Portswigger: Blind SQL injection

Sam Newman

April 2025

1 Website in Question

This section of the website contains a vulnerability with the injection of the user tracking cookie into the SQL search in the website's backend, the purpose of which is to see if you are a returning user or not.

2 Business Impact

This is an extremely dangerous vulnerability. Using it, any user could gain administrator permissions over the system and could accomplish anything that an administrator account could do, in addition to cracking any user's password and logging into their account.

3 How It Works

Begin by checking whether the tracking cookie is vulnerable to SQL injection. We can see that we can modify the website's behavior in the Welcome Back functionality by adding 1' OR 1=1 – to the cookie, and the inverse with AND 1=2, which does not make that response. This is blind, so we will have to manually guess and check each digit of the user password. Use BurpSuite to create a payload of every lower and uppercase letter and every number digit to try at each position of the password. Then use "1' OR SUBSTRING((SELECT password FROM users WHERE username='administrator'), 1, 1)=" –" as the payload for the tracking cookie, using a Burp sniper attack to iterate through everything in your payload list in between the ". I then did that for all 20 characters of the password, and string matched 'Welcome' to the Burp response to see what payload reacted.

4 Why It Works

The website has a vulnerability in the way SQL queries are passed to the server. They are not trimmed and filtered but rather passed as raw input. This should never be done in a SQL server.

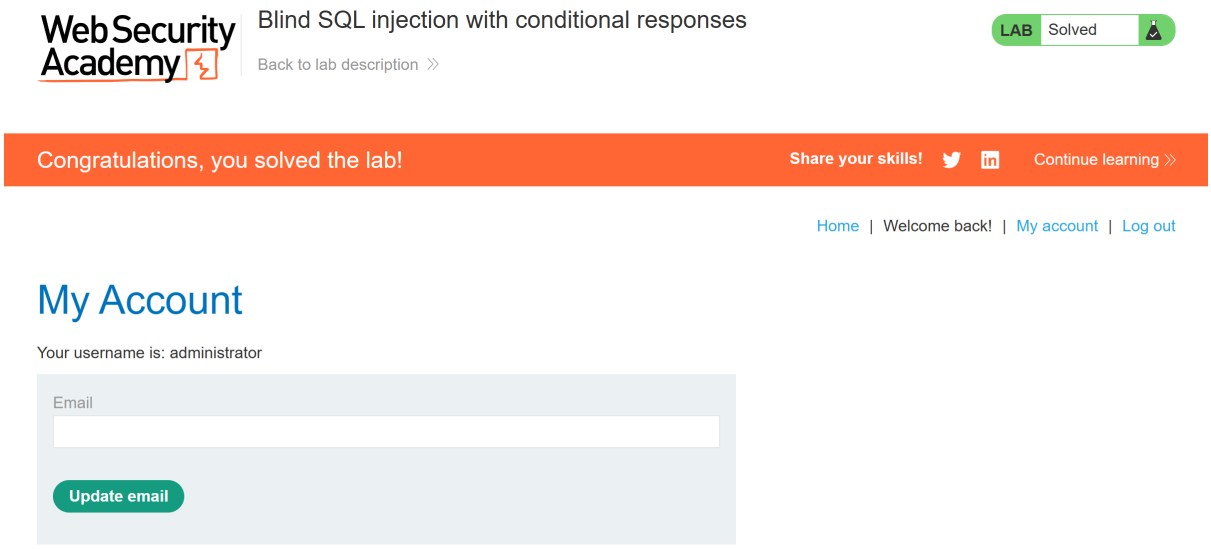

Figure 1: Solved Lab

5 How to Mitigate

Just have to parse the user input to the SQL query with $query methodology, or use a string method to ensure all characters are alphanumeric so quotes cannot be used to break out of the query.