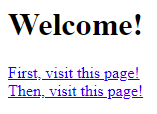
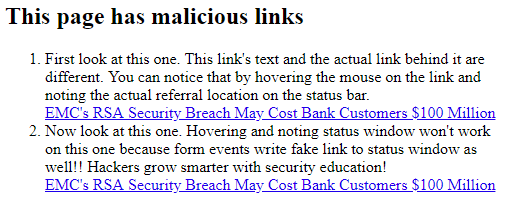
1. Cross Site Scripting (XSS) attack
   1. Host the XSS Web application on your Apache Web server by
2. Copy “XSS.tar.gz” to your VM
3. Extract the .tar.gz file to the appropriate directory of the Apache 2 Web server
4. Properly set the permissions of the extracted directory and files so that webpages in the XSS directory can be properly accessed by anyone through a Web browser, e.g., by typing: <http://192.168.65.128/XSS/> in which that IP address is for my VM.
   1. Visit the XSS Web application using a browser
5. You will see two links as shown below. click the “First, visit this page!” link to visit setgetcookie.htm.



1. On setgetcookie.html, fill the username/password form, click “Set cookie” and then “Show cookie” buttons to make sure this website set a cookie to your browser.
2. Go back and click the “Then, visit this page!” link to visit malURL.htm. You will see two links on this malURL.htm page as shown below, but both of them do not work right now.



1. Change the malURL.htm of your XSS Web application, so that when you click the FIRST “EMC's RSA Security Breach May Cost Bank Customers $100 Million” link, the cookie stored on your browser for this website will be sent to an attacker’s server (that’s my VM) using an HTTP GET request with the following format:

<http://192.168.65.128/XSS/stealcookie.php?info=COOKIES>

Note: you need to use the above address and format with your own IP address but not mine, so that the stealcookie.php on your VM can receive and verify the submission record. Here, COOKIES means the complete cookie content save on your browser for this website (i.e., document.cookie), so it should include a pair of username and password values. The file stealcookie.php is also included in XSS.tar.gz.

Hint: there could be multiple solutions. One solution is to simply replace the current href value ("#") of the first link in malURL.htm with a piece of JavaScript code; nothing else needs to be changed. Note that if you use this solution and if you move the mouse over the link, you may notice that my VM address information is shown on the status bar of your browser. So, a security conscious user may notice this and may not be tricked to click the link.

Question 6: (15 pts)

1. What are your changes to the malURL.htm file? What are your verification results? Explain this XSS attack.
2. Make the attack stealthier, so that when you move the mouse over the second link, my VM address information will not be shown on the status bar of your browser. In this case, a security conscious user may also be tricked to click the second link. Still, when you click this second “EMC's RSA Security Breach May Cost Bank Customers $100 Million” link, the cookie stored on your browser for this website will be sent to an attacker’s server (that’s my VM) using an HTTP GET request with the following format:

<http://192.168.65.128/XSS/stealcookie.php?info=COOKIES>

Note: you need to use the above address and format with your own IP address but not mine just like what you did in the last question.

Hint: there could be multiple solutions. One simple solution is to simply add a

redirectpage.htm webpage in the XSS directory; you don’t need to change malURL.htm

because redirectpage.htm has been specified in it.

Question 7: (15 points)

1. Describe the details of your solution. If you use the redirectpage.htm solution, please provide the content of this file. What are your verification results?
2. Basically, the above two links illustrated the persistent (or stored) XSS attacks. That is, if an attack can store those links on your legitimate webpages or websites, they can perform attacks such as cookie stealing. This XSS Web application is also vulnerable to non-persistent (or reflected) XSS attacks. In this task, demonstrate that the above cookie stealing attacks can happen by using reflected XSS attacks. In other words, reflected XSS attacks can also send the cookie stored on your browser for this website to an attacker’s server (that’s my VM) using an HTTP GET request with the following format:

<http://192.168.65.128/XSS/stealcookie.php?info=COOKIES>

Note: you need to use the above address and format with your own IP address but not mine just like what you did in the last question.

Hint: you don’t need to make any change to this existing XSS Web application. You simply need to demonstrate the vulnerability by cleverly constructing input to this XSS Web application. Probably Google Chrome browser will detect your XSS attacks, then you will need to restart your Google Chrome browser with a specific flag or simply change to another browser such as Firefox.

Question 8: (15 points)

1. Describe the details of your attacks (including the input you used). What are your verification results?