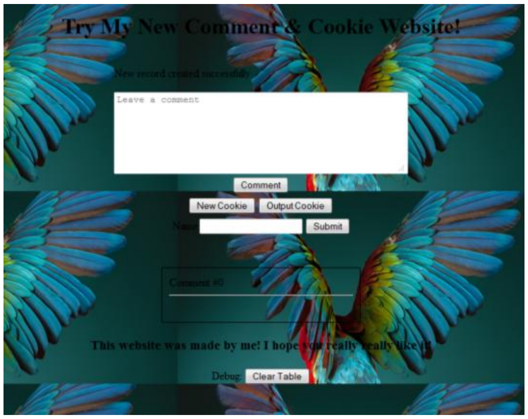
Figure 9 Checking for vulnerability towards stored XSS

Output: On executing the above mentioned script, the pop up message contains the bold words; therefore the particular website is vulnerable towards the stored XSS attack.

2) Script for adding background

Script used: "

<script>document.body.background="http://www.gettyimages.in/gi-resources/images/Homepage/Category-Creative/UK/UK_Creative_462809583.jpg";</script> "



Try My New Comment & Cookie Website!

New record created successfully

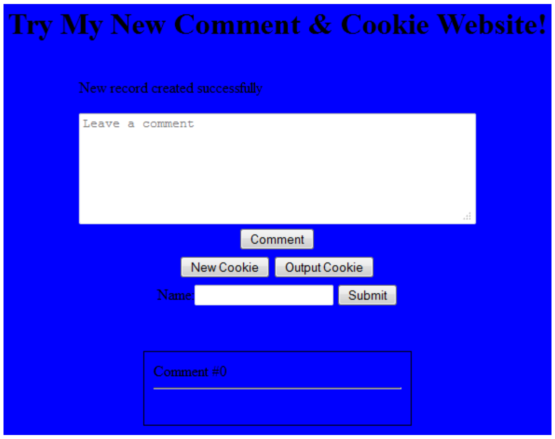
This website was made by me! I hope you really really like it!

Figure 10 Image has been added in the background

Output: Background has been modified by adding an image in the background.

3) Script for changing color of the background

Script used:
“<script>document.body.bgColor="blue";</script>”



Try My New Comment & Cookie Website!

New record created successfully

Figure 11 Background has been changed to blue

Output: Background has been modified by changing the color of the background.

V. TOOLS FOR DETECTING XSS

Some of the tools used for detection of XSS vulnerabilities are discussed below.

TABLE II TOOLS FOR DETECTING XSS

S. No	Tool Name	Description
1	N- Stalker	It is a web scanning tool which is used for the detection of XSS vulnerabilities and other known attacks.
2	Acunetix	It scans the website to check whether the website is vulnerable to XSS attack or not.
3	Paros	Paros first crawls the entire website and then executes the canned vulnerability scanner test.
4	Hackbar	This tool helps the security analyzer to analyze the security holes faster and easily.
5	XSS ME	It is a Exploit-Me tool which is used for the detection of reflected type XSS.

VI. PREVENTING XSS

There are six rules according to which XSS can be prevented they are discussed below:

TABLE III RULES FOR PREVENTING XSS

Rule	Description	Example
Rule 0	*“ Never insert untrusted data except in allowed location”*	*“ <script>....do not put untrusted data here....</script>”* (directly in a script) *“ <!...do not put untrusted data here....-->”* (inside an HTML comment) *“ <div...do not put untrusted data here...=test/> “* (in an attribute name) *“ <...do not put untrusted data here.../href = “/test”>”* (in a tag name)
Rule 1	*“HTML escape before inserting untrusted data into HTML element content”*	*“ <body>....escape untrusted data before putting here....</body>”* *“ <div>....escape untrusted data before putting here....</div>”*
Rule 2	*“Attribute escape before inserting untrusted data into HTML common attributes”*	*“ <div attr=....escape untrusted data before putting here....>content</div>”* (Inside UNquoted attribute) *“ <div attr=’....escape untrusted data before putting here....’>content</div>”* (Inside single quoted attribute) *“ <div attr=”....escape untrusted data before putting here....”>content</div>”* (Inside double quoted attribute) • Escape all the characters whose ASCII values is less than 256 except alphanumeric characters.
Rule 3	*“JavaScript escape before inserting untrusted data into HTML JavaScript data values”*	• *“<script>alert(‘....escape untrusted data before putting here’) ....</script>”* *(Inside quoted string)* • *“<script>x=‘....escape untrusted data before putting here’) ....</script>”* *(one side of quoted expression)*
Rule 4	*“CSS escape before inserting untrusted data into HTML style property values”*	• *“<style>selector{property: ....escape untrusted data before putting here..;} </script>”property value* • *“<span style = property:....escape untrusted data before putting here..;>text</style>”property value*
Rule 5	*“URL escape before inserting untrusted data*	• *<a href=http://www.abc.com?test=...URL escape untrusted data before*

	into HTML URL parameter values”	*putting here...>link</a>*

VII. CONCLUSION

Cross Site Scripting attack (XSS) is the computer security threat which allows the attacker to get access over the sensitive information, when the javaScript, VBScript, ActiveX, Flash or HTML which is embedded in the malicious XSS link gets executed. In this paper we have briefly discussed about cross site scripting attack and checked whether the site is vulnerable towards the XSS or not, discussed about various tools for examining the XSS vulnerability and summarizes the preventive measures against XSS.

ACKNOWLEDGMENT

We would like to thank Ambedkar Institute of Advanced Communication Technologies and

Research for providing us the environment to carry out the research work successfully.

REFERENCES

[1] Kieyzun, A., Guo. P.J, Jayaraman, K., Ernst, M.D. “Automatic creation of SQL injection and cross-site scripting attacks”, software engineering, IEEE 31st international conference, pp. 199-209, 2009.

[2] Bates, D., Barth, A., Jackson, C., “Regular expressions considered harmful in client-side XSS filters”, WWW '10 Proceedings of the 19th international conference on World wide web, ACM, pp. 91-100, 2010

[3] Shar,L.K., Tan,H.B.K, “Defending against cross site scripting attack”, IEEE computer society, vol. 45, pp.55-62, August 2011.

[4] Gundy, M.V., “Noncespaces: Using randomization to defeat cross-site scripting attacks”,Computer and security, Elsevier, vol. 31, pp. 612-628, June 2012.

[5] Shar,L.K., Tan,H.B.K, Briand, L.C., “Mining SQL injection and cross site scripting vulnerabilities using hybrid program analysis”, 35th International Conference on Software Engineering, IEEE, pp. 642-651, May 2013.

[6] EC-Council (2015, June 15). Cross *Site Scripting Attack, Secure ethical hacking (1st ed).* [Online]. Available:http://www.eccouncil.org.