Sistemas de Software Seguros
Segurança de Software
2024/2025
Class Project: Experiments with WebGoat (class 2)

1. Setup the System

Like in the previous class, we will WebGoat in this lab. Whenever there are questions on how to setup the tools, please look at the last project description. Execute the following steps to initiate the class:

a) Start the image of the course in a virtual machine (VirtualBox player)

b) Initiate the WebGoat by opening a Terminal, changing the current directory to `~/apps/WebGoat` and running: `java -jar webgoat*` . Then, you have to run the web browser (Firefox) and open WebGoat by selecting the URL:

```
http://localhost:8080/WebGoat
```

c) To enter, either register a new user or utilize the account with the username “`guestss`” and the password “`guestss`”. Select the “Sign in” button to initiate the lessons.

2. Broken Authentication

Try to solve the lesson Identity & Auth Failure >> **Password Reset** >> step 4

Authentication mechanisms often allow users who forget their passwords to reset/retrieve their passwords by responding to a question with personal data. However, it is often easy to guess the response to the question!

1) The lesson allows you to pick one of three users to find their answer to the question. Which user accounts usually exist in a typical system? From all of those, which one would you like to compromise? Maybe you can try that one.

2) How can you retrieve the password of that user account? Maybe you can brute force the color.

After solving this, go to the step 5 lesson and learn why many security questions you might think of can have problems!

Try to solve the lesson Identity & Auth Failure >> **Secure Passwords** >> step 4

Passwords continue to be one of the commonly used mechanisms for authentication. Therefore, it is fundamental to understand what a good/secure password is. Try with the different passwords of step 4 lesson and produce a password that is 4/4 secure enough.

Try to solve the lesson Sensitive Data Exposure >> **Insecure Login** >> step 2

Here, you want to have access to the credentials of another user. They are sent to the server when you press "Log in". How can you get them and submit them in the username/password fields of the form? (HINT: maybe you can use the proxy ZAP to see them or developer tools to get them … have a look at the annex at the end of the document).

Try to solve the lesson Identity & Auth Failure >> **Authentication Bypasses** >> step 2

Nowadays, web applications often need to provide a way for users to reset their passwords.

1) In this lesson, you are asked to answer two questions before you are allowed to reset your password. Start by experimenting with possible answers to the questions.

2) Not much success, right? Maybe this form of authentication suffers from a problem like the one described above in the web page, where a modification to the request sent to the web server causes an error condition that provides access.

To perform this sort of change, you have two alternatives: (1) use a **web proxy** to intercept the request; then, you change the request in the proxy; and lastly, you forward the changed request to the web server; or (2) you **modify directly their HTML code**, so that the request is done in a different manner by the browser. To execute either solution, you can use the ZAP Web Proxy or the Browser Developer Tools.

3) With this novel capability you can make arbitrary changes to a request sent by a browser. For example, you can delete some of the fields of an HTML form. What else? Maybe you can also *slightly modify the name* of the fields.

Try to solve the lesson Identity & Auth Failure >> **JWT tokens** >> step 4

Nowadays, Java applications use JSON Web Tokens (JWT) for client authentication to allow the client to indicate is identity for further exchange after authentication, such as the use of resources and services. A JWT token is composed of 3 parts, encoded in base 64 and separated by a "." (dot), in a form of *<header>.<claims>.<signature>*. The header describes the cryptographic operations applied to the JWT and, optionally, additional properties. The claims are the claims conveyed by the JWT. You can read Lessons 1, 2 and 3 to learn more about JWT.

Lesson 4: Try to find out what username is included in the JWT token. You can use tab JWT of Webwolf to help you decode the token.

3. XML External Entities (XXE)

Try to solve the lesson Security Misconfigurations >> XXE >> step 4

XXE can make XML interpreters perform unintended actions, namely show information they were not supposed to display. In this lesson, we will try to see a listing of the root directory of your machine. Start by experimenting with the form, by writing a few comments.

How can we perform the XXE attack? Apparently, it should be by accessing the contents of the submitted form. How can we do it? (HINT: maybe ZAP can help)

After you have intercepted the right message, how should you change the XML specification? Maybe you can get inspired in the slides or previous steps of the lesson. You will probably have to introduce something like "`file:///`".

4. Server-Side Request Forgery (SSRF)

In a Server-Side Request Forgery (SSRF) attack, the attacker can abuse functionality on the server to read or update internal resources. The attacker can supply or modify a URL which the code running on the server will read or submit data to. Also, by carefully selecting the URLs, the attacker may be able to read server configuration, connect to internal services like HTTP enabled databases or perform post requests towards internal services which are not intended to be exposed.

Try to solve the lesson Server-side Request Forgery >> Server-Side Request Forgery >> step 2 to 3

Lesson 2: How can we perform a SSRF attack to retrieve the jerry picture from the server. Maybe you can use ZAP or the Browser Developer Tools to help you.

Lesson 3: SSRF attacks are also used to retrieve data from any other server but resorting the connection of the current server. Try to get information from http://ifconfig.pro. Maybe you can use ZAP or the Browser Developer Tools to help you in this task.

Delivery of the Report

The output of the class project is a report answering all the questions and including the justifications for the responses. Each group should deliver the report either by submitting it in the course moodle page, or if there is some difficulty with this method, by emailing it to the professor of the TP class. The file type should be a pdf.

Deadline: 18 November 2024 (there will be no extensions)

=======================================
The next exercises are optional!
=======================================

Try also to do the following exercises

NOTE: *the next lesson is harder than the others, so leave for the end to solve!!*

Try to solve the lesson Identity & Auth Failure >> **JWT tokens** >> step 6

Lesson 6: The signature part of a JWT token allows to verify whenever the token was changed, and so check if it belongs to the connected user. However, for this to work correctly, the application needs to be correctly configured with the cryptographic properties included in the token. In this lesson, you are asked to reset the votes, which only can be performed by those who have admin permissions. Your mission is to compose a JWT token for a user with admin authorization, and then reset the votes. So, in summary, you need to change the JWT token of a user and then reset the votes. To start, you need to select one of the available usernames and the top right of the box. Maybe you can use the ZAP to intercept the request and the Webwolf to help you decode the token.