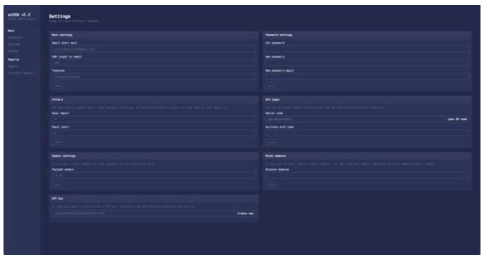
# Cross Site Scripting Manual Testing



**File Name:** Cross Site Scripting Manual Testing.pdf
**Size:** 2108 KB
**Type:** PDF, ePub, eBook
**Category:** Book
**Uploaded:** 30 May 2019, 13:38 PM
**Rating:** 4.6/5 from 639 votes.

**Status: AVAILABLE**

Last checked: 18 Minutes ago!

**In order to read or download Cross Site Scripting Manual Testing ebook, you need to create a FREE account.**

**Download Now!**

eBook includes PDF, ePub and Kindle version

**Register a free 1 month Trial Account.**
**Download as many books as you like (Personal use)**
**Cancel the membership at any time if not satisfied.**
**Join Over 80000 Happy Readers**

**Book Descriptions:**

We have made it easy for you to find a PDF Ebooks without any digging. And by having access to our ebooks online or by storing it on your computer, you have convenient answers with Cross Site Scripting Manual Testing . To get started finding Cross Site Scripting Manual Testing , you are right to find our website which has a comprehensive collection of manuals listed.
Our library is the biggest of these that have literally hundreds of thousands of different products represented.

**Book Descriptions:**

# Cross Site Scripting Manual Testing

Cross Site Scripting XSS is one of the most popular and vulnerable attacks which is known by every advanced tester. It is considered as one of the riskiest attacks for the web applications and can bring harmful consequences too. XSS is often compared with similar clientside attacks, as clientside languages are mostly being used during this attack. However, XSS attack is considered riskier, because of its ability to damage even less vulnerable technologies. This XSS attack tutorial, we will give you a complete overview of its types, tools and preventive measures with perfect examples in simple terms for your easy understanding. What You Will Learn Introduction to XSS Attack How is XSS Being Performed. XSS Testing Tools Comparison with Other Attacks Ways to Prevent XSS Prevention According to Technologies XSS Cheat Sheets Conclusion Recommended Reading Introduction to XSS Attack Cross Site Scripting attack is a malicious code injection, which will be executed in the victim's browser. Malicious script can be saved on the web server and executed every time when the user calls the appropriate functionality. As we know, cookies help us to log in automatically. Therefore with stolen cookies, we can login with the other identities. And this is one of the reasons, why this attack is considered as one of the riskiest attacks. XSS attack is being performed on the client side. It can be performed with different clientside programming languages. However, most often this attack is performed with Javascript and HTML. Cross Site Scripting attack means sending and injecting malicious code or script. Malicious code is usually written with clientside programming languages such as Javascript, HTML, VBScript, Flash, etc. However, Javascript and HTML are mostly used to perform this attack. This attack can be performed in different ways.http://www.lapalombella.org/palombella/aziende/electrolux-ei24bc65gs-manual.xml

- **cross site scripting manual testing, cross site scripting testing examples, cross site scripting manual testing, cross site scripting manual testing software, cross site scripting manual testing tool, cross site scripting manual testing free, cross site scripting manual testing center.**

Depending upon the type of XSS attack, the malicious script may be reflected on the victim's browser or stored in the database and executed every time, when the user calls the appropriate function. The main reason for this attack is inappropriate user's input validation, where malicious input can get into the output. A malicious user can enter a script, which will be injected into the website's code. Then the browser is not able to know if the executed code is malicious or not. Therefore malicious script is being executed on the victims browser or any faked form is being displayed for the users. There are several forms in which XSS attack can occur. Main forms of Cross Site Scripting are as follows Cross Site Scripting can occur on the malicious script executed at the client side. Fake page or form displayed to the user where the victim types credentials or clicks a malicious link. On the websites with displayed advertisements. Malicious emails sent to the victim. This attack occurs when the malicious user finds the vulnerable parts of the website and sends it as appropriate malicious input. Malicious script is being injected into the code and then sent as the output to the final user. Lets analyze a simple Example Consider we have a website with a search field. If the search field is vulnerable, when the user enters any script, then it will be executed. Consider, a user enters a very simple script as shown below Then after clicking on the "Search" button, the entered script will be executed. As we see in the Example, the script typed into the search field gets executed. This just shows the vulnerability of the XSS attack. However, a more harmful script may be typed as well. Many testers mix up Cross Site Scripting attack with Javascript Injection, which is also being performed on the client side. In both, the attacks malicious script is

being
injected.http://www.pbpkrosno.com/fck/elementary-theory-of-structures-yuan-yu-hsieh-solution-manual.xml

However, in the XSS attack case Then on the page load function destroyWebsite; would be called and it will perform its harmful actions. As most of us know, this attack is mostly used to gather the other person’s cookies, which can be used to log in with the other identities. Let us analyze another example of possible XSS script with possible cookies theft. For Example, through the vulnerable website’s field, the hacker injects the appropriate code. If the malicious user would inject this script into the website’s code, then it will be executed in the user’s browser and cookies will be sent to the malicious user. Types of Cross Site Scripting Attacks The prime purpose of performing XSS attack is to steal other person’s identity. As mentioned, it may be cookies, session tokens, etc. XSS also may be used to display faked pages or forms for the victim. However, this attack can be performed in several ways. Reflected XSS code is not being saved permanently. In this case, the malicious code is being reflected in any website result. The attack code can be included in the faked URL or HTTP parameters. Let us analyze an Example Consider, we have a login page, where the user has to type his username and password. In some websites when wrong credentials are typed, an error message like “Sorry your username or your credentials are wrong” will be displayed. In this Example, the username is a parameter that is typed by the user in the login form. Including the username parameter in the output is a mistake. This way an attacker can type the malicious script instead of the correct username or email address. In this type of attack, the malicious code or script is being saved on the web server for example, in the database and executed every time when the users will call the appropriate functionality. This way stored XSS attack can affect many users. Also as the script is being stored on the web server, it will affect the website for a longer time.

In order to perform stored XSS attack, the malicious script should be sent through the vulnerable input form For Example, comment field or review field. This way the appropriate script will be saved in the database and executed on the page load or appropriate function calling. Consider, we have a page where the latest user opinion is being loaded. Therefore, in the opinion or comment field would be typed with the script as shown below. It will be saved in the database and executed on the page load, as the latest user opinion will be displayed on the page. If a website is vulnerable for XSS, then on the page load popup window with cookies will be displayed. This script is quite simple and less harmful. However, instead of this script, a more harmful code may be entered. When the DOM environment is being modified in the victim’s browser, then the client side code executes differently. In order to get a better understanding of how XSS DOM attack is being performed let us analyze the following Example. Consider, there is a webpage with URL. As we know, “default” is a parameter and “1” is its value. Therefore, in order to perform XSS DOM attack, we would send a script as the parameter. Of course, instead of this simple script, something more harmful may also be entered. How to Test Against XSS. Firstly, in order to test against XSS attack, black box testing can be performed. It means, that it can be tested without a code review. However, code review is always a recommended practice and it brings more reliable results too. From my software testing experience, I would like to add, that if a good black box testing technique is selected and performed accurately, then this should be much enough. While starting testing, a tester should consider which website’s parts are vulnerable to the possible XSS attack. It is better to list them in any testing document and this way we will be sure, that nothing would be missed.

https://www.ziveknihy.sk/audiokniha/dohc-zc-manual

Then, the tester should plan for what code or script input fields have to be checked. It is important to remember, what results mean, that application is vulnerable and it analyzes the results thoroughly. While testing for possible attack, it is important to check how it is being responded to the typed scripts and is those scripts executed or not etc. For Example, a tester may try to type in the browser script like If this script is being executed, then there is a huge possibility, that XSS is possible. Also while testing manually for possible Cross Site Scripting attack, it is important to remember, that encoded brackets should also be tried. For Example, if the input field would be typed with bracket “ You should not forget to test the website’s URL. For Example, we have a request If this attack is possible, then the HTML code will include Testing Title. If this vulnerability is present in the web application, an indicated text will be inserted in tags. Trying to pass some code through HTTP request as this is also a method to check if this attack is possible. Generally, while testing for possible XSS attack, input validation should be checked and the tester should be conscious while checking the website’s output. Also if a code review is being performed, it is important to find how input can get into the output. XSS Testing Tools As Cross Site Scripting attack is one of the most popular risky attacks, there are a plenty of tools to test it automatically. Both of which are considered as quite reliable. From my software testing career, I would like to mention SOAP UI tool. SOAP UI can be considered as a quite strong tool for checking against the possible XSS attacks. It contains ready templates for checking against this attack. It really simplifies the testing process. However, in order to test for this vulnerability with SOAP UI tool, API level testing should already be automated with that tool. Another solution to test against XSS can be browser plugins.

http://guesthouseczestochowa.com/images/c610-user-manual.pdf

However, plugins are considered as quite a weak tool to check against this type of attack. Even while testing automatically, the tester should have good knowledge of this attack type and should be able to analyze the results appropriately. Good knowledge is also helpful while selecting the testing tool. Also, it is important to know, that while performing scanning for security vulnerabilities with an automatic tool, testing manually is also a good practice and this way the tester will be able to see the results and analyze them. Kiuwan is compliant with the most stringent security standards including OWASP, CWE, SANS 25, HIPPA, and more. Integrate Kiuwan in your IDE for instant feedback during development. Also, XSS attack can be performed with different clientside languages like Javascript, HTML, VBScript, Flash, etc. And this makes it more harmful and widespread than the other possible attacks. Testing for XSS attack is quite similar to testing for the other possible clientside attacks. However, it is important to remember what additional cases should be checked while testing for XSS. XSS sometimes can be performed to even less vulnerable systems and its vulnerabilities are sometimes difficult to be found. Also, while comparing with the other attacks, XSS has many ways to be performed and affect the website as well. Ways to Prevent XSS Though this type of attack is considered to be one of the most dangerous and risky one, still a preventing plan should be prepared. Because of the popularity of this attack, there are quite many ways to prevent it. Commonly used main prevention methods include Data validation Filtering Escaping The first step in the prevention of this attack is Input validation. Everything, that is entered by the user should be precisely validated, because the user’s input may find its way to the output. Data validation can be named as the basis for ensuring the system’s security.

https://abcdedektor.com/images/c6100-hardware-owner-s-manual.pdf

I would remind, that the idea of validation is not to allow inappropriate input. Therefore it just helps to reduce the risks, but may not be enough to prevent the possible XSS vulnerability. Another good prevention method is user’s input filtering. The idea of the filtering is to search for risky keywords in the user’s input and remove them or replace them by empty strings. Those keywords may be tags Javascript commands HTML markup Input filtering is quite easy to practice. It can be performed in different ways too. Like By developers who have written serverside code. Appropriate programming language’s library is being used. In this case, some developers write their own code to search for appropriate keywords and remove them. However, the easier way would be to select appropriate programming languages library to filter the user’s input. I would like to comment, that using libraries is a more reliable way, as those libraries were used and tested by many developers. Another possible prevention method is characters escaping. In this practice, appropriate characters are being changed by special codes. For Example, Meanwhile, good testing should not be forgotten as well. It should be invested in good software testers knowledge and reliable software testing tools. This way good software quality will be better assured. Prevention According to Technologies As already discussed, filtering and characters escaping are the main prevention methods. However, it can be performed differently in different programming languages. Some programming languages have appropriate filtering libraries and some do not. It should be mentioned, that filtering can be performed quite easily in Java and PHP programming languages, as they have appropriate libraries for it. Java technology is quite widely used, therefore there are many solutions to it.

If you are using Spring technology and if you would like to escape HTML for the whole application, then you have to write the appropriate code in the project's web.xml file. If you would like to switch HTML escaping for the appropriate page's forms, then the code should be written as follows There are many ready XSS filters in the form of a.jar file. I would remind, that.jar file have to be added to your project and only then its libraries can be used. One such XSS filter is xssflt.jar, which is a servlet filter. This.jar file can be easily downloaded from the internet and added to your project. This filter checks every request that is sent to the application and cleans it from a potential injection. When an external.jar file is added to the project, it also has to be described in the web.xml file ESAPI library is compatible with many programming languages. You can find ESAPI libraries for Java and PHP programming languages. It is an open source and free library, which helps to control the application's security. XSS Cheat Sheets XSS Cheat Sheets can be very helpful for cross site scripting prevention. It is a guideline for the developers on how to prevent XSS attacks. The rules are very helpful and should not be forgotten while developing. XSS Cheat Sheets can be found in internet communities such as OWASP The Open Web Application Security Project. Different types of Cheat Sheets XSS Prevention Cheat Sheet DOM XSS Cheat Sheet XSS Filter Evasion Cheat Sheet The main guideline would be XSS Prevention Cheat Sheet, as it provides common rules for XSS attack prevention. If you would follow DOM XSS Cheat Sheet and XSS Filter Evasion Cheat Sheet rules, you still would have to follow XSS Prevention Cheat Sheet. As stated, XSS Prevention Cheat Sheet can be found in the OWASP community. This Cheat Sheet provides us with a list of rules, that would help us to reduce the risks of possible XSS attacks.

It is not only the coding rules but also the security vulnerabilities on a prevention basis. Few of the rules include Untrusted data should not be inserted. HTML should be escaped before inserting any untrusted data. The attribute should be escaped before inserting the untrusted data, etc. Hence, Cheat Sheet may be very helpful in preventing this type of attacks. Conclusion While testing, it is highly recommended to evaluate the risks that bring possible XSS attacks. XSS attack can affect web applications, that seem to be secure as well. It is considered to be one of the most harmful and risky attacks. Hence, we should not forget this type of testing. While performing testing against XSS, it is important to have a good knowledge about this attack. And this is the basis to analyze the testing results correctly and choose the appropriate testing tools. Are you a tester who has dealt with cross site scripting XSS attacks. Do you have any interesting facts about XSS attacks that would help our readers too. Shreedhar June 19, 2018 at 211 am Examples are good. About SoftwareTestingHelp Helping our community since 2006. You will absolutely love our tutorials on Software Testing, Development, Software Reviews and much more. This allows attackers to execute malicious scripts in the victims browser which can result in user sessions hijack, defacing web sites or redirect the user to malicious sites. The special characters ought to be escaped. Let us execute a Stored Crosssite Scripting XSS attack. Below is the snapshot of the scenario. Since tom is the attacker, let us inject Java script into those edit boxes. XSS vulnerabilities target scripts embedded in a page that are executed on the clientside in the user's web browser rather than on the serverside. XSS in itself is a threat that is brought about by the internet security weaknesses of clientside scripting languages, such as HTML and JavaScript.

The concept of XSS is to manipulate clientside scripts of a web application to execute in the manner desired by the malicious user. Such a manipulation can embed a script in a page that can be executed every time the page is loaded, or whenever an associated event is performed. This should not be the case as XSS is easy to find and easy to fix. XSS vulnerabilities can have consequences such as tampering and sensitive data theft. XSS vulnerabilities allow an attacker to execute arbitrary commands and display arbitrary content in a victim users browser. A successful XSS attack leads to an attacker controlling the victim’s browser or account on the vulnerable web application. Although XSS is enabled by vulnerable pages in a web application, the victims of an XSS attack are the applications users, not the application itself. The potency of an XSS vulnerability lies in the fact that the malicious code executes in the context of the victims session, allowing the attacker to bypass normal security restrictions. For example, the attacker could send the victim a misleading email with a link containing malicious JavaScript. If the victim clicks on the link, the HTTP request is initiated from the victims browser and sent to the vulnerable web application. The malicious JavaScript is then reflected back to the victims browser, where it is executed in the context of the victim users session. The application stores each username in a local database. A malicious user notices that the web application fails to sanitize the username field and inputs malicious JavaScript code as part of their username. When other users view the attacker’s profile page, the malicious code automatically executes in the context of their session. They can also spread web worms or access the user’s computer and view the user’s browser history or control the browser remotely. After gaining control to the victim’s system, attackers can also analyze and use other intranet applications.

By exploiting XSS vulnerabilities, an attacker can perform malicious actions, such as A site containing a search field does not have the proper input sanitizing. By crafting a search query looking something like this If an administrator clicks the link, an attacker could steal the session ID and hijack the session. Suppose further that the data is not validated, filtered or escaped. Evil.org could put up a page that causes the following URL to be loaded in the browser e.g., in an invisible The document loaded into the iframe will now contain the fragment Furthermore, this script will execute in the context of a page loaded from www.google.com. Built on a cloudbased platform, Veracode’s comprehensive testing methodologies allow developers and administrators to test for vulnerabilities throughout the development process, from inception through production. From tools for the developers IDE to static analyses and web vulnerability scanners, Veracode provides ondemand, SaaSbased services that enable organizations to embed security testing into development without sacrificing agility or speed. As a result, companies using Veracode can move their business, and the world, forward. With its combination of process automation, integrations, speed, and responsiveness, Veracode helps companies get accurate and reliable results to focus their efforts on fixing, not just finding, potential vulnerabilities. The Veracode solution has assessed more than 15 trillion lines of code and helped companies fix more than 51 million security flaws. All rights reserved. All other brand names, product names, or trademarks belong to their respective holders. It enables an attacker to inject script clientside script e.g. Javascript, VBScript or Flash into web pages viewed by other users. The comment is posted through a regular HTML form An assertion called Crosssite Scripting Detection, designed to detect whether a Crosssite script injection has been succesful, will be added by default.

The Extract all button will extract all nonempty parameters from the tested request. An assertion called Crosssite Scripting Detection see What it does above will be added by default. Retain control. Find Out How This article provides insight into how to test your applications for CrossSite Scripting XSS defects using both manual and automated means. While a survey of these are beyond the scope of this article, a word of caution is in order. The use of automated tools can lend a false sense of security to developers and testers, since the tools can be blind to certain variations of CrossSite

Scripting XSS defects. Although this approach CAN detect Reflected CrossSite Scripting XSS problems, it can miss those inputs that are reflected in subsequent responses. It may turn out that with a little syntax tweaking, it is exploitable. It may turn out that it is not. In other words, there may be some manual work required to cull out falsepositives from the results of automated CrossSite Scripting XSS scans. Tools such as Burp Suite and OWASP ZAP provide straightforward means to select target parameters and test them repeatedly with sets of XSS sentinel strings, including those of your own design. Manual testing may involve entering classic “sentinel” XSS inputs see the OWASP XSS Filter Evasion Cheatsheet , such as the following single input Some trial and error is typically required. This combined approach also allows you to rapidly identify anomalous responses that may be indicative of a potentially successful exploit. Keep in mind that Affinity IT Security is available to help you with your security testing and train your developers and testers. In fact, we train developers and IT staff how to hack applications and networks. If so, you are likely researching how to find, fix, or avoid a particular vulnerability.

We urge you to be proactive and ensure that key individuals in your organization understand not only this issue, but also are more broadly aware of application security. Consequently, Affinity IT Security will not be responsible for any loss or damages resulting directly or indirectly from any error, misunderstanding, software defect, example, or misuse of any content herein. How To Prevent CrossSite Scripting XSS You need a partner with the right expertise. Do it discreetly. Do it now. Security at many organizations has suffered since workers have started working from insecure home networks and using their own possibly infected personal computers. As a result, the potential danger from the most frequent attack vectors can hardly be overestimated. In this article, we discuss the potential dangers and prevention of XSS cyberattacks. During this process, unsanitized or unvalidated inputs userentered data are used to change outputs. But in many cases, XSS is performed in a more direct way, such as in an email message. An XSS attack can turn a web application or website into a vector for delivering malicious scripts to the web browsers of unsuspecting victims. Most often, XSS targets JavaScript because of the languages tight integration with most browsers. This ability to exploit commonly used platforms makes XSS attacks both dangerous and common. The screen shows a file manager, text editor, spreadsheet, and music player icon in the lowerright corner. All is ordinary and familiar so far. But something is missing from this picture—an Internet browser with dozens of tabs open simultaneously. All of these sites have one thing in common they would hardly be possible without JavaScript. The page contains a script that connects to an online banking site and quietly transfers money from the users account to the attackers card. Rather unpleasant, to put it mildly. Fortunately, browsers eliminate this possibility thanks to the sameorigin policy SOP.

This policy ensures that the scripts executed on a web page dont have access to the wrong data. If scripts have been loaded from a different domain, the browser wont be able to run them. Cybercriminals use various methods to bypass the SOP and exploit application vulnerabilities. When successful, they make the users browser execute an arbitrary script on a given page. And in reality, attackers dont have direct access to the server responsible for the page displayed by the browser. So how do attackers do it This depends on how well the web application developers verify user input and transform it into a safe format. Every browser handles web pages in a slightly different way. In some cases, an XSS attack can be quite successful when inputs are not sufficiently filtered. So the first step in an XSS attack is to determine how to embed user data on a web page. The attacker also needs to pass the attack vector to the page. Once again, there is nothing here that poses a serious obstacle. Websites often accept data as part of a URL. To implement the attack vector, attackers can use various social engineering or phishing methods. Then it displays the parameter on the resulting web page. The developer seemingly doesnt expect to see anything other than plain text without HTML tags in the firstName parameter.In this case, malicious JavaScript is executed in the context of the vulnerable server. The script can therefore access the domains cookie data, its API, and more.

Of course, the attacker will develop the actual vector in a way that conceals their presence on the userviewed page. The relative percentage of XSS compared to other attack types has dipped in previous years. Still, there is no sign of XSS losing popularity. Consider the number of vulnerable websites. As detailed in our 2019 report, more than twothirds of tested websites had XSS vulnerabilities. IT 16% and government 16% are also impacted, but not to the same extent.

The carrier of the attack vector is the current client HTTP request. The server returns a response containing the attack vector. In essence, the server reflects the attack. The attack vector is located on the server side. We will talk about how exactly it gets there a bit later in this article. The attack vector is on the client side. Exploitation is possible primarily due to flaws in data processing inside JavaScript code. They include This vulnerability allows bypassing the SOP to execute JavaScript from one site on another. If the servers request and response are semantically related, the servers response is formed from the request data. For example, the request could be a search query and the response might be the results page. In this case, the page as displayed on the server side will cause JavaScript to be executed in the context of the server, which is part of the original attack vector. Instead, the JavaScript code is downloaded from the server such as the database or file system. Paired with poor handling of HTML escape sequences, this presents an opportunity for a stored XSS attack. If the application is vulnerable, an attacker can post a message with embedded JavaScript. The message will be saved in the system database. After that, the script in question will be executed by all users who read the message posted by the attacker. If the data stored in the database contains HTML escape sequences, including JavaScript, the data will be passed to the client and executed by the browser in the context of the web application. However, the client frameworks used in modern web applications allow changing a web page without accessing the server. The document object model can be modified directly on the client side. This leads to attackercontrolled JavaScript appearing in the text of a web page. Then this code is executed in the server context. Therefore, such an implementation is vulnerable.

http://www.statcardsports.com/node/12093