1. Reflected cross-site scripting

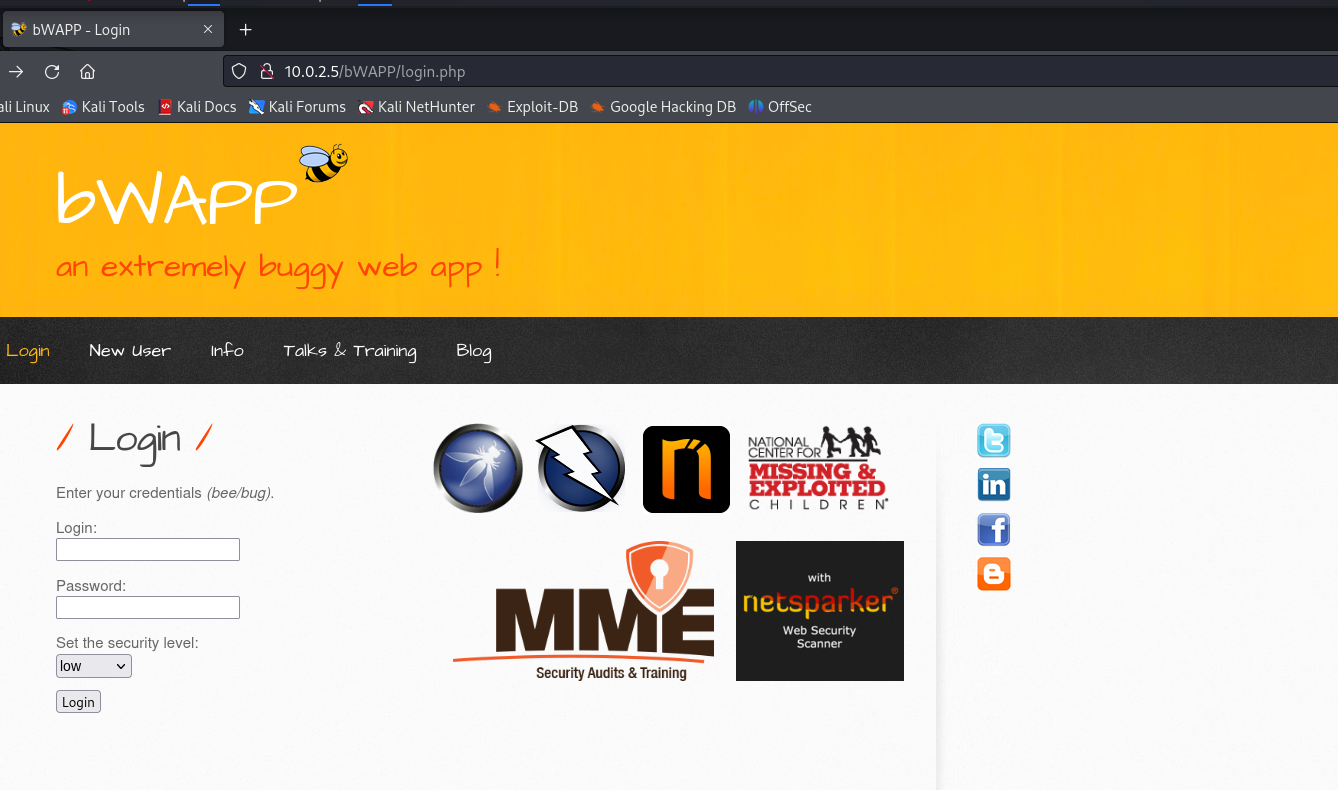
Vulnerable lines of code in xss\_get.php in bwapp



The vulnerability lies in the echo "Welcome " . xss($firstname) . " " . xss($lastname); line. The xss function is supposed to sanitize the input based on the security level, but if the security level is set to "0", the xss function performs no\_check, which means the input is not sanitized at all. This can lead to reflected XSS when the security level is low.

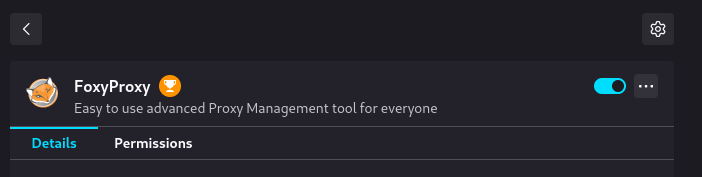
To exploit the vulnerability, the bee box is installed in a virtual box and the host environment is Kali Linux.

So the bwapp from the bee box is accessed by its IP in the kali linux.

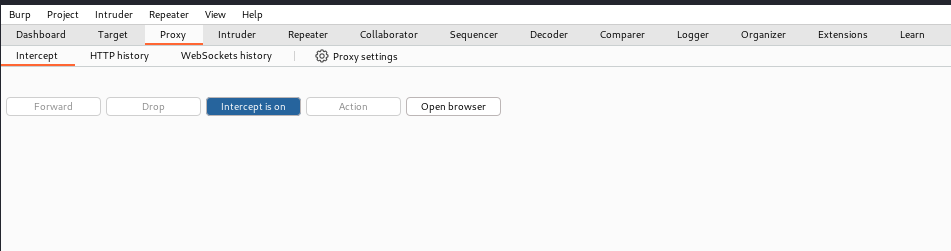


We will use the burp suite community edition to exploit the vulnerability.

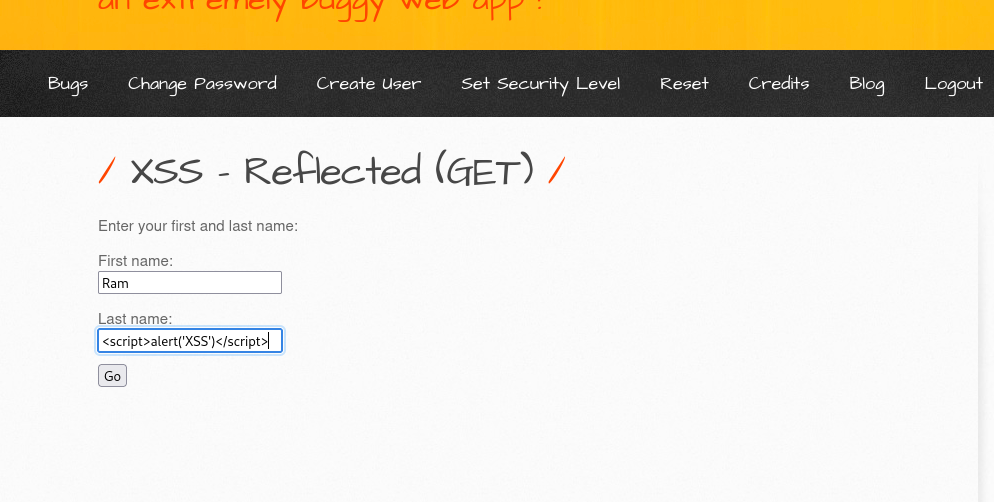
So the extension froxyproxy is turned on to switch internet connection across one or more proxy servers.



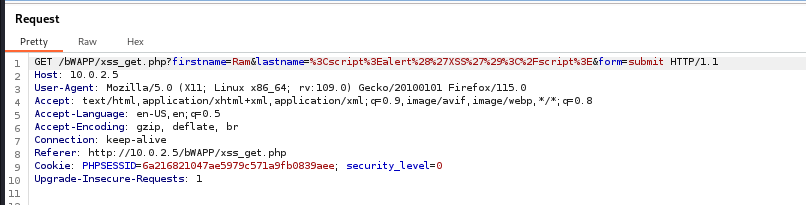
After that in the proxy tab in Burp suite, we will make sure the intercept is on.

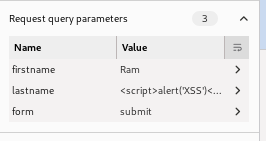


Now the payload is injected in bwapp along with the script in the input field.

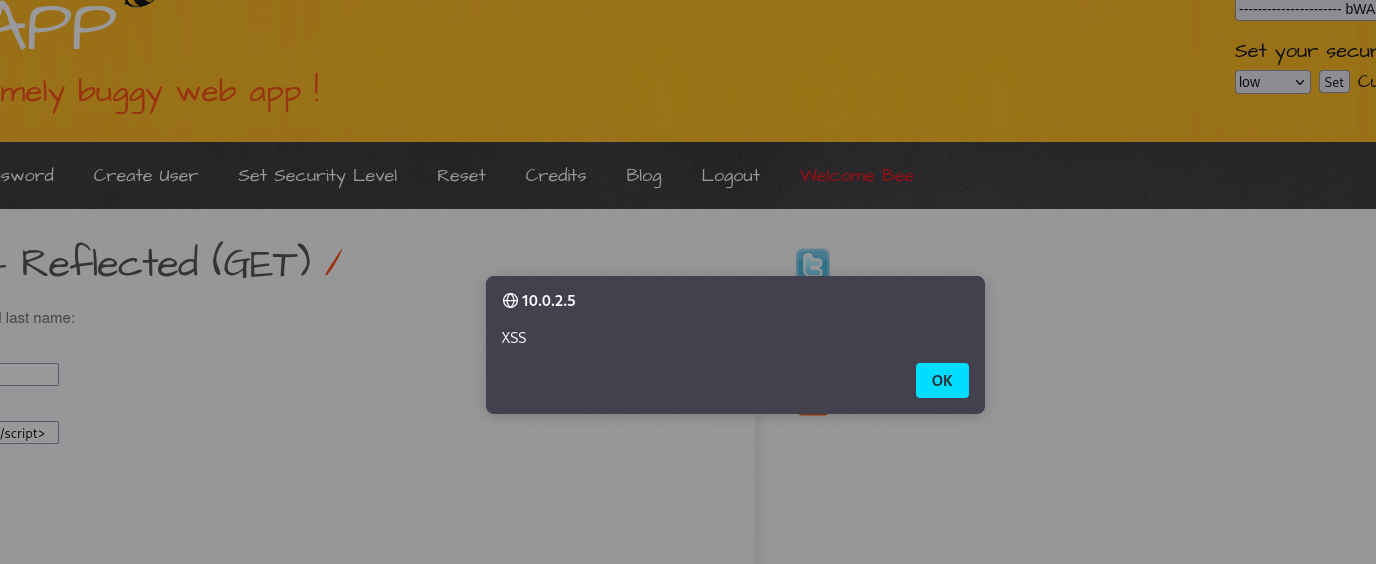


When you submit the form, Burp Suite intercepts the request. The intercepted request may be viewed and analyzed in the HTTP history sub-tab. The payload is one of the URL query parameters.





Now turn off the intercept and see the results in the browser.



The script executes and an alert box with the message “XSS” can be seen. The reflected XSS vulnerability is successfully found.

1. login cross-site scripting

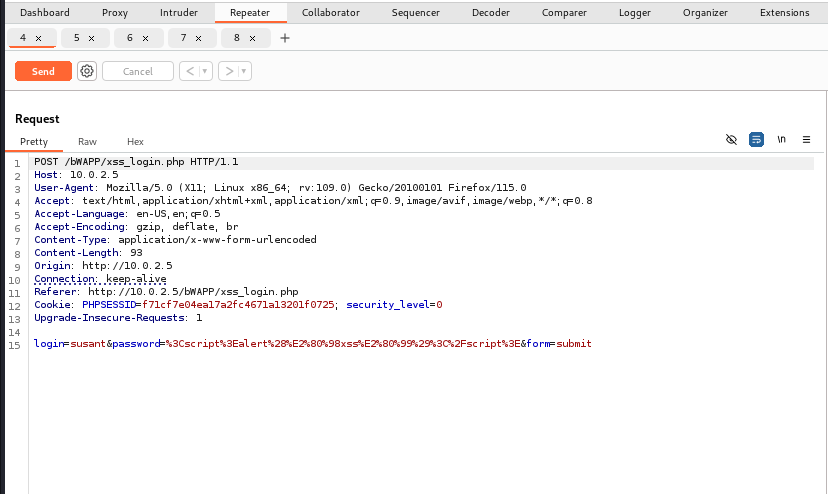




The vulnerability lies in how the $message variable is constructed and echoed back to the user. User-supplied input ($row["login"] and $row["secret"]) is directly concatenated into the HTML response without proper escaping or validation.

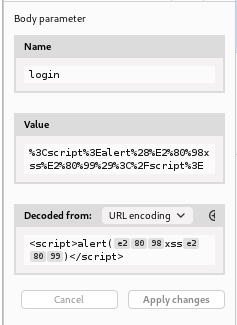


After that, the data in HTTP history is sent to the repeater in burpsuite.

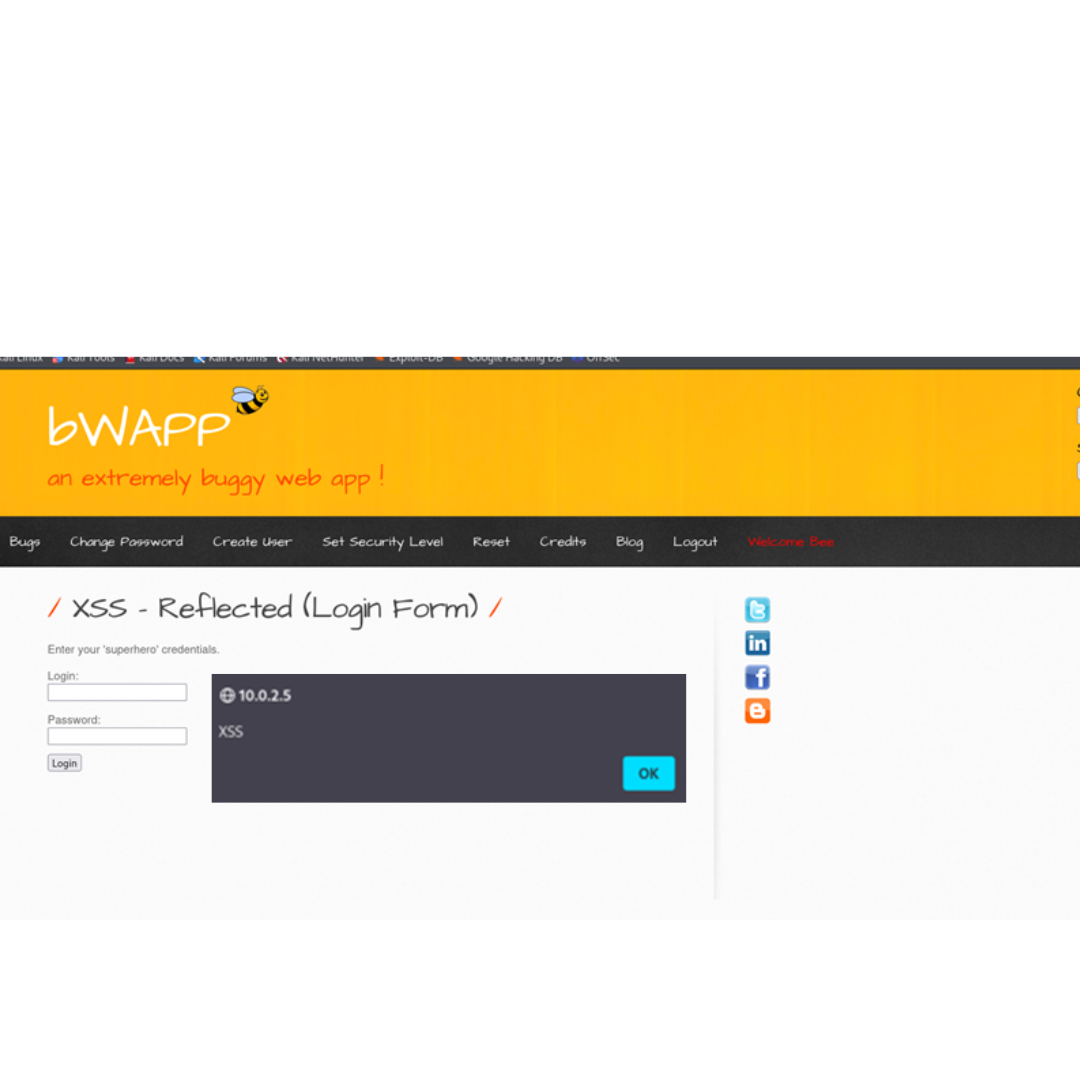


In the response tab of Burp Suite, look for the HTML code that includes your payload.

The <script>alert('XSS')</script> is a part of the response successfully.



Then the URL is copied from the browser and the alert box is generated in the browser.



So, the reflected login XSS vulnerability is successfully found.