**Incident Report 11740: Offensive XSS Attacks**

**03/02/22**

**Executive Summary:**

The TTA Bookstore has multiple XSS vulnerabilities on it. Explore the web app and perform at least two different styled attacks. Write a report including the attacks you performed, and screenshots of the result of each attack.

**Application Details:**  https://ttabookstore.herokuapp.com/#/

**Attack Narrative**

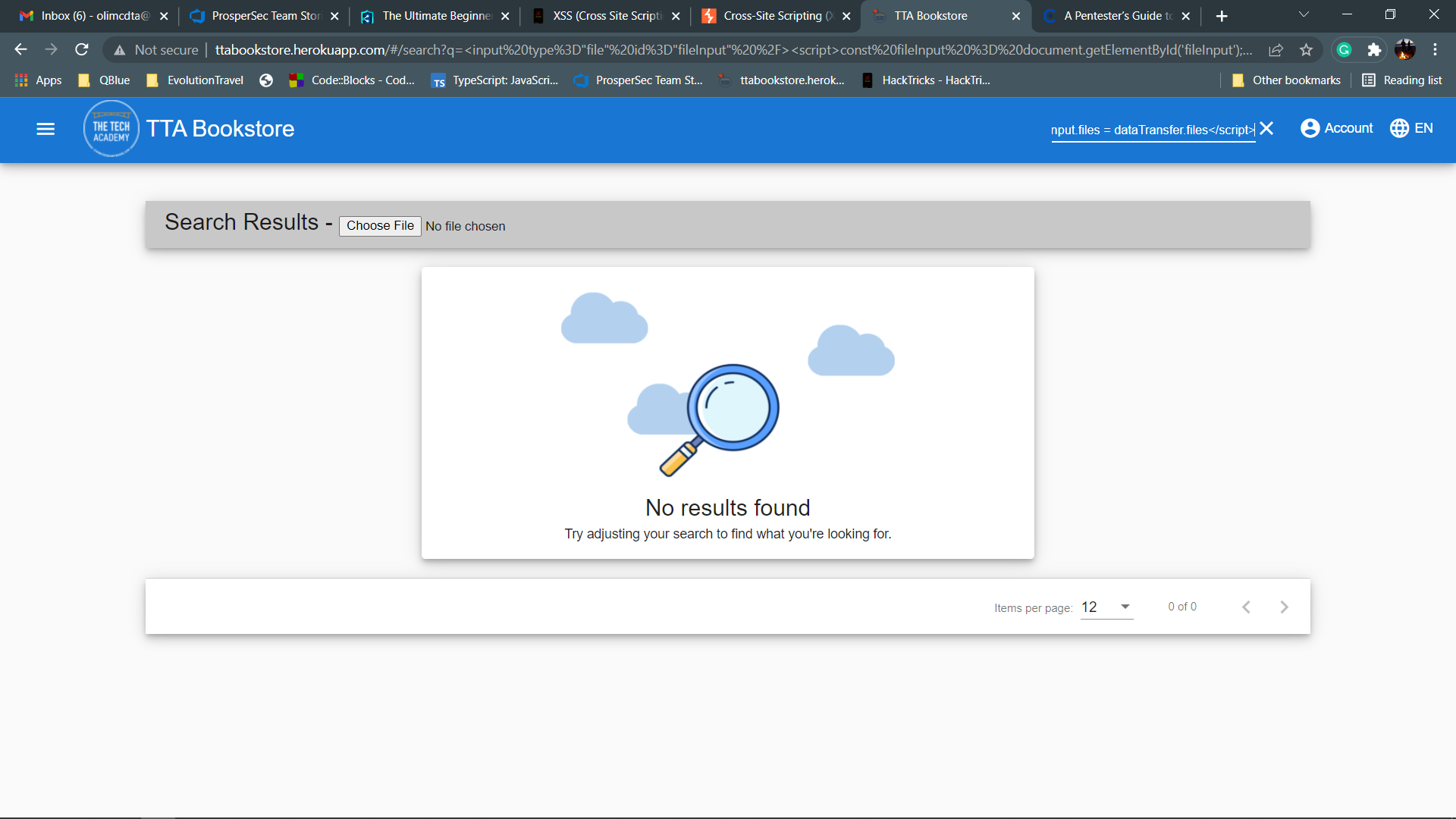
I tried several escape characters in the “search” field. I noticed that the “<” was allowed. But “script” was blacklisted. However “frame” and “input” type were allowed. So I was able to execute the two attacks listed above

**Results**

I Inserted script in th “Search” field that allows a file upload:

Here is the Script:

<input type="file" id="fileInput" /><script>const fileInput = document.getElementById('fileInput');const dataTransfer = new DataTransfer();const file = new File(['Hello world!'], 'hello.txt', {type: 'text/plain'});dataTransfer.items.add(file);fileInput.files = dataTransfer.files</script>

Screenshot of results

I also inserted a script into the “search” field that redirects to another website. In this case, Portswigger.com.

Here is the script:

[<meta http-equiv="refresh" content="0; url=//portswigger-labs.net">](https://portswigger-labs.net/xss/xss.php?x=%3Cmeta%20http-equiv%3D%22refresh%22%20content%3D%220%3B%20url%3D%2F%2Fportswigger-labs.net%22%3E&context=html)

**Conclusion:**

Further sanitisation of the “Search” field is needed to prevent XSS attacks. The creation of White list is recommended.

Here are some other tips:

* Use secure frameworks that, by design, automatically encode content to prevent XSS.
* Coding the unreliable HTTP requirements data into the HTML output fields (body, attributes, JavaScript, CSS, or URL) resolves the Reflected XSS and Stored XSS.
* Sanitize all data entry, avoiding any type of special character.
* Accept all types of data, but escape them correctly.
* Accept all types of data, but remove inappropriate content.
* Accept all types of data, but transform them to an accepted data type.

More resources for securing your website can be found here: https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Cross\_Site\_Scripting\_Prevention\_Cheat\_Sheet.md