Nathan Parker and Alden Harcourt

Part 1: Cookies

1. Go to FDF and use your browser's Inspector to take a look at your cookies for cs338.jeffondich.com. Are there cookies for that domain? What are their names and values?
    a. There is one cookie, which is “theme” currently the value is “default”
2. Using the "Theme" menu on the FDF page, change your theme to red or blue. Look at your cookies for cs338.jeffondich.com again. Did they change?
    a. Yes, the value changes to whatever you change the theme to.
3. Do the previous two steps (examining cookies and changing the theme) using Burpsuite. What "Cookie:" and "Set-Cookie:" HTTP headers do you see? Do you see the same cookie values as you did with the Inspector?
    a. I see the “Cookie:” header in one of the HTTP GET request headers, and its value is currently “theme=default”. In one of the response headers, I see the “Set-Cookie:” header, with the value of “theme=red”, along with an expiration date.
4. Quit your browser, relaunch it, and go back to the FDF. Is your red or blue theme (wherever you last left it) still selected?
    a. Yes
5. How is the current theme transmitted between the browser and the FDF server?
    a. Whenever the browser is opened, because of the cookie, the browser sends a request for the theme stored in the cookie.
6. When you change the theme, how is the change transmitted between the browser and the FDF server?
    a. Through an HTTP get request that is sent every time the user updates the setting, and it sets the cookie variable.
7. How could you use your browser's Inspector to change the FDF theme without using the FDF's Theme menu?
    a. You could change the cookie variable’s value
8. How could you use Burpsuite's Proxy tool to change the FDF theme without using the FDF's Theme menu?
    a. You can intercept the GET request in burpsuite, and change the cookie header before sending it back out.
9. Where does your OS (the OS where you're running your browser and Burpsuite, that is) store cookies? (This will require some internet searching, most likely.)
    a. On a mac, they get stored in your user profile folder, on windows, they are in the INetCookies folder in the C: drive.

Part 2: Cross-Site Scripting

1. Provide a diagram and/or a step-by-step description of the nature and timing of Moriarty's attack on users of the FDF. Note that some of the relevant actions may happen long before other actions.

    a. First, Moriarty submits a message to the forum that includes some javascript code that executes some bad thing (in this case making the text red, or a pop up). Next, a person clicks on the message. This initiates a database search for the page that relates to his message. That data (his message with code) is put into the html code for the site to be displayed. The html is then executed, and when it hits Moriarty's message, his imputed script runs, executing his evil plan.
2. Describe an XSS attack that is more virulent than Moriarty's "turn something red" and "pop up a message" attacks. Think about what kinds of things the Javascript might have access to via Alice's browser when Alice views the attacker's post.
    a. A malicious actor could easily write a short bit of code that downloads a file to your computer when you click on the post. That file could contain anything, but would probably be some form of malware.
3. Do it again: describe a second attack that is more virulent than Moriarty's, but that's substantially different from your first idea.
    a. They could redirect you to a fake version of the site, say you need to re-log in, and then steal your username and password from there. If you used your real email, and reused that password on other sites, they could gain access to many different accounts that you own.
4. What techniques can the server or the browser use to prevent what Moriarty is doing?
    a. The server should really clean the input data from all user input to make sure there are no scripts of any kind, in order to prevent the kinds of attacks mentioned here.