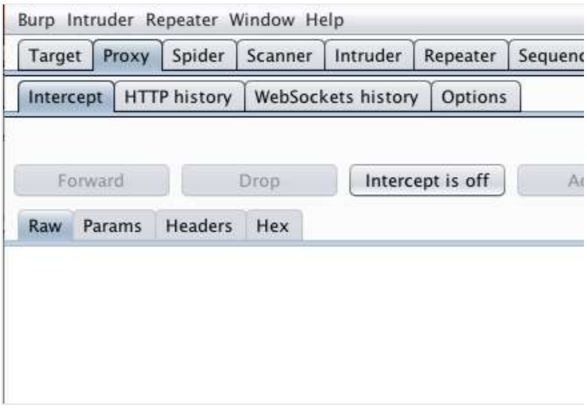
Using Burp to Manually Test for Reflected XSS

Reflected cross-site scripting vulnerabilities arise when data is copied from a request and echoed in to the application’s immediate response in an unsafe An attacker can use the vulnerability to construct a request which, if issued by another application user, will cause JavaScript code supplied by the attack execute within the user's browser in the context of that user's session with the application. The attacker-supplied code can perform a wide variety of actic such as stealing the victim's session token or login credentials, performing arbitrary actions on the victim's behalf, and logging their keystrokes.

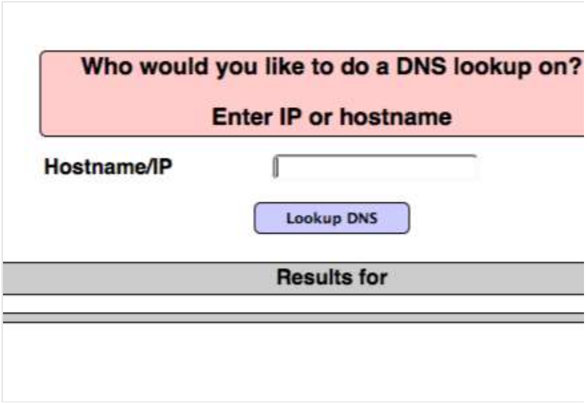
In this tutorial we will demonstrate how to generate a proof-of-concept reflected XSS exploit. The example uses a version of "Mutillidae" taken from OWA Broken Web Application Project. Find out how to download, install and use this project.

First, ensure that Burp is correctly configured with your browser.

With intercept turned off in the Proxy "Intercept" tab, visit the web application you are testing in your browser.

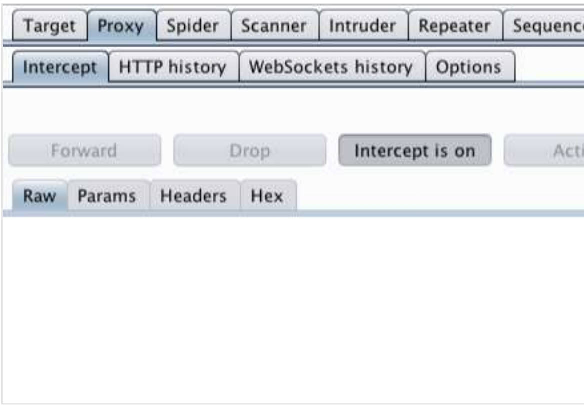

Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequenc
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is off A
Raw Params Headers Hex

Visit the page of the website you wish to test for XSS vulnerabilities.

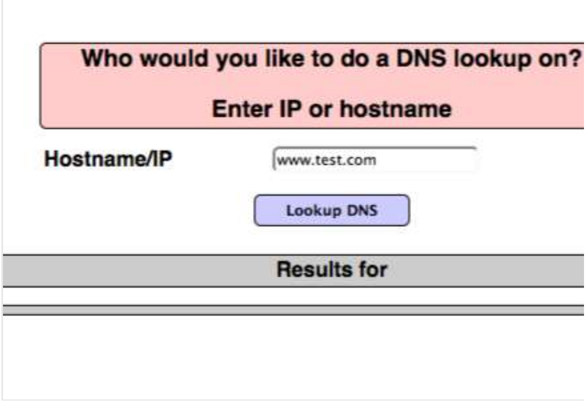


Return to Burp.

In the Proxy "Intercept" tab, ensure "Intercept is on".


Target Proxy Spider Scanner Intruder Repeater Sequenc
Intercept HTTP history WebSockets history Options
Forward Drop Intercept is on Acti
Raw Params Headers Hex

Enter some appropriate input in to the web application and submit the request.



Who would you like to do a DNS lookup on?

Enter IP or hostname

Hostname/IP: www.test.com

Lookup DNS

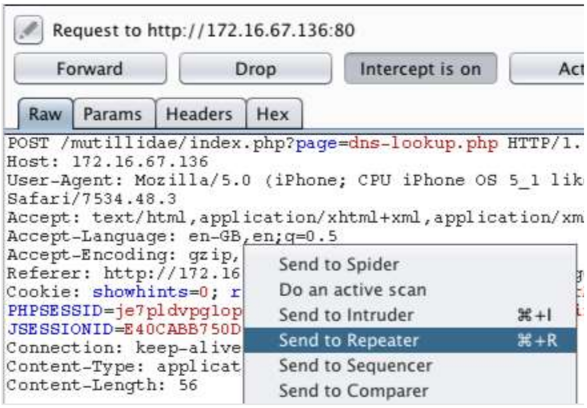
Results for

The request will be captured by Burp. You can view the HTTP request in the Proxy "Intercept" tab.

You can also locate the relevant request in various Burp tabs without having to use the intercept function, e.g. requests are logged and detailed in the "HTTP history" tab within the "Proxy" tab.

Right click anywhere on the request to bring up the context menu.

Click "Send to Repeater"



```
POST /mutillidae/index.php?page=dns-lookup.php HTTP/1.
Host: 172.16.67.136
User-Agent: Mozilla/5.0 (iPhone; CPU iPhone OS 5_1 lik
Safari/7534.48.3
Accept: text/html,application/xhtml+xml,application/xm
Accept-Language: en-GB,en;q=0.5
Accept-Encoding: gzip,
Referer: http://172.16
Cookie: showhints=0; r
PHPSESSID=je7pldvpglop
JSESSIONID=E40CABB750D
Connection: keep-alive
Content-Type: applicat
Content-Length: 56
```

Go to the "Repeater" tab.

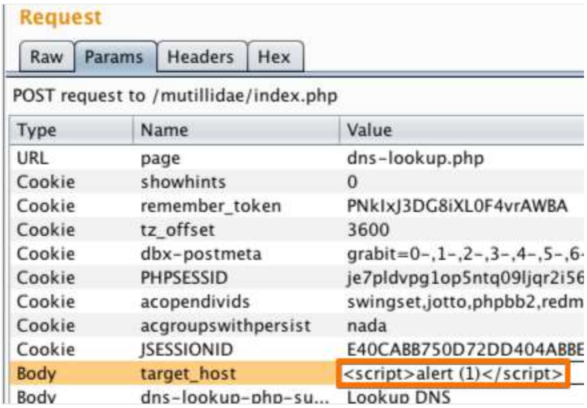
Here we can input various XSS payloads into the input field.

We can test various inputs by editing the "Value" of the appropriate parameter in the "Raw" or "Params" tabs.

A simple payload such as **<s>** can often be used to check for issues.

In this example we have used a payload that attempts to perform a proof of concept pop up in our browser.

Click "Go".



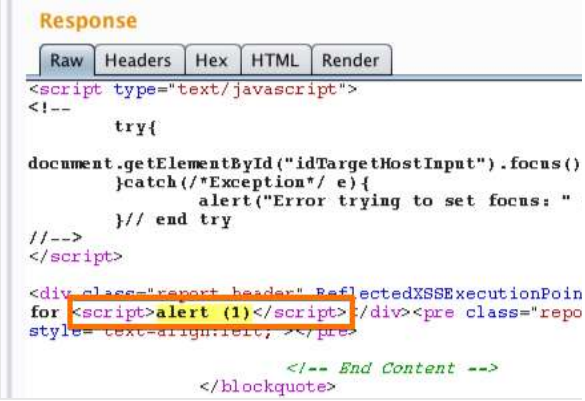
POST request to /mutillidae/index.php

Type	Name	Value
URL	page	dns-lookup.php
Cookie	showhints	0
Cookie	remember_token	PNkIxJ3DG8iXL0F4vrAWBA
Cookie	tz_offset	3600
Cookie	dbx-postmeta	grabit=0-,1-,2-,3-,4-,5-,6-
Cookie	PHPSESSID	je7pldvpg1op5ntq09ljqr2i56
Cookie	acopendivids	swingset,jotto,phpbb2,redm
Cookie	acgroupswithpersist	nada
Cookie	JSESSIONID	E40CABB750D72DD404ABBE
Body	target_host	<script>alert (1)</script>
Body	dns-lookup-php-su...	Lookup DNS

We can assess whether the attack payload appears unmodified in the response. If so, the application is almost certainly vulnerable to XSS.

You can find the response quickly using the search bar at the bottom of the response panel.

The highlighted text is the result of our search.

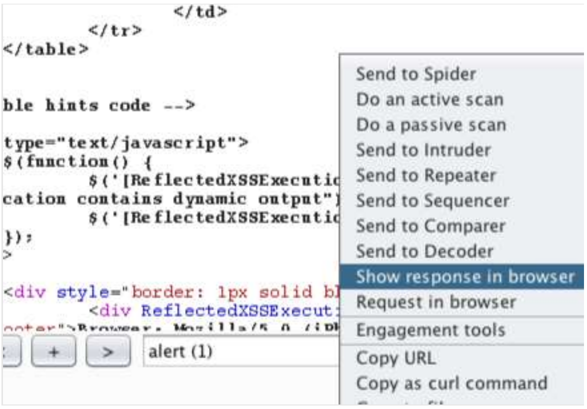


```
<script type="text/javascript">
<!--
        try{

document.getElementById("idTargetHostInput").focus()
        }catch(/*Exception*/ e){
                alert("Error trying to set focus: "
        }// end try
//-->
</script>

<div class="report header" ReflectedXSSExecutionPoin
for <script>alert (1)</script></div><pre class="repo
style="text-align:left;"></pre>

                        <!-- End Content -->
                </blockquote>
```

Right click on the response to bring up the context menu.

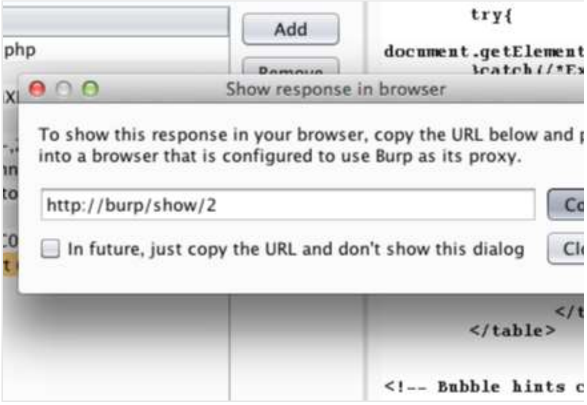
Click "Show response in browser" to copy the URL.

You can also use "Copy URL" or "Request in browser".



```
                    </td>
               </tr>
</table>

ble hints code -->

type="text/javascript">
$(function() {
        $('[ReflectedXSSExecutic
cation contains dynamic output"]
        $('[ReflectedXSSExecutic
});
>

<div style="border: 1px solid bl
        <div ReflectedXSSExecut:
```

In the pop up window, click "Copy".



Show response in browser

To show this response in your browser, copy the URL below and p into a browser that is configured to use Burp as its proxy.

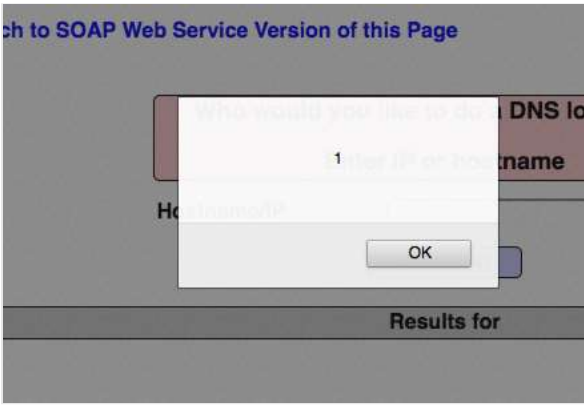
http://burp/show/2

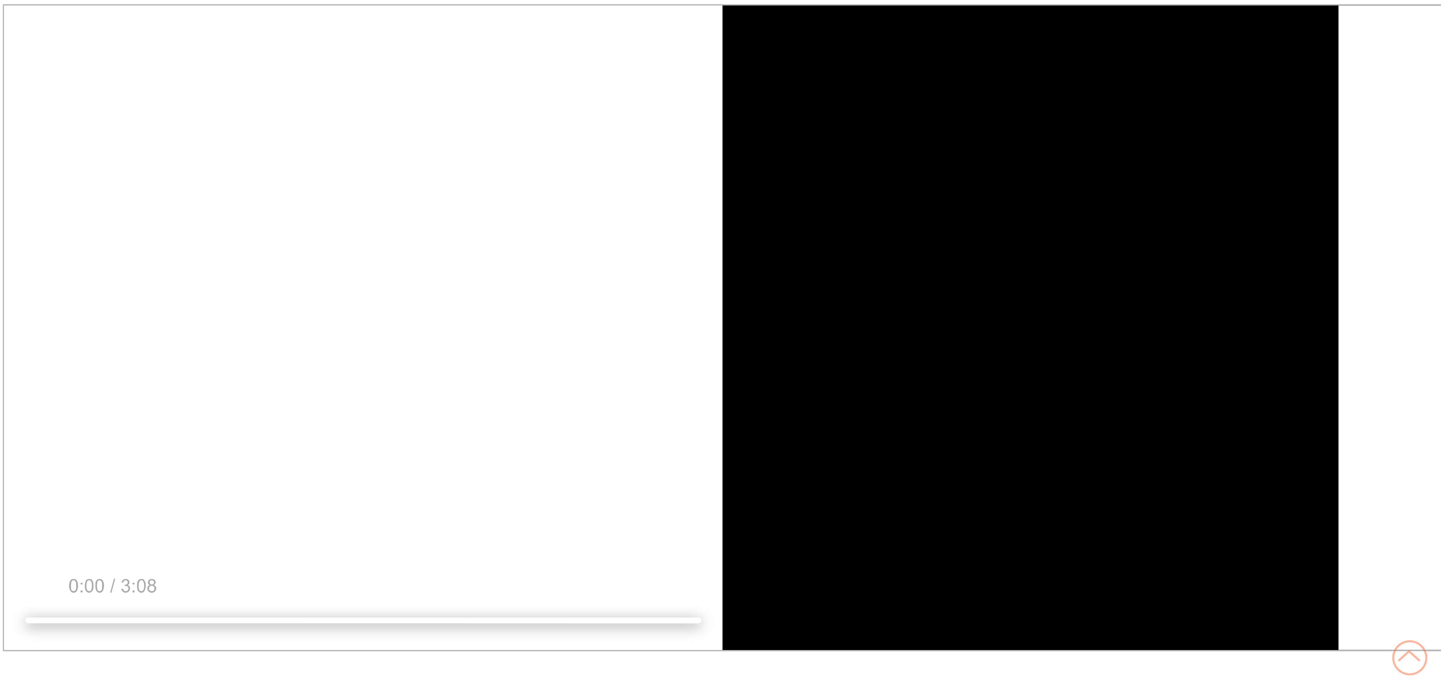
□ In future, just copy the URL and don't show this dialog

```
try{
document.getElement
}catch(/*Ex
</t
</table>

<!-- Bubble hints c
```

Copy the URL in to your browser's address bar.

In this example we were able to produce a proof of concept for the vulnerability.





Related articles:

- Getting started with Burp Proxy
- Using Burp Repeater

Burp Suite

Web vulnerability scanner
Burp Suite Editions
Release Notes

Vulnerabilities

Cross-site scripting (XSS)
SQL injection
Cross-site request forgery
XML external entity injection
Directory traversal
Server-side request forgery

Customers

Organizations
Testers
Developers

Company

About
PortSwigger News
Careers
Contact
Legal
Privacy Notice

Insights

Web Security Academy
Blog
Research
The Daily Swig

Follow us