ISEC 400 Homework 9 Name: Megan Leonard

Answer the following questions based on your reading of the textbooks, any supplemental material, and the instructor’s presentation this week. If you use an external source (i.e. a web page, the required textbook, or an additional book) to help you answer the questions then be sure to cite that source. Hint: you should probably always be citing a source.

## Questions

1. **[5 points]** Explain, in your own words, the vulnerabilities that lead to successful SQL injection attacks.

Vulnerabilities that lead to successful SQL injection are error messages with too much information, not using prepared statements within the database, letting input from the user be taken and used instead of using placeholders, and just letting the user input interact directly to the server and database.

1. **[5 points]** Explain, in your own words, the common pitfalls that developers fall into when trying to prevent SQL injection attacks (i.e. what doesn’t work, and why).

Common pitfalls start with not allowing any strings that have a single quote as this is needed for some names. Another is to blacklist specific common injections as there are still ways around those. Finally, another pitfall is making a blacklist/whitelist with regular expressions as they could be altered by the user input so they can be passed through.

1. **[5 points]** Find two up-voted examples of answer code on [Stack Overflow](https://stackoverflow.com/) that contain SQL injection vulnerabilities. Provide a link, summary of the wrong answer, a rationale as to why it is wrong, and a replacement answer of your own based on best practices in Application Security.

<https://stackoverflow.com/questions/601300/what-is-sql-injection>

The examples I found are the top two answers for this post at 93 and 19 up votes. The first offers a PHP example which is vulnerable as it allows the password to be changed for every account.

PHP:

$password = $\_POST['password'];

$id = $\_POST['id'];

$sql = "UPDATE Accounts SET PASSWORD = '$password' WHERE account\_id = $id";

SQL:

UPDATE Accounts SET PASSWORD = 'xyzzy' WHERE account\_id = account\_id

The best way to change these is the use of prepared statements.

$s = $db-> prepare(“INSERT INTO user(id, password) VALUES(:id, :password)”);

$s->bindParam(‘id’, $id);

$s->bindParam(‘password’, $ password);

$s->execute();

The second example has the person being able to stop the password section from being read so they can get in without needing the password.

$sql = "SELECT FROM users WHERE username='".$\_GET['username']."' AND password='".$\_GET['password']."'";

The method from the previous example can be used here as well to parameterize and store the values.

$s = $db-> prepare(“INSERT INTO user(username, password) VALUES(:username, :password)”);

$s->bindParam(‘username’, $ username);

$s->bindParam(‘password’, $ password);

$s->execute();

1. **[5 points]** Describe the three solutions that do help prevent injection attacks and how they work. Do any of these need to work together for a defense in depth solution?

The first solution is to replace all single quote characters in the user input with two single quotes. The second is prepared statements as they bind parameters to placeholders. The third is stored procedures which help avoid user queries. The stored procedures work best with the prepared statements for a defense in depth solution.

1. **[5 points]** Describe the problems and solutions associated with insecure direct object reference vulnerabilities.

The problems associated with insecure direct object reference vulnerabilities are the exposure of internal implementations, users being able to alter these internal implementations, and the authorization issue where it is bypassed. The solution is the 3x3 model with the first 3 being the what dimension with subjects, objects, and operations. The second 3 being the when dimension with the before loading interface, before requests are submitted, and before granting final access.

1. **[5 points]** In approximately 300 to 400 of prose (i.e. sentences, not bullet lists) using APA style citations if needed, summarize and interact with the content that was covered this week in class. In your summary, you should highlight the major topics, theories, practices, and knowledge that were covered. Your summary should also interact with the material through personal observations, reflections, and applications to the field of study. In particular, highlight what surprised, enlightened, or otherwise engaged you. Make sure to include at least one thing that you’re still confused about. In other words, you should think and write critically not just about what was presented but also what you have learned through the session. Ask at least one question that your instructor can answer in the returned assignment or class discussion.

This week we went over more SQL injection with examples of bad and good practices and insecure direct object references. SQL injection is a topic that I have covered several times but still find interesting as there is a lot that can be done to cause the injection and a lot that can be done to stop it. Out of the solutions, I like the prepared statements the most as that is one that I have worked on prior to this class and it is interesting to see how it all comes together. My question is do you have a favorite solution for SQL injection?

Citation:

Whittaker, T., &amp; Whitney, T. (2022). Week08 Power Point. Retrieved March 26, 2023, from <https://franklin.instructure.com/courses/11070/files/1516723?module_item_id=623556>