ENPM809W - Introduction to Secure Coding

Lab - 6 – Defense Lab – Secure Authentication Authorization

Author: Syed Mohammad Ibrahim
UID: iamibi
Email: iamibi@umd.edu

Phase 1: Adding Timeouts to Login

a) Provide the URL of the WebGoat.NET application page where you are exercising the question. Do not include any query parameters or any other special characters in the answer.

http://localhost:52251/WebGoatCoins/CustomerLogin.aspx

b) For the given CWE-ID, Filename, Line Number of the weakness identified in the previous Attack Lab, describe how you intend to fix the weakness. This can be as detailed as possible to explain how it will address the weakness.

The intended fix is to keep a text-based log file which the server will write to in case of a failed login attempt. The log file will contain the timestamp (in UTC) and the email address being used. The server will then try to validate the current login attempt first by checking the log file and respond whether to fail silently or login the user successfully.

c) Describe why your intended fix will address the weakness (either directly or indirectly) and whether to your knowledge it will prevent future attacks along the lines of the attack vector used in the previous lab. This can be as detailed as possible to explain why it will address the weakness.

Considering the previous lab attack vectors, the fix will make sure that we don't query the database or use Argon 2 functionality. This will save the server resources and failing silently will not give an attacker any kind of hint. Precisely, we are limiting the number of failed login attempts to three after which even if the user enters a correct password, they will still not be able to login for the one hour window.

d) List all the paths with filenames you are changing to implement the fix for the weakness.
- .vs/WebGoat.NET/config/applicationhost.config
- WebGoat/WebGoatCoins/Logs/logfile.txt
- WebGoat/WebGoatCoins/CustomerLogin.aspx.cs
- WebGoat/WebGoat.NET.csproj

e) Commit ID: cfb6ea3d4a0f46d084ebb512e8bae183ea012405

Phase 2: Setting up TLS to prevent eavesdropping

a) Provide the URL of the WebGoat.NET application page where you are exercising the question. Do not include any query parameters or any other special characters in the answer.

   N/A

b) For the given CWE-ID, Filename, Line Number of the weakness identified in the previous Attack Lab, describe how you intend to fix the weakness. This can be as detailed as possible to explain how it will address the weakness.

   Generating a local trusted certificate and enabling SSL on Visual Studio project.

c) Describe why your intended fix will address the weakness (either directly or indirectly) and whether to your knowledge it will prevent future attacks along the lines of the attack vector used in the previous lab. This can be as detailed as possible to explain why it will address the weakness.

   Enabling SSL on project prevents the eavesdropping by Burp Suite. This doesn't allow anyone to see the request being made to capture it.

d) List all the paths with filenames you are changing to implement the fix for the weakness.
   - .vs/WebGoat.NET/config/applicationhost.config
   - WebGoat/WebGoat.NET.csproj

e) Commit ID: 1cce4fd446aaefbaa97847da7b8e504a0956090d

Phase 3: Secure Password Reset

a) Provide the URL of the WebGoat.NET application page where you are exercising the question. Do not include any query parameters or any other special characters in the answer.

   http://localhost:52251/Content/ForgotPassword.aspx

b) For the given CWE-ID, Filename, Line Number of the weakness identified in the previous Attack Lab, describe how you intend to fix the weakness. This can be as detailed as possible to explain how it will address the weakness.

Generating a time limit based OTP and following RFC 6238 Section 5.2 for OTP. Saving the OTP in the database and verifying the OTP when the user tries to change the password which was sent as part of the change password email.

c) Describe why your intended fix will address the weakness (either directly or indirectly) and whether to your knowledge it will prevent future attacks along the lines of the attack vector used in the previous lab. This can be as detailed as possible to explain why it will address the weakness.

Having a change password email being sent as part of authorizing the user to change their password when they forget it prevents an attacker from spoofing the identity or trying unrestricted logins. As part of the email, we pass a URL with a time limited OTP code which expires after 60 seconds of server window. This OTP is helpful in identifying the requests uniquely for users as we are saving them in the database as well. When the user clicks on the URL and is landed on the change password page, we try to authenticate using the OTP and verify that it is actually the same user that originated the request of change password. After verification we change the password in the database successfully.

d) List all the paths with filenames you are changing to implement the fix for the weakness.
- C:\Users\student\Workspace\webgoat\WebGoat\App_Code\DB\MySqlDbProvider.cs
- C:\Users\student\Workspace\webgoat\WebGoat\App_Code\Util.cs
- C:\Users\student\Workspace\webgoat\WebGoat\Content\ChangePwd.aspx.cs
- C:\Users\student\Workspace\webgoat\WebGoat\Content\ChangePwd.aspx
- C:\Users\student\Workspace\webgoat\WebGoat\Content\ForgotPassword.aspx.cs
- C:\Users\student\Workspace\webgoat\WebGoat\App_Code\SendEmailUtil.cs

e) Commit ID: a7c63688e50459411917ad9e9958ee87dbd3fb01