Lab 06: SQLi

Goals

- Identify vulnerabilities of a specified type in an application
- Remediate identified vulnerabilities using standard methods

Notes

The lab is available in the Learn org of Dakota State University's Information Assurance Lab within the vApp <username>_CSC234_Zwach_Base. It is accessible at https://ialab.dsu.edu. You can start the vApp by double clicking on it and then clicking the play button in the Flash client or clicking Actions and Start in the HTML5 client.

The code that comprises the entire application is available within the ~/course_files/_assignments/sqli /app directory. Make your changes there using a text editor of your choosing. Changes are immediately available once saved. Simply save the changes and then refresh the page. Remember, you can permanently revert your changes to the codebase at any time with the command `revertall`.

The username of the Kali VM is `student` and the password is `Password1!`.

Discovery

We will start our analysis by determining what is vulnerable in the application. We can assume it is vulnerable to SQL injection since this is a lab for the SQL injection period of the course. That said, take some time to respond to the following prompts.

1. Review the source code for the application of interest (~/course_files/_assignments/sqli/app).
   a. What language is used for the application?
   b. What library is used for interaction with the DBMS?
   c. What are the username and password used to access the database?
   d. Which webpage contains the vulnerability?
   e. Provide a screenshot of the vulnerable line of code.
2. Access the application by navigating to http://localhost:8000 within your Kali VM and selecting **SQLi Assignment** from the list of links.
   a. Search for the username jo. How many results are there?
   b. Go back to the homepage. Press F12 to access developer tools and then switch to the network tab. Search for the username bill. What are the parameters and values passed to the application as form data?
   c. Based on your knowledge of the source of the application, what would be a good string to test the index.php page with?
   d. Enter that string and submit the form. Provide a screenshot of the results.

Testing

Next, use either sqlmap or your own attack strings to confirm your assumptions about SQL injection and our application. Note that we are using the POST method instead of GET so either `--forms` or `--data` are required arguments to sqlmap.

3. Use sqlmap or your own attack strings to test for SQL injection

   a. Provide a screenshot of the sqlmap command or custom attack string used to collect banner information from this application.
   b. Provide the banner as a screenshot or string.
   c. Provide a screenshot listing the tables in the database.
   d. How many records are in the users table?

Remediation

After confirming the vulnerability, use your knowledge and available resources to secure the vulnerable portion of the application as you see fit.

4. Edit the source code from within the ~/course_files/_assignments/sqli/app directory. Changes will be reflected immediately within the docker container.
   a. Resolve the issue you identified in 1d and document your solution with a screenshot or screenshots allowing review of the code
   b. Describe the changes you made and why
5. Test your changes
   a. Are you able to submit any malicious queries?
   b. Is sqlmap or any other tool you would like to use able to detect any vulnerabilities? Provide screenshots of your attempts.
      i. **NOTE**: You will have to run `sqlmap --purge` to clear the previous session

Scoring

The rubric will be published at scoring time. Each portion of the assignment has the following points assigned.

Section	Points
1	4
2	4
3	2
4a	2
4b	6
5	2
Total	**20**