Cookies and cross-site scripting (XSS)

Part 1: Cookies

a) There are cookies for this domain. Name: theme, Value: default.

b) Every time the page's theme changes, the theme cookie’s value changes to match the site's current theme.

c) When initially changing the site theme, in the HTTP Request section you see Cookie: theme=default and in the HTTP Response section you see SetCookie: theme=blue; Expires=Thu, 23 Jan 2025 22:05:30. These are the same cookie name and values you can see through the inspect tool.

d) Yes, The site displays my last selected theme. ‘

e) When the user attempts to access the browser, the browser sends an HTTP GET request to the server to retrieve the site information. The site responds to this request with a response containing the site's contents alongside the default theme cookie settings. The browser takes these cookies' values pairs them and stores them somewhere. Later, when the user attempts to access the site, the browser makes the HTTP GET request along with the stored theme cookie to ensure it returns a site containing the saved theme preference set by the user.

f) When the user changes the browser’s theme, the browser saves this preference in a cookie, which is sent to the server through an HTTP request.

g) If you go to the main tag in the inspect tool you can change the theme colors by modifying the class name to one of the following: container (Default), container red (Red), container blue (Blue).

h) When viewing the GET Request shown from the Burpsuite proxy you can change the theme's color by modifying the Cookie: Theme= "color" header.

i) On Windows computers running Google Chrome, cookies are stored in a single file called cookies. This file can be accessed by navigating through the file explorer to the path C:\Users\Your_User_Name\AppData\Local\Google\Chrome\User Data\Default\Network. On Burpsuite all the cookies issued by the sites you visit are stored in a container called the Cookie Jar, which is shared between all the Burp's tools. You manually edit and remove cookies from this container.

Part 2: Cross-Site Scripting (XSS)

a)

i) Moriarty creates a post with CSS and Javascript elements embedded in the message.

ii) This embedded code appears as standard text but is formatted in a way that the site renders and executes it.

iii) Whenever a user clicks on a post created by Moriarty, the CSS and Javascript render as intended.

iv) The nature of Moriarty's attacks is harmless and only adds to the aesthetics of his post.

b) Moriarty can modify or delete the contents of other user's posts, threatening the integrity of users' information and data that they display.

c) Moriarty could remove users from the database, taking away their access to their content.

d) The browser or the server could restrict the usage of certain special characters and require substitution, to ensure no Javascript or CSS commands could be manually inputted.