

Forum:[Hack Forums](#)

Thread: [NEWB XSS TUTORIAL //PICTURES INCLUDED / New vulnerable Site Used](#)

madmax889 - August 17, 2012, 7:11 pm

References:

<http://www.breakthesecurity.com/2011/10/cross-site-scriptingxss-complete.html>

<http://hackers.org/xss.html>

<http://centricle.com/tools/ascii-hex/>

Today i'm going to show you how to find and exploit xss (cross site scripting) vulnerabilities.

For our example we will be using this site. www.huhmagazine.co.uk/

Note: This vulnerability is new and working and I have not done anything with it except use it for this tutorial. Feel free to hack this site using my tutorial .

What is XSS?

Cross Site Scripting also known as XSS , is one of the most common web application vulnerability that allows an attacker to run his own client side scripts(especially Javascript) into web pages viewed by other users. In a typical XSS attack, a hacker inject his malicious javascript code in the legitimate website . When a user visit the infected or a specially-crafted link , it will execute the malicious javascript. A successfully exploited XSS vulnerability will allow attackers to do phishing attacks, steal accounts and even worms.

Types of XSS Based on persisting capability:

Based on Persistence capability, we can categorize the XSS attack into two types namely Persistent and Non-Persistent.

Persistent XSS:

The Persistent or Stored XSS attack occurs when the malicious code submitted by attacker is saved by the server in the database, and then permanently it will be run in the normal page.

For Example:

Many websites host a support forum where registered users can ask their doubts by posting message , which are stored in the database. Let us imagine , An attacker post a message containing malicious javascript code instead. If the server fail to sanitize the input provided, it results in execution of injected script. The code will be executed whenever a user try to read the post. If suppose the injected code is cookie stealing code, then it will steal cookie of users who read the post. Using the cookie, attacker can take control of your account.

Non-Persistent XSS:

Non-Persistent XSS, also referred as Reflected XSS , is the most common type of XSS found now a days. In this type of attack, the injected code will be send to the server via HTTPrequest. The server embedd the input with the html file and return the file(HTTPResponse) to browser. When the browser executes the HTML file, it also execute the embedded script. This kind of XSS vulnerability frequently occur in search fields.

XSS Attack:

Step 1: Finding Vulnerable Website

Hackers use google dork for finding the vulnerable sites for instance " inurl:search.php?q= "

Step 2: Testing the Vulnerability:

First of all, we have to find a input field so that we can inject our own script, for example: search box, username,password or any other input fields.

Input this script into the search or comment box. `<script>alert('xss')</script>`. This should give you a pop up showing "xss"

Our example: Popup

Spoiler ([Click to View](#))

[IMG]<http://img831.imageshack.us/img831/6092/xss1new.png>[/IMG]

Once we input a script and we are able to see the alert , such as "xss" popup. We next need to check the page source to see if the vulnerability is persistent or non persistent.

To do this right click >"View page source".

Our example: page source:

Spoiler ([Click to View](#))

[IMG]<http://img521.imageshack.us/img521/8782/xsspagesourcenew.png>[/IMG]

As you can see our input has not been changed in any way. It is just sitting in there just as we had entered it.

This means that the website in our example is not sanitizing our input.

However if we want persistent xss, we need to find a place where our inputted data would normally be kept on the site for good. An example would be a comments section (every users who vists the site will see our comment and if we input a script into this section while we know it is not being changed in any way we know that our script will run for every person visiting the site.

Our example:Comments section

Spoiler ([Click to View](#))

[IMG]<http://img688.imageshack.us/img688/3676/commentsnotsource.png>[/IMG]

Again we can check to see if input into this field is being sanitized, or changed to avoid malicious code from being run.

Our example:Comments section, page source.

Spoiler ([Click to View](#))

[IMG]<http://img607.imageshack.us/img607/4457/commentssectionnew.png>[/IMG]

Once you find a comment section or other input field and you notice that the vulnerability is persistent you can move on to uploading a shell and using it for any purpose you like (such as defacing ?)

```
<script>alert('xss')</script>
```

Now if you tried the above script in a search box or comments section and it did not produce a pop up then it is more than likely our input is being filtered to avoid XSS attacks from being run.

If this is the case we need to try to bypass the filters and try again.

Another good resource in addition to the examples of bypassing filters below can be found here:

<http://hackers.org/xss.html>

Sometimes, website owner use XSS filters(WAF) to protect against XSS vulnerability.

For eg: if you put the `<script>alert("hi")</script>`, the Filter will escape the " (quote) character, so the script will become

```
<script>alert(>xss detected<)</script>
```

Now this script won't work. Likewise Filters use different type of filtering method to give protection against the XSS. In this case, we can use some tricks to bypass the filter. Here I am going to cover that only.

1. Bypassing magic_quotes_gpc

Spoiler ([Click to View](#))

The `magic_quotes_gpc=ON` is a PHP setting (configured in `PHP.ini` File), it escapes the every ' (single-quote), " (double quote) and \ with a backslash automatically.

For Eg: `<script>alert("hi");</script>` will be filtered as `<script>alert(\hi\)</script>`. so the script won't work now.

This is well known filtering method, but we can easily bypass this filter by using ASCII characters instead.

For Eg: `alert("hi");` can be converted to `String.fromCharCode(97, 108, 101, 114, 116, 40, 34, 104, 105, 34, 59)`

so the script will become `<script>String.fromCharCode(97, 108, 101, 114, 116, 40, 34, 104, 105, 34, 59)</script>`. In this case there is no " (quotes) or ' (single quotes) or / so the filter can't filter this thing. Yes, it will successfully run the script.

`String.fromCharCode()` is a javascript function that converts ASCII value to Characters.

How to convert to ASCII values?

There are some online sites that convert to ASCII character. But I suggest you to use Hackbar Mozilla add-on.

After installing hackbar add-on, press F9. It will open the small box above the url bar. click the XSS->`String.fromCharCode()`

Now it will popup small window. enter the code for instance `alert("Hi")`. click ok button. Now we got

the output.

copy the code into the `<script></script>` inside and insert in the vulnerable sites

For eg:

`hxxp://vulnerable-site/search?q=<script>String.fromCharCode(97, 108, 101, 114, 116, 40, 34, 104, 105, 34, 41, 59)</script>`

2. HEX Encoding

Spoiler [\(Click to View\)](#)

we can encode our whole script into HEX code so that it can't be filtered.

For example: `<script>alert("Hi");</script>` can be convert to HEX as:

`%3c%73%63%72%69%70%74%3e%61%6c%65%72%74%28%22%48%69%22%29%3b%3c%2f%73%63%72%69%70%74%3e`

Now put the code in the vulnerable site request.

For ex:

`hxxp://vulnerable-site/search?q=%3c%73%63%72%69%70%74%3e%61%6c%65%72%74%28%22%48%69%22%29%3b%3c%2f%73%63%72%69%70%74%3e`

Converting to HEX:

This site will convert to hex code: <http://centricle.com/tools/ascii-hex/>

3. Bypassing using Obfuscation

Spoiler [\(Click to View\)](#)

Some website admin put the script, alert in restricted word list. so whenever you input this keywords, the filter will remove it and will give error message like "you are not allowed to search this". This can be bypassed by changing the case of the keywords (namely Obfuscation).

For eg: `<ScRipt>ALeRt("hi");</sCRipT>`

This bypass technique rarely works but giving a trial is worth.

4. Closing Tag

Spoiler [\(Click to View\)](#)

Sometimes putting ">" at the beginning of the code will work.

`"><script>alert("Hi");</script>`

This will end the previous opened tag and open our script tag.

Example:

`hxxp://vulnerable-site/search?q="><script>alert("Hi");</script>`

Thanks for reading my small tutorial.

Hikki - August 23, 2012, 8:59 pm

Really nice tutorial. Thanks for this man.

haczord - August 23, 2012, 9:23 pm

good one,nice tutorial thanks for this thread

qurfy - September 19, 2012, 8:25 am

Thanks for the tut , 35char was it not sure blah blah blah

Dinouli98 - September 19, 2012, 8:41 am

man i have one Q

i have tried more than 100 sites and none of them was vulnerable!!

What do i do wrong? :(

Mr.Phoenix - September 19, 2012, 10:43 am

Hmmmm :) nice tutorial man for noobs :) :) keep it up buddy :)

Hmmmm :) nice tutorial man for noobs :) :) keep it up buddy :)

Brezzy - July 12, 2014, 11:50 am

Nice ,im gonna try this wish me luck

canadaboy25 - July 16, 2014, 10:28 pm

Thanks for the tut.

Gonna try this out

Kitty - July 17, 2014, 4:01 am

OK I never did XSS but I wanna ask.. is it possible to inject a permanent iframe code on xss vuln site? Will it affect only the page you injected it on or the whole website? If you have any good tutorials for me to start how to find such vuln sites let me know!! :)

edit: oops, never realized this was such an old topic! D: