

Forum:[Hack Forums](#)

Thread: [\[TuT\] XSS - Behind It All \[TuT\]](#)

zer0pwn - April 6, 2012, 11:29 am

Hello everyone.

I am back once again, with another tutorial. I've been wanting to write this since the minute I learned XSS. My intention is for everyone to learn something useful for their future experiences with Pentesting, or Pure Black-Hat hacking. I'm not responsible for anything you do, this is simply for educational reasons.

Spoiler ([Click to View](#))

I'm supposed to say that shit right?

Anyways.

My tutorial today, is again, on XSS. Behind it all. I will be explaining how everything works. Advanced vectors, and how to make the most of your vulnerability. I've been getting countless messages on how XSS REALLY works. So I decided I'd just write a tutorial on it. First of all, if you don't know anything about XSS, read my tutorial here:

<https://hackforums.net/showthread.php?tid=1838379>

Now. This is the root of EVERYTHING we can execute. Sometimes we'll be limited to what we're capable of. Don't doubt that there will be certain security filters to block our queries and vectors. A lot of the time, it's pretty easy, but sometimes there's some advanced methods or advanced vectors that need to be used to bypass anything that the webmasters put in our way. I like to think of it as a personal challenge when I search for vulnerabilities on a site. A personal challenge between me, and the webmaster.

First of all. Let's look at a simple search bar, okay? Well, here's an example:

Code `<input style="background-color: #fff;" name="q" value="WHAT-WE-SEARCHED-GOES-IN-HERE" class="autocomplete" type="text" size="30">`

So that's our search bar's source once we search for something. Not all searches will be the same. Now, when we execute a script with XSS, it's all through the source. This is an amazing exploit, just because of how it works. If we want to execute a script, we'll have to close the current tag. So it looks like our current one is "input". And it's an event handler, so we don't need to do "</input>" we can do: "">" or ">". Event handlers are different than the basic tag. Here's some example of Event handlers:

Code `
<body onload=alert(1);>
<p onmouseover=alert(1);>
<script src='http://www.google.com/script.js';>
<p onclick=alert(1);>
<p onmouseover=alert(1);>`

I think by now you get the point.

But if we want to close input, we can use: ;> or ">. Personally, the "> is more known, and would probably be more likely to work in this scenario. So let's try a basic vector:

Code "><script>alert(1)</script>

Hit search, and see what happens. If you got an alert box, congratulations you've just performed an XSS attack. Our source should look like this once we've executed our script:

Code <input style="background-color: #fff;" name="q" value=""><script>alert(1)</script>"
class="autocomplete" type="text" size="30">

Do you see how it works? We closed "input" and added our own script in there. Where it says: value="", we closed the quotes so that it wouldn't error. That's why "><script>alert(1)</script> works better than <script>alert(1)</script>

Sometimes we will run into certain problems, or different sources. But it all depends on how good you are at Javascript and HTML. I highly recommend doing some quick reading on Javascript and HTML. I can GUARANTEE it will help you.

Now for persistent XSS. It's pretty much the same idea. But again, depends on your source code. Sometimes they will block quotes, so you either have to find a way to bypass that. Or you can find another page that has a different source so you don't have to use quotes.

Here's an example of a persistent XSS. On a fairly high-profile site aswell. Feel free to check out the source, I'm sure you should be pretty interested.

<http://www.totaljerface.com/profile.php?id=1358278>

That's all for today, I know it's not a whole lot. But this is basically just an explanation of what triggers our XSS in the source. I didn't show very many examples of source code possibilities, but if you have any questions, do NOT hesitate to PM me. I usually get back to people right away :).

The trick, is being familiar with Javascript and HTML. They're not hard to learn the basics at all, so I recommend taking a shot at learning them :) I hope you enjoyed the tutorial :)

Here's some places to learn Javascript and HTML, thanks to Sasori :)

Javascript

<http://w3schools.com/js/default.asp>

<http://www.codecademy.com/>

<https://developer.mozilla.org/en-US/learn/javascript>

HTML

<http://w3schools.com/html/default.asp>

<http://www.html.net/tutorials/html/>

<https://developer.mozilla.org/en-US/learn/html>

Vikta - April 6, 2012, 11:35 am

Just done a similar tutorial though I better like yours tutorial:-)

Also very good explaining!

tomchez - April 6, 2012, 11:37 am

Really good tutorial and easy to follow, good job man.

DCA00X - April 6, 2012, 11:45 am

Sure so, first of all - Nice tut. But secondly when I did started reading... I was thinking... well why the heck am I reading like a non-persistence attack based on XSS.. This might be usefull, but using the new egg method on

Code "><script>alert(1);</script> aint going to make it a fully compromise on the host. If you could show the consumers about how to place an attack vector based on a buffer, which result in a reverse shell... Well then you got it! - that would I take my hat off for.

Anyways, not trying to criticise your thread here, only pointing out ... why the heck Non-persistence!

~ DCA ~

Chris ?ansen - April 6, 2012, 11:50 am

This helped me out a lot! Thanks! Bookmarked! <3

Assertsafe - April 6, 2012, 12:15 pm

I really enjoy reading your tutorials, major props!

Sasori - April 6, 2012, 12:15 pm

Great thread Zer0Pwn.

As you stated, Javascript and HTML would be a great thing to know since it will help you in the long run.

Here are some links for some people to read up on.

Javascript

<http://w3schools.com/js/default.asp>

<http://www.codecademy.com/>

<https://developer.mozilla.org/en-US/learn/javascript>

HTML

<http://w3schools.com/html/default.asp>

<http://www.html.net/tutorials/html/>

<https://developer.mozilla.org/en-US/learn/html>

Feel free to add it in the OP if you choose to.

zer0pwn - April 6, 2012, 12:17 pm

Thanks a lot Sasori, added into the OP :)

VipVince - April 6, 2012, 12:20 pm

Good tut for people to see what the process actually involves. The three steps I follow are:

1. Find out what is being filtered.
2. Distinguish what tags we may currently be inside.
3. Close/open script and deliver vector

lalalox4 - April 6, 2012, 6:24 pm

Hey man, very nice TuT here. Love your vids btw theyre very instructive! thanks!

BeggFoMercy - April 6, 2012, 6:40 pm

Nice work man , Keep it up ;) XD

t o o s h o r t

