**SSD – Lab 4 – CSRF**

**For this lab, you may need IntelliJ idea IDE. You may download the evaluation version from here. You may use the Ubuntu VM from the earlier lab for this or you may use your own Windows or Linux PC.**

<https://www.jetbrains.com/idea/download/>

1. Extract the my-app.zip file.

2. Open the project using IntelliJ idea (you may use the enterprise evaluation edition. The project may be opened by other Maven supporting IDEs as well).

3. Build the project and build artifact my-app.war (build → build artificats → my-app.war)

Note: if the ‘build artifacts’ menu is disabled, you may enable it using the steps given in the following forum.

<https://stackoverflow.com/questions/36303535/intellij-build-build-artifacts-deactivated>

4. Download and extract the apache-tomcat web server.

<https://tomcat.apache.org/download-70.cgi>

5. Copy the my-app.war file in the my-app/target directory to the webapps directory in apache-tomcat directory.

6. Run tomcat using *bash catalina.sh run* (or Catalina.bat in windows) in the bin directory.

7. You should be able to access my-app using <http://localhost:8080/my-app>. Type the name and see whether you get redirected to welcome.jsp.

8. Currently the welcome.jsp page is vulnerable to CSRF attacks. You may install RESTClient plugin in firefox and try to send a POST request to the same url (http://localhost:8080/my-app/welcome.jsp), it should still give the welcome response with the name (this is because the cookie is valid, as the url is matching). The cookie is persistent for 5 mins. After 5 mins, the cookie is invalid.

<https://addons.mozilla.org/en-US/firefox/addon/restclient/> (if you get an error, try installing an older version)

Note:

You should be able to view the ‘name’ cookie being sent with each request by installing and running the SAML Tracer plugin in firefox.

<https://addons.mozilla.org/en-US/firefox/addon/saml-tracer/> (if you get an error when trying to install this, try installing an older version of the same plugin)

***Protecting against CSRF attack (using the OWASP CSRFGuard library)***

9. Add the following entry to the pom.xml file dependencies. Then right click the pom.xml and select Maven → Reimport. This should download the CSRFGuard library.

<!-- https://mvnrepository.com/artifact/org.owasp/csrfguard -->

<dependency>

<groupId>org.owasp</groupId>

<artifactId>csrfguard</artifactId>

<version>3.0.0</version>

</dependency>

10. Add the following entries to the WEB-INF/web.xml file.

<context-param>

<param-name>Owasp.CsrfGuard.Config</param-name>

<param-value>WEB-INF/Owasp.CsrfGuard.properties</param-value>

</context-param>

<context-param>

<param-name>Owasp.CsrfGuard.Config.Print</param-name>

<param-value>true</param-value>

</context-param>

<filter>

<filter-name>CSRFGuard</filter-name>

<filter-class>org.owasp.csrfguard.CsrfGuardFilter</filter-class>

</filter>

<filter-mapping>

<filter-name>CSRFGuard</filter-name>

<url-pattern>/\*</url-pattern>

</filter-mapping>

<listener>

<listener-class>org.owasp.csrfguard.CsrfGuardServletContextListener</listener-class>

</listener>

<listener>

<listener-class>org.owasp.csrfguard.CsrfGuardHttpSessionListener</listener-class>

</listener>

11. Create a new file under the WEB-INF directory and name it Owasp.CsrfGuard.properties. Add the following entries to it.

org.owasp.csrfguard.Logger=org.owasp.csrfguard.log.ConsoleLogger

org.owasp.csrfguard.ProtoctedMethod=POST,PUT,DELETE,GET

org.owasp.csrfguard.TokenPerPage=true

org.owasp.csrfguard.Rotate=false

org.owasp.csrfguard.Ajax=false

org.owasp.csrfguard.action.Log=org.owasp.csrfguard.action.Log

org.owasp.csrfguard.action.Log.Message=potential cross-site request forgery attack thwarted(user %user%, ip %remote\_ip%)

org.owasp.csrfguard.Redirect=org.owasp.csrfguard.action.Redirect

org.owasp.csrfguard.TokenName=OWASP\_CSRFTOKEN

org.owasp.csrfguard.SessionKey=OWASP\_CSRFTOKEN

org.owasp.csrfguard.TokenLength=32

org.owasp.csrfguard.PRNG=SHA1PRNG

org.owasp.csrfguard.Protect=true

org.owasp.csrfguard.protected.index=/my-app/welcome.jsp

12.Add the following entry at the top of the index.jsp file.

<%@ **taglib** uri="http://www.owasp.org/index.php/Category:OWASP\_CSRFGuard\_Project/Owasp.CsrfGuard.tld" prefix="csrf" %>

13. Add the following hidden field to the form in index.jsp

<input type="hidden" name="<**csrf:token-name**/>" value="<**csrf:token-value** uri="welcome.jsp"/>"/>

14. Repeat steps 3-8. Try to access <http://localhost:8080/my-app>

you should be able to see the CSRF token passed as a HTTP body parameter from SAML-tracer.

15. Try to access the <http://localhost:8080/my-app>/welcome.jsp directly from RESTClient. Now you should not get a response when you try to access welcome.jsp from RESTClient (this is because now a token is required to access the welcome.jsp. So even though the cookie is sent, the client can’t access the welcome.jsp page without the CSRF token).

Submission:

Upload the updated code in a zip file to the courseweb link. The zip file name should be your registration number.