


Northumbria University
Department of Computer and Information Sciences

PE7045 Secure Web Development
Final Assignment
Security Report

Student: Scott Cumming
Student Number: 
Date: 20th September 2022

Contents

1. Introduction	2
2. Security Goals	2
3. Architecture, Trust Levels and Entry Points	2
4. Vulnerability Assessment	4
5. Conclusion	6
Appendix A – Bibliography	7

Word Count: **1042** (excluding tables, headings, footnotes and the appendix)

1. Introduction

This report forms part of the author's submission for the PE7045 Secure Web Development module final assessment. It critically reviews the security of the web application created by the author which is hosted on the newnumyspace server¹.

2. Security Goals

The web application and business are subject to the provisions of the Data Protection Act 2018 and UK General Data Protection Regulation. This legislation requires organisations to secure any personal data that they hold, protecting it from unauthorised access. If they fail to do this, the Information Commissioner's Office can issue them a significant fine (Information Commissioner's Office, no date).

Consequently, the business should make the protection of customer data a priority to avoid severe legal and financial repercussions. Similarly, it needs to protect the integrity of the application as a successful attack can undermine customer confidence and cause severe reputational damage, which can ultimately impact upon the profitability of the business (Investopedia, 2022).

3. Architecture, Trust Levels and Entry Points

As noted in the introduction, the website is hosted on the newnumyspace web server. Customer details, login credentials, accommodation data and booking information are all stored on a separate database server running MySQL.

Users can interact with the application at the entry points listed in the table below:

ID	Name of Entry Point	Description	Trust Levels
1	Home Page	This is the main entry point for all users.	- Anonymous Web User - User with Valid Login Credentials - User with Invalid Login Credentials
2	Register Page	Anyone can visit this page and complete an HTML form to register their details in the database.	- Anonymous Web User - User with Valid Login Credentials - User with Invalid Login Credentials
2.1	Register Process Page	When registration details are submitted using the form in 2, this page displays any errors made. It also contains a button to remove the errors and redisplay the form for	- Anonymous Web User - User with Valid Login Credentials - User with Invalid Login Credentials

		resubmission. During this process, a SQL query is submitted to check whether the email address is already registered in the database. If there are no errors, the details are entered into the database and a confirmation message is displayed. The user is not automatically signed in at this stage.	
3	Sign In Page	Anyone can access this page, which asks the user to enter their email address and password in an html form to login.	<ul style="list-style-type: none"> - Anonymous Web User - User with Valid Login Credentials - User with Invalid Login Credentials
3.1	Authenticate Page	When the html form in 3 is submitted, this page displays any errors that were found. During this process, a SQL query is made to compare the password submitted against the stored hash linked to the email address given. If there are no errors, the user is logged in and a session is started.	<ul style="list-style-type: none"> - Anonymous Web User - User with Valid Login Credentials - User with Invalid Login Credentials
4	Accommodation Search Page	Anyone can search through the accommodation listed on the site. This involves the user completing an html form, which is used to populate a SQL query that returns the search results.	<ul style="list-style-type: none"> - Anonymous Web User - User with Valid Login Credentials - User with Invalid Login Credentials
5	Accommodation Listing Page	This page displays various details for a selected property. Only signed in users can book the accommodation. If not signed in, a link to the sign in page is provided.	<ul style="list-style-type: none"> - Anonymous Web User - User with Valid Login Credentials - User with Invalid Login Credentials
5.1	Accommodation Listing Page (Validated users only)	If the user is signed in, the link to the sign in page mentioned above in 5 is replaced with an html booking form.	<ul style="list-style-type: none"> - User with valid login credentials only
5.2	Booking Page	When a booking is submitted through the form in 5.1, the user is taken to this page. If there are errors in the form, these are displayed. Otherwise, details of the proposed booking are displayed and the user is	<ul style="list-style-type: none"> - User with valid login credentials only

		requested to confirm them by clicking a button.	
5.3	Booking Confirmation Page	After clicking the button in 5.2, the booking details are entered into the database and a message is displayed telling the user their booking was successful. They are then redirected to the home page. If the booking was unable to be completed (e.g. due to a database issue), the user is asked to try again later.	- User with valid login credentials only
6	Contact Us Page	This page contains an email address for the business which all users can contact if they have a query or need to reset their password.	- Anonymous Web User - User with Valid Login Credentials - User with Invalid Login Credentials

4. Vulnerability Assessment

The register page (ID 2) asks the user to input the following information: first name; surname; date of birth; address details; email address; and password (twice). When the user submits this information, the web server carries out validation checks using the `preg_match()` and `filter_var()` functions to ensure the data is in the correct format. These checks filter out any characters which could be used to write malicious code, such as '<', '>', '/', '=', or '+'. As a result, they should limit an attacker's ability to launch an SQL injection or XSS attack.

Allowing users to set their own username is not effective at preventing it from being enumerated by an attacker (Stuttard & Pinto, 2011, p.166), therefore the user's email address is employed as the username in the application. This approach means the email address must be unique and not held by more than one user. Consequently, an SQL query is used to check whether the email address already exists in the database during the registration stage. If it is, the following error message is displayed:

"The email address you entered is already registered - please go to the sign in page"

This is a vulnerability as an attacker could use brute force to identify which email addresses are already used as login credentials (Hope & Walther, 2008, pp.248-249). To mitigate this vulnerability, a non-descript message should be displayed on the screen instead and an email should be sent to the recipient advising them what to do next (Stuttard & Pinto, 2011, p.196), although this would require access to a SMTP server.

Passwords entered by the user must contain at least 9 characters and one capital letter, lower case letter, number and special character. This is considered to be a

relatively strong password, however there is scope to check it does not match the username/email address entered or contain sequences (Najera-Gutierrez et al., 2019, p.174).

The password is stored as a hash in the database using the bcrypt algorithm, which strengthens its protection against brute force attacks (Dulaney & Easttom, 2018, p.249). When entered into the database with the other information submitted by the user, the data is binded to a prepared statement as this increases the protection against SQL injection attacks (Najera-Gutierrez et al., 2019, p.232). This approach is used throughout the application whenever a query with user input is sent to the database.

The sign in page (ID 3) requires the user to enter their email address and password. Again, a prepared statement is used to check whether any records in the database match the email address entered. If a record is returned, the hash generated by the password entered is compared against the hash stored alongside the email address. If the hashes match, the user is logged in and redirected to the home page. If the email address is not recognised or the password was incorrect, the user is told so explicitly via a message on the screen. As before, this constitutes a vulnerability as an attacker could use these messages to enumerate usernames/email addresses.

The security of the sign in function could be improved using dual factor authentication, an account lockout process and a policy that requires users to change their password on a frequent basis. However, these features aren't fool proof and can introduce further vulnerabilities if not implemented correctly (Najera-Gutierrez et al., 2019, p.156; Hope & Walther, 2008, p.192; Stuttard & Pinto, 2011, p.172).

Password resets are currently requested via the email address provided on the contact us page. Although this enables the business to manually authenticate each user, it could also offer an attacker a means to obtain access to an account using impersonation and social engineering (Salahdine & Kaabouch, 2019). The introduction of a password reset function provides an alternative, but this can be exploited too if it is poorly designed (Stuttard & Pinto, 2011, p.173).

Other measures could be implemented to improve the security of the application, such as setting the httponly flag on cookies to prevent some XSS attacks (OWASP, 2022). Authentication-related events could also be logged and monitored for anomalies (Stuttard & Pinto, 2011, p.201), but this may not be feasible for a small business.

The main vulnerability identified does not arise from the web application *per se* but from the server it is hosted on. This does not encrypt traffic using SSL or TLS, meaning personal data and login credentials are sent in cleartext and could be intercepted by an attacker conducting a man-in-the-middle attack (Imperva, no date). This vulnerability was observed using the Burp Suite proxy server (PortSwigger, 2022) and should be addressed as a matter of urgency.

5. Conclusion

The web application implements some good security measures that should prevent most SQL injection and XSS attacks, although error messages should be reworded to deter username enumeration. Further measures could be implemented as the business grows, but these need to be carefully designed to avoid introducing new vulnerabilities. The main vulnerability identified relates to the web server's use of HTTP; this should be remedied urgently using SSL or TLS encryption.

Appendix A – Bibliography

Data Protection Act 2018, c.12. Available at: <https://www.legislation.gov.uk/ukpga/2018/12/contents/enacted> (Accessed: 20th September 2022).

Dulaney, E. and Easttom, C. (2018) *CompTIA Security+ Study Guide*. 7th edn. Indianapolis, Indiana: John Wiley & Sons.

Hope, P. and Walther, B. (2008) *Web security testing cookbook*. Sebastopol: O'Reilly.

Imperva (no date) Man in the middle (MITM) attack. Available at: <https://www.imperva.com/learn/application-security/man-in-the-middle-attack-mitm/> (Accessed: 20th September 2022).

Information Commissioner's Office (no date) *Penalties*. Available at: <https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-le-processing/penalties/> (Accessed: 20th September 2022).

Investopedia (2022) *6 Ways Cybercrime Impacts Business*. Available at: <https://www.investopedia.com/financial-edge/0112/3-ways-cyber-crime-impacts-business.aspx> (Accessed: 20th September 2022).

Najera-Gutierrez, G. et al. (2019) *Improving Your Penetration Testing Skills : Strengthen Your Defense Against Web Attacks with Kali Linux and Metasploit*. Birmingham: Packt Publishing.

OWASP (no date) *HttpOnly*. Available at: <https://owasp.org/www-community/HttpOnly> (Accessed: 20th September 2022).

PortSwigger (2022) *Burp Suite Community Edition (Version 2022.8.4)* [Computer program]. Available at: <https://portswigger.net/burp/communitydownload> (Accessed: 20th September 2022).

Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (United Kingdom General Data Protection Regulation) (Text with EEA relevance) (SI 2016/679). Available at: <https://www.legislation.gov.uk/eur/2016/679/contents#> (Accessed: 20th September 2022).

Salahdine, F. and Kaabouch, N. (2019) 'Social Engineering Attacks: A Survey', *Future Internet*, 11(4), article number 89. Available at: <https://doi.org/10.3390/fi11040089> (Accessed: 20th September 2022).

Stuttard, D. and Pinto, M. (2011) *The Web Application Hacker's Handbook : Finding and Exploiting Security Flaws*. 2nd edn. Indianapolis: Wiley Publishing Inc.