



Faculty of Computer Science and Information Technology

OWASP A03 Injection vulnerability in Web Application Development

Andy Chieng Ging Wei

Bachelor of Computer Science with Honours (Software Engineering)

2023

OWASP A03 Injection vulnerability in Web Application Development

ANDY CHIENG GING WEI

This project is submitted in partial fulfilment of the requirements for the degree of Bachelor of Computer Science with Honours (Software Engineering)

Faculty of Computer Science and Information Technology

UNIVERSITI MALAYSIA SARAWAK

2023

KERENTANAN SUNTIKAN OWASP A03 DALAM PEMBANGUNAN APLIKASI WEB

ANDY CHIENG GING WEI

Projek ini merupakan salah satu keperluan untuk Ijazah Sarjana Muda Sains
Komputer dengan Kepujian (Kejuruteraan Perisian)

Fakulti Sains Komputer dan Teknologi Maklumat

UNIVERSITI MALAYSIA SARAWAK

2023

UNIVERSITI MALAYSIA SARAWAK

THESIS STATUS ENDORSEMENT FORM

TITLE OWASP A03 Injection vulnerability in Web Application Development

ACADEMIC SESSION: 2022/2023

ANDY CHIENG GING WEI

(CAPITAL LETTERS)

hereby agree that this Thesis* shall be kept at the Centre for Academic Information Services, Universiti Malaysia Sarawak, subject to the following terms and conditions:

1. The Thesis is solely owned by Universiti Malaysia Sarawak
2. The Centre for Academic Information Services is given full rights to produce copies for educational purposes only
3. The Centre for Academic Information Services is given full rights to do digitization in order to develop local content database
4. The Centre for Academic Information Services is given full rights to produce copies of this Thesis as part of its exchange item program between Higher Learning Institutions [or for the purpose of interlibrary loan between HLI]
5. ** Please tick (✓)

CONFIDENTIAL (Contains classified information bounded by the OFFICIAL SECRETS ACT 1972)

RESTRICTED (Contains restricted information as dictated by the body or organization where the research was conducted)

UNRESTRICTED

Validated by

ANDY
(AUTHOR'S SIGNATURE)

(SUPERVISOR'S SIGNATURE)

Permanent Address

2F, Jalan Kulas Timur

96000, Sibu Sarawak

Date: 30/6/2023

Date:

Note * Thesis refers to PhD, Master, and Bachelor Degree

** For Confidential or Restricted materials, please attach relevant documents from relevant organizations / authorities

DECLARATION

I hereby declare that this project is my original work. I have not copied from any other student's work or from any other sources except where due reference or acknowledgement is not made explicitly in the text, nor has any part had been written for me by another person.

Hudi

.....

(Andy Chieng Ging Wei)

01July 2023

Matric No: 69073

ACKNOWLEDGEMENT

I would like to thank my supervisor, Dr. Lim Phei Chin for the advice, guidance, and encouragement throughout my final year project. I would also like to thank my examiner which is Dr.Ling Yeong Tyng for the advice on this project. I would also like to thank the God to give me wisdom in doing this project. I would also like to thank my friends and my family who gave me opinions and help me throughout my final year project.

ABSTRACT

In this information technologies era, more and more hacker appeared to attack the vulnerable web application on the internet. As a web developer to be, the junior web developer has to be equipped with knowledge and be prepared to handle the common vulnerability in web application. To accommodate with this goal, this project is being proposed and developed to teach the web developer on the fundamental about Open Web Application Security Project (OWASP) Top 10, focusing on OWASP Top 10 A03 - Injection vulnerability. OWASP A03 Injection vulnerability is one of the most common vulnerabilities can be found in the web application. In this web application, web developer will be well-educated on the vulnerability as well as the defencing ways to enhance the security of the web application developed that can defence against this vulnerability. Besides, several research paper and the existing application related will be reviewed as a resource of requirement gathering which helps in designing on the functionality and features of web application complied with OWASP A03 Injection vulnerability. Other than that, the development of the web application will be documented. Besides, the result on testing on the web application will be analysed in this report. From the positive result gained from testers involved in both laboratory usability testing and usability testing using system usability scale, it is believed that this project can help to increase understanding of the web developer on the OWASP A03 vulnerability.

ABSTRAK

Pada era teknologimaklumatini, semakinbanyakpenggodammenyerangaplikasi web yang rentan di internet. Sebagai pembangun web, pembangun web junior harus dilengkapidengan pengetahuan dan bersedia untuk menanganikerentanan umum dalam aplikasi web. Untuk memenuhi tujuan ini, projek ini dicadangkan dan dikembangkan untuk mengajarpembangun web mengenai asas mengenai Open Web Application Security Project (OWASP) Top 10, memberitumpuan kepada OWASP Top 10 A03 - Kerentanansuntikan. Kerentanansuntikan OWASP A03 adalah salah satu kelemahan yang paling biasadijumpai dalam aplikasi web. Dalam aplikasi web ini, pembangun web akan berpendidikan tinggi mengenai kerentanansertacara-cara pertahanan untuk meningkatkan keselamatan aplikasi web yang dikembangkan yang dapat mempertahankan diri dari kerentanannya. Selain itu, beberapamakalah penyelidikan dan aplikasi yang ada akan dikaji sebagai sumber pengumpulan keperluan yang membantudalam merancang fungsi dan ciri aplikasi web yang mematuhi kerentanansuntikan OWASP A03. Selain itu, pengembangan aplikasi web akan didokumentasikan. Selain itu, hasil pengujian pada aplikasi web akan dianalisis dalam laporan ini. Dari hasil positif yang diperolehdaripadapenguji yang terlibat dalam ujian kebolegunaan makmal dan ujian kebolegunaan menggunakan skala kebolegunaan sistem, dipercayai bahawa projek ini dapat membantumeningkatkan pemahaman pembangun web mengenai kerentanansuntikan OWASP A03.

Table of Contents

ACKNOWLEDGEMENT	1
ABSTRACT	2
ABSTRAK	3
List of Figures	9
List of Tables	13
CHAPTER ONE: INTRODUCTION	14
1.1 Background	14
1.2 Problem Statement	16
1.4 Project Scopes	17
1.5 Significant of the Project	17
1.6 Project Outline	17
CHAPTER TWO: LITERATURE REVIEW	19
2.1 Introduction	19
2.2 OWASP Top 10 A03 - Injection Vulnerability	19
2.2.1 Whitelisting and Blacklisting	20
2.2.2 Prepared Statement/Parameterized Queries	22
2.2.3 Stored Procedure	23
2.2.4 Defensive Coding Practice	24
2.2.5 Taint Based Approach / Taint Analysis	24
2.2.6 Proxy Filters	25
2.2.7 Using built in function	26
2.2.9 Low Privileges	28
2.2.10 Output Escaping	29
2.3 Secure Web Development using OWASP as a guidelines	29

2.4 Review on the Existing Web Application complied with OWASP.....	32
2.4.1 OWASP WebGoat	33
2.4.3 OWASP Juice Shop	40
2.4.4 Hacksplaining.com.....	42
2.5 Comparison of defensive technique in existing web application complied with OWASP.....	45
<i>CHAPTER THREE: METHODOLOGY.....</i>	50
3.1 Introduction.....	50
3.2 Requirement Analysis.....	51
3.2.1 Common Weakness Enumeration (CWE).....	51
3.2.2 Software and Hardware Requirement.....	55
3.3 System Design.....	56
3.3.1 User Interface	56
3.3.2 UML Diagram	58
3.3.2.1 Use Case Diagram.....	58
3.3.2.2 Class Diagram.....	59
3.3.2.3 Sequence Diagram	59
3.3.3 Data Dictionary	62
3.4 Implementation	63
3.4.1 Extension used in Microsoft Visual Studio Code.....	63
3.5 Testing.....	64
3.5.1 Laboratory Usability Testing.....	65
3.5.2 Usability Testing Using System Usability Scale (SUS)	65
3.6 Deployment.....	65
3.7 Maintenance	66

3.8 Summary.....	66
CHAPTER FOUR: IMPLEMENTATION.....	67
4.1 Introduction.....	67
4.2 Software and Tool Used.....	67
4.3 Environment Set up	68
4.4 Implementation of the features	68
4.4.1 Login Page	69
4.4.2 Registration Page	69
4.4.3 Homepage	70
4.4.4 Whitelisting and Blacklisting Lesson Page.....	70
4.4.5 Prepared Statement/Parameterized Queries Lesson Page.....	71
4.4.6 Stored Procedure Lesson Page	72
4.4.7 Defensive Coding Practice Lesson Page.....	72
4.4.8 Taint Based Approach Lesson Page.....	73
4.4.9 Proxy Filters Lesson Page	74
4.4.10 Using Built in Function Lesson Page.....	74
4.4.11 Instruction Set Randomization Lesson Page.....	75
4.4.12 Low Privileges Lesson Page	76
4.4.13 Output Escaping Lesson Page.....	76
4.4.14 Editor Page	77
4.5 Core Features Explanation	78
4.5.1 Login.....	78
4.5.3 Lesson List	82
4.5.5 Editor Page	83

4.5.6 Editor Page backend.....	84
4.6 Web Application Hosting	85
4.6.1 Hosting of Laravel Part of web application.....	85
4.6.2 Hosting of Editor Page.....	87
4.6.3 Database hosting	88
4.7 Summary.....	88
<i>CHAPTER FIVE: TESTING.....</i>	<i>90</i>
5.1 Introduction.....	90
5.2 Laboratory Usability Testing.....	90
5.2.1 Test Environment.....	90
5.2.2 Test Scenario	91
5.2.3 Test Metrics	92
5.2.4 Analysis of Laboratory Usability Testing Result	94
5.3.1 Analysis of demographic information of the testers	97
5.3.2 Analysis of System Usability Scale (SUS)	99
5.4 Discussion.....	108
5.5 Summary.....	109
<i>CHAPTER SIX: CONCLUSION.....</i>	<i>110</i>
6.1 Introduction.....	110
6.2 Project Achievement.....	110
6.3 Project Limitations	111
6.4 Future works	111
6.5 Summary.....	112

REFERENCES 113
APPENDIX 117

List of Figures

Figure 2.1: Example of Blacklisting detection code	21
Figure 2.2: Example of whitelisting detection code	22
Figure 2.3: Example source code of prepared statement / parameterized query	23
Figure 2.4: Example of stored procedure using sp_executesql.....	23
Figure 2.5: Example of regular expression to will only accept letters and white space	24
Figure 2.6: Proposed model of proxy filter.....	26
Figure 2.7: Example of PHP code that use mysqli_real_escape_string function	27
Figure 2.8: filter_var function to sanitize user input by FILTER_VALIDATE_EMAIL and FILTER_VALIDATE_URL parameter.....	27
Figure 2.9: SQL query before implementing ISR.....	28
Figure 2.10: SQL query after implementing ISR.....	28
Figure 2.11: SQL queries that grant only 'SELECT' privilege to the user.....	29
Figure 2.12: Proposed idea by Lala, Kumar and T. (2021)	29
Figure 2.13: Binding function that bind queries to query string Lala et al.(2021)	30
Figure 2.14: Turn off the multi SQL statement in the SQL connection	31
Figure 2.15: Uses of session to redirect user back to login page if the user does not login	31
Figure 2.16: restrict only the question and the option of the question will be sent to client side	32
Figure 2.17: Encrypt the password into SHA-256.....	32
Figure 2.18: Rendering the home page, with login information sent as a JSON object to the EJS viewer.	32
Figure 2.19: Explanation of the SQL Injection vulnerability and the list of goals	33
Figure 2.20: Explanation on SQL and learn by doing section on the webgoat	35
Figure 2.21: Example of assignment solved by the user	35

Figure 2.22: Example of safe code written by the user after having mitigation lesson.....	35
Figure 2.23: User can change the security level and the PHPIDS of the web application.	37
Figure 2.24: User Interface for the user to attack in low security mode.....	37
Figure 2.25: User Interface for the user to attack in medium security mode.....	37
Figure 2.26: View Button in every vulnerability page	38
Figure 2.27: side by side code comparing on the SQL injection vulnerability in medium and low security level	39
Figure 2.28: Homepage of OWASP Juice Shop.....	40
Figure 2.29: Example of performing SQL Injection attack on the OWASP Juice Shop.....	41
Figure 2.30: score board of the Juice Shop.....	42
Figure 2.31: Visualization on explanation on SQL Injection	43
Figure 2.32: example of interactive prompt.....	43
Figure 2.33: Examples of precaution ways and code example	44
Figure 2.34: Example of Safer Python Code against SQL Injection	45
Figure 3.1: Waterfall Model	51
Figure 3.2: Introduction Page of Defensive Technique lesson	57
Figure 3.3: Second page of Defensive technique lesson.....	57
Figure 3.4: Use Case Diagram of the Web Application	58
Figure 3.5: Class Diagram of the web application.....	59
Figure 3.6: Sequence diagram for user registration and user login	60
Figure 3.7: Sequence diagram for user to choose a lesson from the lesson list.....	60
Figure 3.8: Sequence diagram during user take a lesson	61
Figure 4.1: Login Page.....	69
Figure 4.2: Registration Page.....	70
Figure 4.3: Homepage.....	70

Figure 4.4: Whitelisting and Blacklisting Lesson Page	71
Figure 4.5: Prepared Statement/Parameterized Queries Lesson Page	71
Figure 4.6: Stored Procedure Lesson Page	72
Figure 4.7: Defensive Coding Practice Lesson Page	73
Figure 4.8: Taint Based Approach Lesson Page	73
Figure 4.9: Proxy Filters Lesson Page	74
Figure 4.10: Using Built in Function Lesson Page	75
Figure 4.11: Instruction Set Randomization Lesson Page	75
Figure 4.12: Low Privileges Lesson Page.....	76
Figure 4.13: Output Escaping Lesson Page	77
Figure 4.14: Editor Page	77
Figure 4.15: Input Field of User's Email Address in Login Page	78
Figure 4.16: Input Field of User's Email Address in Login Page	79
Figure 4.17: Login button and remember me in Login Page.....	80
Figure 4.18: Registration Form in Registration Page	81
Figure 4.19: Lesson List in the side navigation bar.....	82
Figure 4.20: Code to display lesson content from system database.	82
Figure 4.21: Code that shows the editor interface	83
Figure 4.22: JavaScript that is responsible to handle the HTML text input by the user.....	84
Figure 4.23: Deploy of GitHub Repository in Heroku	86
Figure 4.24: Deployment option of Web Application	86
Figure 4.25: Build log of web application	87
Figure 4.26: Web application deployed succesfully on Heroku Platform.....	87
Figure 4.27: PHP code uploaded to the htdocs folder.	88
Figure 4.28: SQL table structure of the Web Application	88

Figure 5.1: Year of study of respondents	98
Figure 5.2: Response of “Have you heard about OWASP A03 Injection vulnerability before?” question	98
Figure 5.3: Response of “I think that I would like to use this application frequently to learn” question	99
Figure 5.4: Response of “I found the application unnecessarily complex” question	100
Figure 5.5: Response of “I thought the application was easy to use for learning” question .	101
Figure 5.6: Response of “I think that I would need the support of a technical person to be able to use this application for learning” question	102
Figure 5.7: Response of “I found the various functions in this application were well integrated for learning” question	103
Figure 5.8: Response of “I thought there was too much inconsistency in this application for learning” question	104
Figure 5.9: Response of “I would imagine that most people would learn using this application very quickly” question	105
Figure 5.10: Response of “I found the application very cumbersome to use for learning.” question	106
Figure 5.11: Response of “I felt very confident using the application for learning.” question	107
Figure 5.12: Response of “I can learn a lot of things in this application for learning” question	108

List of Tables

Table 2.1: Comparison of defensive techniques adopted by existing web application	48
Table 3.1: CWE Weakness mapped to Defensive Techniques.....	52
Table 3.2: Hardware Requirement of Microsoft Visual Studio.....	55
Table 3.3: Hardware Specification of Developing Device	56
Table 3.4: Data Dictionary for User	62
Table 3.5: Data Dictionary for Lesson.....	62
Table 3.6: Data Dictionary for users_dummy.....	63
Table 5.1: Test metrics on evaluating testers' written code.....	92
Table 5.2: Evaluation of Testers' written code with evaluation metrics	94
Table 6.1: Objectives that achieved in this project	110

CHAPTER ONE: INTRODUCTION

1.1 Background

Starting from a few decades ago, information technology has grown exponentially. Various information technology like World Wide Web (WWW) had merged in the current market. Talking about the world wide web, Jacksi and Abass (2019) defined web as “Sharing of information, document and resource between users across the internet and also it is as tunnel to access storage data on servers and display it on client by browser through the internet, information is related to on client by browser through the internet, information is related to gather by links that include text, picture, sound, video.” The very first web is developed by the British scientist, Tim Berners-Lee. The concept of Web1.0 was then designed as the first model of web which can only display information to the client without posting anything. In 1999, Darcy DiNucci announced the Web 2.0 which became famous in the year 2004. In web 2.0, it has the capability to perform read and write operation and is known as Participative and Social Web. One of the most famous Web 2.0 applications is Facebook where the clients can connect and interact with each other across the internet. In 2015, Web 2.0 is further upgraded to Web 3.0, established as semantic network or in the other words, “Internet of Things”.

As web technologies expand, web application security becomes one of the biggest concerns especially for large organizations or companies. This is because there are getting more and more illegal cyber-attacks had been carried out towards the web application for stealing clients’ personal data, like Personal Identifiable Information (PII) and Personal Health Information (PHI) (Singh et al., 2021) by cybercriminal like Black Hat Hackers. The personal data of the client might be used by the hacker to do illegal activity, for example, uses of victims’ identity to pretend as the victim, and carry out scam activities among the victims’ friend zones. Thus, it is very necessary to have the proper and effective way to prevent cyber-attacks to minimize the chance for the leakage of the web users’ personal data.

To accommodate this situation, a non-profit organization, Open Web Application Security Project (OWASP) has come up with a standard which can be used as a guideline by the web developer to enhance their web application's security. Those standards or guidelines are also known as OWASP Top 10.

Lala et al. (2021) claims that currently there is an overwhelming number of vulnerability in any of the web application during development phase. OWASP Top 10 provides resources, documentation, and tools that assist security analysts, pen testers, developers, and hobbyists in addressing vulnerabilities. Hence, having OWASP Top 10 as a security guideline for the developer to patch the vulnerabilities listed in OWASP Top 10 is the most efficient way to enhance the web security.

OWASP Top 10 contains top 10 common web application security risks in the current market. This standard will be updated once in four year. According to the latest OWASP version, which is OWASP Top 10 2021(Introduction, 2021), the rank of the common web application security risks are:

- A01: Broken Access Control
- A02: Cryptographic Failures
- A03: Injection
- A04: Insecure Design
- A05: Security Misconfiguration
- A06: Vulnerable and Outdated Components
- A07: Identification and Authentication Failures
- A08: Software and Data Integrity Failures
- A09: Security Logging and Monitoring Failures
- A10: Server-Side Request Forgery

1.2 Problem Statement

In this era with exponential growth of information technologies, there are more and more web applications developed by the web developer which are widely being used by the public. Those web applications might require user's personal data such as name, ID number or even bank card number or bank account for web application like e-commerce application. If the web security of those web applications is not high enough, hackers can attack the web application through injection attack. According to OWASP (Introduction, 2021), the injection attack is categorized as A03 Injection vulnerability which rank at the top 3 in the OWASP Top 10. The most common injection attack in A03 injection vulnerability are SQL Injection and Cross-Site Scripting (XSS) injection(Williams, Jmanico, & Kingthorin, 2021). Through the injection attack like SQL Injection, the attacker can have full access to the web application database. This situation may lead to the exposure of the users' personal data or their sensitive data. Sensitive data like bank card numbers might be used by the attacker to transfer the users' wealth. Thus, it is very important to protect user personal data and their sensitive. Besides, another type of injection attack is XSS injection. In XSS injection attack, the attacker can do malicious actions like malware spreading, cookies-stealing, session hijacking to steal users' credentials or even redirect user to a malicious web application (Farea et al., 2021). Hence, it is very necessary to have the knowledge on A03 Injection vulnerability so that this vulnerability can be prevented and prohibited.

1.3 Objectives

- To design and develop a vulnerable web application of A03 Injection detection for educational purposes.
- To test the effectiveness of the implemented web application in improving knowledge on A03 Injection.

1.4 Project Scopes

Develop a web application prototype that complies with OWASP Top 10 A03 – Injection vulnerability. This web application is aimed to deliver knowledges on the OWASP A03 – injection vulnerability and provides lessons to the user on the implement the defensive techniques to defense against injection attack.

1.5 Significant of the Project

This project aims to develop a web application prototype complying with OWASP Top 10 A03 – Injection vulnerability. This web application prototype aims to raise awareness about injection vulnerabilities and educate users about the impact of such vulnerabilities. Web application prototype is aimed to teach about injection attacks, including how they occur and their potential consequences. The prototype also provides a hands-on session where users can practice attacking the web application correctly. In case users encounter difficulties or are unsure about the next steps, the prototype offers features such as hints or tips to guide them in correctly attacking the web application. Upon completion of the hands-on session, the web application prototype suggests the best practices to users to prevent the specific vulnerability they targeted during the lesson. Through this interactive experience, the web application prototype will raise user's awareness of injection vulnerabilities and learn how to prevent them while developing their own web applications.

1.6 Project Outline

The outline of the project is organized as follows:

- Chapter one introduces the background, problem statement, objectives and significant of the project.

- Chapter two includes the literature review made to gather the information related to the project.
- Chapter three discuss on the methodology used, requirement analysis, system design and prototyping of the proposed web application.
- Chapter four discuss about development process of the web application, including the screenshot of the web application developed as well as the core coding of the web application.
- Chapter five shows the result of how web application developed in achieving the objectives of this project in testing process.
- Chapter six shares about the conclusion, limitation and future work on this project

CHAPTER TWO: LITERATURE REVIEW

2.1 Introduction

In chapter 2, several coursework and research papers related to the OWASP Top 10 which are done by other researchers will be reviewed. From reviewing through the coursework and research paper, we can get the knowledge which is related to the project topics. That knowledge will be useful and act as a guideline for us throughout this project. Other than reviewing coursework and research papers, several web applications which are compiled to the OWASP Top 10 vulnerabilities will also be reviewed. The applications that will be reviewed in this chapter are OWASP Webgoat, Damn Vulnerable Web Application, OWASP Juice Shop, Free Online Bank Web by Micro Focus Fortify and Hacksplaning.

After reviewing the coursework and the existing web applications, a summary for each of the reviewed items will be extracted. For the reviewing of the existing web application, a comparative table will be made to compare the offering features of each of the existing web application. Those extracted information will be used in analyzing and designing the requirements and features that should be included in this project. To conclude this chapter, a summary conclusion will be done at the end of the chapter.

2.2 OWASP Top 10 A03 - Injection Vulnerability

According to Marashdeh et al. (2021), SQL injection refers to the type of attack towards the web application database through injecting malicious code to gain the information or the data existing in the database without proper authentication. Other than that, the attacker can perform dangerous actions such as retrieving data without authorization from the database, deleting MySQL tables and others. Those dangerous actions are irreversible and will cost permanent damage to the web application that had been attacked.