**Homework 3**

**Authentication, Authorization, and Session Management Security Controls Assignment**

**Student**: Patrick Walsh

**School**: University of Maryland Global Campus

Date: 9/24/2021

**Course**: SDEV 460 6380

**Professor**: Dr. Chris Howard

1. **Test Role Definitions (OTG-IDENT-001)**

Create a test matrix for the following roles:

* Tutor role who can accept student tutoring sessions, cancel student tutoring sessions,   
  and email students with tutor details.
* Student role who can register to receive tutoring, request tutoring sessions, cancel   
  student tutoring sessions and email tutors with student details.

The following table was based on the OWASP Testing Guide v4 (n.d.-a) and includes the role, the role’s permission, the object data that the permissions pertain to, and any constraints applicable to the role:

|  |  |  |  |
| --- | --- | --- | --- |
| **Role** | **Permission** | **Object** | **Constraints** |
| Tutor | Read | Tutoring session data, Student email data | Can view data related to tutoring sessions they are leading, and name and contact info for Student email data |
| Student | Read | Tutoring session data, Tutor email data | Can only view data related to tutoring sessions they have registered for, and name and contact info for Tutor email data |

1. **Test User Registration Process (OTG-IDENT-002)**

Using the OWASP testing guide, define and make recommendations for a student registration   
process. The registration can be as simple as requiring the student to complete a form with a   
few fields (e.g., name, state, DOB) and the system assigns an EMPLID. The student should also   
create a secure password.

The OWASP Testing Guide v4 recommends (n.d.-b) that identity requirements for user registration are aligned with business and security requirements. Since a college or university likely wants to limit student registration to students at their university, the registration process could include a step to verify that the person registering is a student at the school. One way to do this would be to provide a temporary, one-time use access code to the student at the time of registration. New registrations could include identifying information like student name, home address, date of birth, and the access code.

The second recommendation is to validate the registration process (OWASP Testing Guide v4, n.d.-b). To validate the student’s identification during registration, the student would be required to show student I.D. to prove their identity. As an added, though possibly unnecessary layer of security, registration could include two-factor authentication (2FA) where the student must verify their phone number with a temporary verification code through SMS.

1. **Testing for Credentials Transported overrun Encrypted Channel (OTG-AUTHN-001)**

Assume the Web Application has not implemented HTTPS for the forms and registration. What recommendations do you have to improve this process? How do other sites authenticate a user?

It is very unsafe to send sensitive information through an HTTP request because the information is not encrypted. Any malicious user with a web traffic sniffer such as Wireshark could intercept the communications and read any sensitive information like username or password (OWASP Testing Guide v4, n.d.-c). The ‘s’ in HTTPS is critical because it means the web traffic is encrypted. Any pages that contain forms, registration pages, or logins should send requests via encrypted HTTPS so protect user information (OWASP Testing Guide v4, n.d.-c). To authenticate a user, the user login credentials entered on the webpage are sent through a POST request. As long as the web application is using the HTTPS protocol, the request will be encrypted (OWASP Testing Guide v4, n.d.-c). Additionally, web pages should not use GET requests for sending sensitive information because the information is sent in the URL in plain text (OWASP Testing Guide v4, n.d.-c).

1. **Testing for default credentials (OTG-AUTHN-002)**

What process and resources would you use to attempt to guess a username and password for a   
student user? (Hint: Are there common password lists?) Can you guess their email or emplid   
based on a known pattern (e.g., [studentname@student.umgc.edu](mailto:studentname@student.umgc.edu))?

Guessing the username should not be difficult since many websites use the email as the username, or the first part of the email, before the @ symbol. In the case of UMGC, the email follows a general pattern of first letter of the first name, last name, and sometimes numbers if there are multiple students with the same last name and first initial. For example, John Smith’s email would start with jsmith or jsmith3, etc. The domain is always @student.umgc.edu.

To guess the password, there are lists of commonly used passwords. As of 2021, the most common passwords are still as follows (Meyer, 2021):

1. 123456
2. 123456789
3. qwerty
4. password
5. 12345
6. qwerty123
7. 1q2w3e
8. 12345678
9. 111111
10. 1234567890

However, none of these passwords meet UMGC’s password complexity requirements which specify that passwords must be between 10-127 characters in length and must contain at least one lowercase letter, one uppercase letter, number, and special character (UMGC, n.d.).

There are also dumps available online of mass password hacks. It is not advisable to go searching for these websites, but there were an estimated 1.4 billion passwords available online from various password breaches (Porup, 2018). Some of these password dumps are sold on the dark web (Porup, 2018). As many people reuse the same password across multiple sites, even of passwords that are many years old, this represents a serious security threat (Porup, 2018).

1. **Testing for Weak lock out mechanism (OTG-AUTHN-003)**

What is recommended if a student attempts to login with improper credentials for several   
attempts? Provide a sample algorithm (i.e., pseudocode) you would recommend implementing   
for this process.

According to the OWASP Testing Guide v4, accounts are typically locked after 3 to 5 unsuccessful login attempts (n.d.-e). After an account is locked, there is often a predetermined time limit that must pass before the user can attempt another login. In some cases, the user must first unlock the account through a self-service unlock mechanism or intervention by an administrator (OWASP Testing Guide v4, n.d.-e).

A simple example of pseudocode to represent an account lock procedure could be this:

// variables set for each user session

Login\_attempts = 3

Account\_locked = False

// user clicks submit button on login page:

If Login\_attempts < 1:

Account\_locked = True

If Account\_locked == False AND username == correct\_username AND password == correct\_password:

Load\_login\_page()

Else:

Login\_attempts -= 1

1. **Testing for Weak password policy (OTG-AUTHN-007)**

What password recommendations would you have for the student and tutor users? Be sure to   
use and reference the latest NIST password recommendations.

The latest guidelines for secure passwords were published by the National Institute of Standards and Technology (NIST) in 2017 and updated in January 2021 (Poza, 2021). The guides include some surprising recommendations that contradict password security practices of the past. For example, NIST recommends that organizations should remove periodic password resets, contrary to what many security experts recommended in the past (Poza, 2021). The reason for this is that forcing users to reset their passwords every few months often leads them to change their passwords in very minor, predictable patterns, making guessing the passwords easy if they are breached (Poza, 2021). Another recommendation by NIST is to enable the password field to be visible when typing if the user allows it (Poza, 2021). The thinking behind this recommendation is that users can be discouraged from entering longer and more complex passwords when the password field is hidden because it is easier to make a mistake. Along similar lines, NIST recommends allowing passwords to be pasted in as an option (Poza, 2021). This allows users to enter longer, more complex passwords that are more difficult to guess. If the password field does not allow a paste-in option, it forces the user to manually type out the password every time, discouraging the user from choosing a strong password (Poza, 2021).

Perhaps the most important recommendation from NIST is in the creation of the password itself. NIST states that password length is much more important than password complexity (Poza, 2021). The reason is that longer passwords are more difficult to decrypt if they are stolen (Poza, 2021).

1. **Testing for logout functionality (OTG-SESS-006)**

How would implement session logout properly? Provide an example in the code of your choice.

Proper session termination is critical to preventing certain attacks that take advantage of active user sessions. Such attacks include Cross Site Scripting (CSS) and Cross Site Request Forgery (CSRF) attacks which can be launched when user sessions have not been properly logged out (OWASP Testing Guide v4, n.d.-f).

In Java, sessions are managed with the HttpSession interface defined in the javax.servlet packge (Minh, 2019). There are two basic ways a session is terminated. The first is an automatic timeout. Each web server has its own standard for length of the timeout session, but 30 minutes is standard practice (Minh, 2019). The session timeout length can also be modified in the web.xml file (Minh, 2019).

The second way of terminating a session in Java is for the user to logout of the application. When the user logs out, the application should terminate the session so that no session credentials are left lingering where malicious users could launch a CSS or CSRF attack. To terminate a session in Java, use the invalidate() method (Minh, 2019). Below is a code snippet of a logout page which uses the invalidate() method to terminate the user session upon logging out:

import java.io.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

public class Logout extends HttpServlet {

    protected void doGet(HttpServletRequest request, HttpServletResponse response)

        session = request.getSession();

session.invalidate();

        response.sendRedirect(request.getContextPath() + "/index.html");

    }

}

**References**

Meyer, B. (2021, September 22). *Most common passwords: latest 2021 statistics*. CyberNews. https://cybernews.com/best-password-managers/most-common-passwords/

Minh, N. H. (2019, June 28). *How to use Session in Java web application*. Codejava.Net. https://www.codejava.net/java-ee/servlet/how-to-use-session-in-java-web-application

OWASP Testing Guide v4. (n.d.-a). *Test Role Definitions (OTG-IDENT-001) | Owasp Testing Guide v4*. Kennel209.Gitbooks.Io. https://kennel209.gitbooks.io/owasp-testing-guide-v4/content/en/web\_application\_security\_testing/test\_role\_definitions\_otg-ident-001.html

OWASP Testing Guide v4. (n.d.-b). *Test User Registration Process (OTG-IDENT-002) | Owasp Testing Guide v4*. Kennel209.Gitbooks.Io. https://kennel209.gitbooks.io/owasp-testing-guide-v4/content/en/web\_application\_security\_testing/test\_user\_registration\_process\_otg-ident-002.html

OWASP Testing Guide v4. (n.d.-c). *Testing for Credentials Transported over an Encrypted Channel (OTG-AUTHN-001) | Owasp Testing Guide v4*. Kennel209.Gitbooks.Io. https://kennel209.gitbooks.io/owasp-testing-guide-v4/content/en/web\_application\_security\_testing/testing\_for\_credentials\_transported\_over\_an\_encrypted\_channel\_otg-authn-001.html

OWASP Testing Guide v4. (n.d.-d). *Testing for default credentials (OTG-AUTHN-002) | Owasp Testing Guide v4*. Kennel209.Gitbooks.Io. https://kennel209.gitbooks.io/owasp-testing-guide-v4/content/en/web\_application\_security\_testing/testing\_for\_default\_credentials\_otg-authn-002.html

OWASP Testing Guide v4. (n.d.-e). *Testing for Weak lock out mechanism (OTG-AUTHN-003) | Owasp Testing Guide v4*. Kennel209.Gitbooks.Io. https://kennel209.gitbooks.io/owasp-testing-guide-v4/content/en/web\_application\_security\_testing/testing\_for\_weak\_lock\_out\_mechanism\_otg-authn-003.html

OWASP Testing Guide v4. (n.d.-f). *Testing for logout functionality (OTG-SESS-006) | Owasp Testing Guide v4*. Kennel209.Gitbooks.Io. https://kennel209.gitbooks.io/owasp-testing-guide-v4/content/en/web\_application\_security\_testing/testing\_for\_logout\_functionality\_otg-sess-006.html

Porup, J. M. (2018, March 28). *1.4B stolen passwords are free for the taking: What we know now*. CSO Online. https://www.csoonline.com/article/3266607/1-4b-stolen-passwords-are-free-for-the-taking-what-we-know-now.html

Poza, D. (2021, January 22). *NIST Password Guidelines and Best Practices for 2020*. Auth0 - Blog. https://auth0.com/blog/dont-pass-on-the-new-nist-password-guidelines/

UMGC. (n.d.). *Help Center | Article Topic | UMGC*. Umgc.Edu. https://www.umgc.edu/help/help-topic.cfm?id=kA0d0000000DG1xCAG&table=FAQ\_IT\_\_kav&action=getArticle&fbSource=RelatedArticle