Forum:Hack Forums

Thread: How to show that XSS is dangerous

Niely - *September 28, 2014, 3:38 am*

Hello

I have to make a report about a XSS-vulnerability for a (serious) website.
But how do I show them that non-persistent XSS is dangerous?

I already have cookie-stealing, and phising, but how can XSS (non-persistent) also be dangerous on other ways?

How do I show them it is dangerous? (Small codes)

Thanks for reading,
Niely

Jon Snow - *September 28, 2014, 3:39 am*

You can't. All you can do is steal the cookie or redirect them to another page.

Niely - *September 28, 2014, 4:17 am*

^Isn't it possible to do load a PHP code which is stored on another website, with JavaScript?

How bout using your brain - *September 28, 2014, 5:08 am*

What do you mean with load a php code? If sending a request to a php script, then yes it is possible, but what is the point of it? It's basically similar to a redirect.

You can use XSS to steal cookies, make http requests as the victim, phish, make people download some shit, etc.

Niely - *September 28, 2014, 9:15 am*

(Sep 28, 2014 05:08 AM)SirRootALot Wrote: *[CITE]* What do you mean with load a php code? If sending a request to a php script, then yes it is possible, but what is the point of it? It's basically similar to a redirect.

You can use XSS to steal cookies, make http requests as the victim, phish, make people download some shit, etc.
How'd I load a PHP script from my own website to the vulnerable website?
If this code is on my website:
Code <?php
echo "<script>alert(1)</script>;";
?>
How do I launch that on that vulnerable website?

Pillows - *September 28, 2014, 9:16 am*

(Sep 28, 2014 09:15 AM)?Niely Wrote: *[CITE]* How'd I load a PHP script from my own website to the vulnerable website?
If this code is on my website:
Code <?php

echo "<script>alert(1)</script>;";
?>
How do I launch that on that vulnerable website?
For the last part, you can't. What you can do is redirect people to a dangerous website through the XSS.

Soulo - *September 28, 2014, 10:54 am*
If the website has any sort of admin panel/CMS which allows for RCE (via theme editing, arbitrary file upload, etc), then it is completely possible to leverage a XSS attack to RCE in a matter seconds by using an admin's session to send arbitrary requests.

See the following:
[Video: http://www.youtube.com/watch?v=-EGUfPgK_lw]

Niely - *September 28, 2014, 11:00 am*
^Thanks for the interesting video mate! :)
But I don't think there is a CMS in use.

Nitai - *September 28, 2014, 12:58 pm*
You can't, PHP using the server, XSS using the client .

?Criminal - *September 28, 2014, 1:19 pm*
lookup facebook xss PoC a guy made a javascript file to update someones facebook email and steal the chat logs using a basic xss payload so basically anything that javascripts capable which is alot can be used against someone including scraping chat logs and stuff like that(basically sending http requests as the victim like sirootalot said).

P?Nq - *September 28, 2014, 1:45 pm*
(Sep 28, 2014 04:17 AM)?Niely Wrote: *[CITE]* ^Isn't it possible to do load a PHP code which is stored on another website, with JavaScript?No you can not it is not the whole source code since you the have there php db.

cheatson - *September 28, 2014, 1:58 pm*
Meh. Nobody gives a shit about reflective XSS unless they have an established bounty program or you get lucky. If you have to explain why it is dangerous, the odds of getting anything out of it are slim to none is my feeling [Hidden exploit pack attacks; privilege escalation with session surfing and or in conjunction with CSRF/other complementary vulns]

ShadowCloud - *September 28, 2014, 11:31 pm*
The best example I can think of is loading up some js via an external location. The most obvious use for some js like this would be to manipulate the login form...

You simply change the form's action from /somefolder/login.php to evilhost.com/myfolder/mylogger.php. When the user submits this form you record his login details then make a POST request from mylogger.php back to somefolder.login.php so the user will legitimately see a "Incorrect username or password" message, when they try again (assuming finger trouble) they'll be logged in and be none the wiser.

Frosteh - *September 29, 2014, 1:54 am*
The worst security implications of XSS are stealing the cookies of a user and being able to make requests to the server as their identity, in the case of an admin panel or CMS being used that might allow you to gain access to an admin panel you wouldn't otherwise be able to access.

You cannot load remote PHP code onto a site with XSS, PHP is run on the web server before the page is rendered and javascript is run in the clients browser after the page has rendered.

In your example you're trying to load remote PHP which only contains javascript, you can in fact load remote javascript through XSS, simply link a script tag to your hosted script, something like:

<script src="mywebsite.js"></script>

You can use XSS as part of larger attacks however, social engineering and things like that, getting admin credentials using XSS requires you to get a user with an admin account to follow your link and your script to reflect their cookies back to you, then you get into the CMS and possibly upload a shell or make your own admin account, etc.

XSS is more of a risk to user data and phishing than a direct threat to the security of the server itself.

(Sep 28, 2014 01:58 PM)cheatson Wrote: *[CITE]* Meh. Nobody gives a shit about reflective XSS unless they have an established bounty program or you get lucky. If you have to explain why it is dangerous, the odds of getting anything out of it are slim to none is my feeling [Hidden exploit pack attacks; privilege escalation with session surfing and or in conjunction with CSRF/other complementary vulns]
I dunno about that, if you have any kind login system where some users have elevated privileges such admin/mod/author, then you can do a good social engineering attack with reflected XSS by sending them a link to follow, grabbing their credentials/cookies and elevating your own privs.

For example

Quote:Dear support team, I was browsing your website the other day and all the images stopped working, this link doesn't work in firefox or chrome

http://www.mywebsite.com/index.php?id=%3C%73%63%72%69%70%74%20%73%72%63%3D%22%6D%79%65%76%69%6C%73%63%72%69%70%74%2E%6A%73%22%3E%3C%2F%73%63%72%69%70%74%3E

That link is going to get passed back probably to at least a CMS mod/author, and then maybe an admin/developer and god knows how many newbs in between.

Sometimes they'll be savvy enough to URL decode the link first, then browse the remote script first before using the link, most of the time not, that's a good attack vector any security team ought to be wary of.

Niely - *September 29, 2014, 8:44 am*
Okay, thanks all!
Got some ideas from here. :)

If someone still know other things, feel free to post!
The smallest things can help me out.

Niely - *October 4, 2014, 7:58 am*
I used these things to show it's dangerous:
Cookie-stealing
Phishing

Malware download
Browser exploits
Dangrous rejections
...

If someone would matter. :)

Jeremy White - *October 4, 2014, 8:30 am*
I know that you have already noted cookie stealing, but you should really explain how dangerous cookie stealing really can be if used in the correct situation - just saying.

Niely - *October 4, 2014, 8:47 am*
^In my report I give examples and clear explanation of what those things are. :)

c00perzz - *October 5, 2014, 3:42 am*
depending on the website, cookie-stealing can be disastrous

if the authorisation service only depends on the cookie without any other validations etc then

Scenario 1:

1. Steal user cookies
2. If website has forms you can manipulate requests to upload malicious file (shelll) instead of images etc
3. If you have shell you can probably root the server

Scenario 2:

1. Steal ADMIN cookies
2. Upload shell through admin panel
3. root server

keep in mind that not many websites use tokens for authorisation so a very big number of current websites are prone to "disastrous" cookie-stealing

Niely - *October 5, 2014, 4:01 am*
(Oct 5, 2014 03:42 AM)c00perzz Wrote: [CITE] depending on the website, cookie-stealing can be disastrous

2. If website has forms you can manipulate requests to upload malicious file (shelll) instead of images etc
Could you give the JS example of that? I'm really wondering how I'd do that. :)

c00perzz - *October 5, 2014, 5:45 am*
(Oct 5, 2014 04:01 AM)?Niely Wrote: [CITE] Could you give the JS example of that? I'm really wondering how I'd do that. :)
heh , i didnt mean to do that with XSS directly

with XSS you would steal the cookie.

then with the cookie you would login to that user's account and from there you would go to any form that allows uploading files and manipulate POST request to the server (using tamper data or live http headers etc) to upload the shell

Niely - *October 5, 2014, 8:20 am*
(Oct 5, 2014 05:45 AM)c00perzz Wrote: [CITE] heh , i didnt mean to do that with XSS directly

with XSS you would steal the cookie.

then with the cookie you would login to that user's account and from there you would go to any form that allows uploading files and manipulate POST request to the server (using tamper data or live http headers etc) to upload the shell
Ah, that only works if you are able to upload PHP-files.
It needs to have an upload form for that though.

?Criminal - *October 5, 2014, 3:15 pm*
(Oct 4, 2014 07:58 AM)?Niely Wrote: [CITE] I used these things to show it's dangerous:
Cookie-stealing
Phishing
Malware download
Browser exploits
Dangrous rejections
...

If someone would matter. :)
Don't you mean dangerous redirections?

Zabuza? - *October 5, 2014, 8:26 pm*
I believe that you could technically deface a site with XSS by displaying HTML over the rest of the page with a site like C99.nl. The only issue is that it is non persistent, but you could send them the URL with the code in it. If they are stupid enough they will fall for it.

Niely - *October 6, 2014, 8:29 am*
^Yeah probably, but defacing isn't really an issue because it's non-persistent. :)

Abdilo - *October 6, 2014, 4:21 pm*
You could use beef

EPIXIA???? - *October 6, 2014, 6:39 pm*
You can do pretty much everything with reflected xss as you can with persistent. There are some pitfalls like browser protection and you'd have to a bit more social engineering. Anyways, you can always use html5's window.history.pushState to hide the url after it's clicked.

pwntoken - *October 6, 2014, 7:25 pm*
use SOTF, aka Shell of the Future, google it and PM me if you found this useful.

Niely - *October 7, 2014, 11:49 am*
^Thanks a lot!
Never heard of it before, but I'll keep it in my mind for the future.
Would like to add it but the report is already given...

Still looking for those interesting suggestion for in the future!

(Oct 6, 2014 04:21 PM)Grey Hat Mafias Bitch Wrote: [CITE] You could use beef
Yeah, I've included the BeEF framework in browser-exploits. :)