# Case study on Cross Site Scripting

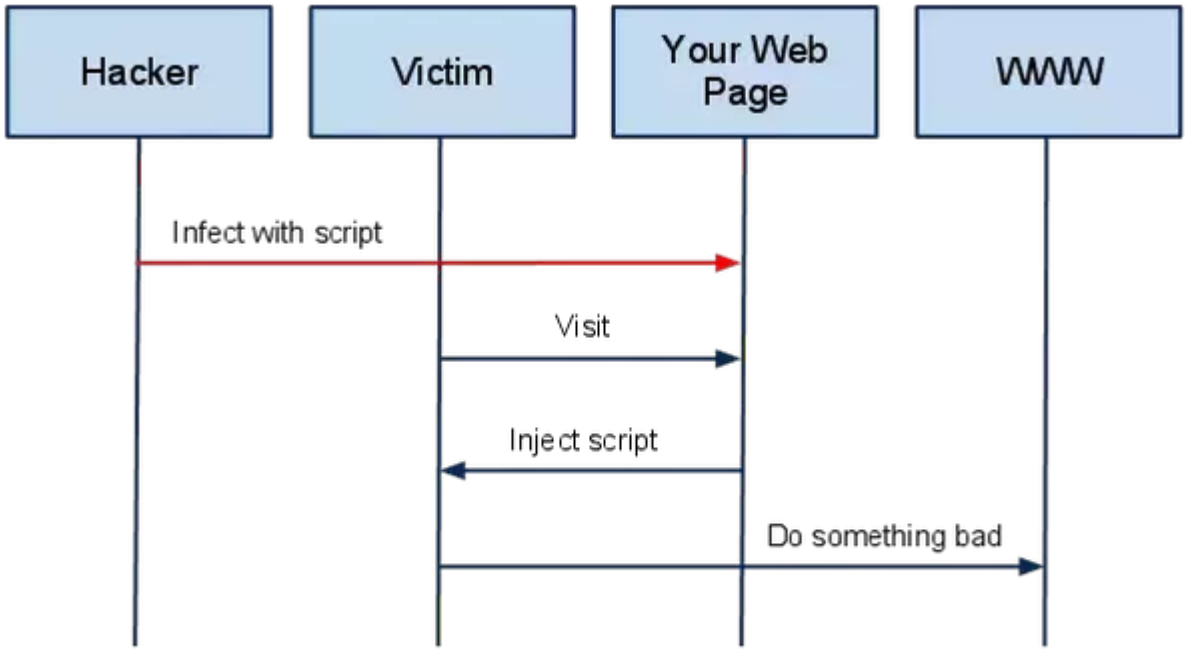
*Name-Kunal.S.Kasodekar*

*RegNo-15BCE1045*

## 1 Name of Cyber Crime:-Cross Site Scripting

### 1.1 Definition :- What is Cross-site scripting (XSS) attack?

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications. XSS enables attackers to inject client-side scripts into web pages viewed by other users. A cross-site scripting vulnerability may be used by attackers to bypass access controls such as the same-origin policy. Cross-site scripting carried out on websites accounted for roughly 84% of all security vulnerabilities documented by Symantec as of 2007. Bug bounty company HackerOne in 2017 reported that XSS is still a major threat vector. XSS effects vary in range from petty nuisance to significant security risk, depending on the sensitivity of the data handled by the vulnerable site and the nature of any security mitigation implemented by the site's owner.




A High Level View of a typical XSS Attack

### 1.2 Basic Goal affected by the Crime

XSS is a versatile attack vector which opens the door to a large number of social-engineering and clientside attacks. As shown, it could be used to steal sensitive information, such as session tokens, user credentials or commercially valuable data, as well as to perform sensitive operations. Additionally, it can be the foothold an attacker needs in order to obtain access to a computer or even an internal network. For companies XSS can have serious implications from a reputational, legal and even financial point of view.As security consultants, we should do our best to explain the risks. In the case of XSS and penetration test reports, it is likely a good time to move away from the traditional proof of concept alert box payload as it can be rather misleading for security stakeholders.

### 2 Cyber Law that addresses the Attack:-

Title 18 of the US Code Section 1030 addresses the federal law with regards to cybercrime, as well as the minimum and maximum sentencing. If you are convicted in a federal court on a violation of 1030 you may find yourself facing 5-20 years in prison, with the maximum looking more likely given the Silk Road conviction. It should be self-evident that having permission to execute these attacks negates the legal issue, so long as systems not outside the control of the tester or company are compromised.

**Sections:**

18 U.S.C. § 1030(a)(2): Computer trespassing, and taking government, financial, or commerce info

18 U.S.C. § 1030(a)(7): Threatening to damage a protected computer

18 U.S.C. § 1030(b): Conspiracy to violate (a)

18 U.S.C. § 1030(c): Penalties

18 U.S.C. § 1030(a)(4): Committing fraud with computer

18 U.S.C. § 1030(a)(5): Damaging a protected computer (including viruses, worms)

## 3 Case Study:-

(1)In the following case study, a Cross-site scripting attack in which phone's contacts are sent to the attacker's server is demonstrated. Let us assume that the user installs Facebook app, which is one of the most popular third party app. The user views the Facebook login page in the WebView. The malicious application fetches the sensitive information from the phone such as user's contact details, phone numbers, Email-id etc., and sends it to the attacker's server through the HttpClient as shown in Figure 2. The attack is very easy to launch but difficult to detect. The user is not aware of such attack as he views only the legitimate content.

(2)One famous case of such attack is the case of Samy Kamkar's XSS exploit of MySpace known as the Kamkar "Samy Worm." His code, using AJAX would then make the attacked user's profile add Kamkar's as a friend, append "but most of all, Samy is my hero" to their pages, and finally copy the same code into the profile that has just been attacked. The result was a work created through Cross-Site Scripting that was spreading to more than one million profiles in 20 hours, and eventually crashed MySpace.

(3) In November 2007 a security flaw was discovered in Canada's passport website. It has allowed easy access to the personal information such as social insurance number, dates of birth, driver's license numbers, etc. A man from Ontario found this flaw after he completed his passport application. He discovered to alter one character in the Internet address displayed by his Web browser, the information displayed was other person's social insurance numbers, driver's license numbers, addresses, and other forms of identification. Once he saw someone else's information he decided to report this cross site scripting vulnerability, instead of exploit it.

## 4 Tools/Methods-

Cross-site scripting attacks may occur anywhere that possibly malicious users are allowed to post unregulated material to a trusted web site for the consumption of other valid users.The most common example can be found in bulletin-board web sites which provide web based mailing list-style functionality.The following JSP code segment reads an employee ID, eid, from an HTTP request and displays it to the user.

```
<% String eid = request.getParameter("eid"); %>
        ...
    Employee ID: <%= eid %>
```

The code in this example operates correctly if eid contains only standard alphanumeric text. If eid has a value that includes meta-characters or source code, then the code will be executed by the web browser as it displays the HTTP response.

Initially this might not appear to be much of a vulnerability. After all, why would someone enter a URL that causes malicious code to run on their own computer? The real danger is that an attacker will create the malicious URL, then use e-mail or social engineering tricks to lure victims into visiting a link to the URL. When victims click the link, they unwittingly reflect the malicious content through the vulnerable web application back to their own computers. This mechanism of exploiting vulnerable web applications is known as Reflected XSS.

## 5 Implications or effects -

The consequence of an XSS attack is the same regardless of whether it is stored or reflected . The difference is in how the payload arrives at the server. Do not be fooled into thinking that a "read only" or "brochureware" site is not vulnerable to serious reflected XSS attacks. XSS can cause a variety of problems for the end user that range in severity from an annoyance to complete account compromise. The most severe XSS attacks involve disclosure of the user's session cookie, allowing an attacker to hijack the user's session and take over the account. Other damaging attacks include the disclosure of end user files, installation of

Trojan horse programs, redirect the user to some other page or site, or modify presentation of content. An XSS vulnerability allowing an attacker to modify a press release or news item could affect a company's stock price or lessen consumer confidence. An XSS vulnerability on a pharmaceutical site could allow an attacker to modify dosage information resulting in an overdose.

**6 Prevention strategies** -

**TOOLS-**

1) **Acunetix**
2) **IBM Rational App Scan**
3) **Burp Suite**

**METHODS-**

It's crucial that you turn off HTTP TRACE support on all webservers. An attacker can steal cookie data via Javascript even when document.cookie is disabled or not supported on the client. This attack is mounted when a user posts a malicious script to a forum so when another user clicks the link, an asynchronous HTTP Trace call is triggered which collects the user's cookie information from the server, and then sends it over to another malicious server that collects the cookie information so the attacker can mount a session hijack attack. This is easily mitigated by removing support for HTTP TRACE on all webservers. A study of majority of web attacks reveals that they are caused due to improper coding of web applications and inability to filter or sanitize input coming into web. The attacks such as XSS and injection attacks occur due to non sanitization of user input. The majority of these web application attacks are mitigated either on the client side or server side. The client side mitigation normally involves input validation techniques. These techniques normally restrict a user from supplying malicious data to the web page. On the other hand, server side mitigation involves filtering the user input or output sanitation. One solution is also to block all JavaScript in your browser but that will restrict the user from developing or viewing interactive web applications .Some researchers have also suggested using browser plug-in that will incorporate some kind of artificial intelligence to restrict or filter user input thereby adding an intelligence factor to browser. Analyzing xss attacks, there are number of client side solutions implemented by the developers all over the world but still to completely secure a web application is still its infancy.

**7 References** –


https://www.incapsula.com/web-application-security/cross-site-scripting-xss-attacks.html

https://en.wikipedia.org/wiki/Cross-site_scripting

https://www.acunetix.com/websitesecurity/cross-site-scripting/

https://www.netsparker.com/blog/web-security/cross-site-scripting-xss/

https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

http://searchsecurity.techtarget.com/definition/cross-site-scripting

https://blog.detectify.com/2015/12/16/what-is-cross-site-scripting-and-how-can-you-fix-it/

https://www.veracode.com/security/xss

https://blog.sucuri.net/2016/04/what-is-an-xss-vulnerability.html