EFFECTIVENESS OF STRUCTURED QUERY LANGUAGE INJECTION ATTACKS DETECTION MECHANISMS

NURUL ZAWIYAH BINTI MOHAMAD

A project report submitted in fulfilment of the
requirements for the award of the degree of
Master of Science (Computer Science)

Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia

OCTOBER 2008

ABSTRACT

Database security is one of the most essential factors in keeping stored information safe. These days, web applications are used widely as a meddler between computer users. Web applications are also used mostly by e-commerce companies, and these types of applications need a secured database in order to keep sensitive and confidential information. Since SQL injection attacks occurred as a new way of accessing database through the application rather than directly through the database itself, they have become popular among hackers and malicious users. Many prevention and detection mechanisms are developed to handle this problem but these mechanisms have their limitations. In this study, two mechanisms, AMNESIA and SQL Guard are adopted for a practical evaluation to search for the better technique in detecting SQL injection attacks. These techniques will be called Technique A and Technique B respectively and will be evaluated on their effectiveness and efficiency using precision and recall measure against two web applications, Mekar and myMarket. The study will show that Technique B is a better approach on detecting SQL injection attacks.

ABSTRAK

Keselamatan pangkalan data adalah salah satu faktor yang amat penting dalam penyimpanan maklumat. Aplikasi yang berasaskan web digunakan oleh kebanyakan syarikat – syarikat e-dagang dan aplikasi seperti ini memerlukan pangkalan data yang selamat kerana melibatkan maklumat peribadi dan juga maklumat sensitif para pengguna lain. Serangan suntikan SQL adalah salah satu cara untuk mencerobohi pangkalan data melalui laman web, dan cara ini telah terkenal di kalangan penggodam untuk mendapatkan maklumat peribadi pengguna lain, dan digunakan untuk kepentingan diri. Pelbagai teknik telah diperkenalkan untuk mengesan serangan suntikan SQL ini, namun teknik-teknik ini masih mempunyai pembatasan tersendiri. Di dalam laporan ini, dua teknik, iaitu AMNESIA dan SQL Guard telah dipilih untuk diujikaji dari segi keberkesanan dan kecekapan dalam mengesan serangan suntikan SQL. Teknik yang terpilih, dinamakan teknik A dan teknik B akan diujikaji ke atas dua aplikasi web, iaitu Mekar dan myMarket. Kajian ini menunjukkan bahawa teknik B adalah teknik yang lebih berkesan dalam mengecam serangan suntikan SQL.

TABLE OF CONTENTS

CHAPTER 1

PROJECT OVERVIEW

1.1 Introduction

Database security is the degree to which all data is fully protected from tampering or unauthorized acts. Security vulnerability, security threat and security risk are the menaces to database. In security threat, individuals are one of its types of threat, where individuals intentionally or unintentionally inflict damage, violation, or destruction to all or any of the database environment components. These individuals include hackers, terrorists, organized criminals, and employees.

In this 21st century, web applications are extensively used for generating information, distributing information from an organization to the users over a network, and also as a platform to run e-Commerce websites. These web applications usually have a back-end database to keep the customers' information and all relevant information about the customers, including credit card details, and private data. As these web applications usually accept data from users and bring these data to access the

back-end database, these types of applications carry the possibility of being exposed to the SQL injection attacks.

An SQL injection attack (SQLIA) is a subset of the un-sanitized input vulnerability and occurs when an attacker attempts to change the logic, semantics or syntax of a legitimate SQL statement by inserting new SQL keywords or operators into the statement. (Muthuprasanna *et al.*, 2006). It is also known as a technique that abuses the security vulnerability occurring in the database layer of an application. The attack happens when an attacker is able to insert a series of SQL statements into a query by manipulating input data into the application. Once the attacker successfully injects the attack into the database, the database will be susceptible of being altered, extracted or even dropped. SQL injection happens on web application (ASP, PHP, JSP etc.) itself rather than on the web server or services running in the operating system. According to Muthuprasanna *et al.* (2006), very little emphasis is laid on securing these applications.

1.2 Problem Background

SQL injection attacks have been a serious problem since 2002. Studies by Thomas and William (2007) showed that since 2002, 10% of total cyber vulnerabilities were SQL injection vulnerabilities. The percentage keeps on increasing as developers discover ways to prevent and detect the SQL injection attacks, resulting to attackers developing a wide array of attack techniques that can be used to exploit SQL injection vulnerabilities. The main problem in preventing and detecting any type of SQL injection attacks is the development of a defense mechanism which guarantee no false positives in detecting the attacks (high precision) while ensuring the actual attacks injected are detected (high recall).

There are seven main types of SQL injection attacks (Halfond and Orso, 2008). They are tautologies, union queries, piggybacked queries, malformed queries, inference, alternate encodings and leveraging stored procedures. Each of these types has different techniques to launch an SQL injection attack. They also have their goal; which is the intention of the attacker, whether to add or modify data, or to extract data for personal use and many more.

According to Muthuprasanna *et al.*, (2006) there is no known fool-proof defense against the SQLIAs. The researchers in this field have built several mechanisms, or framework to handle SQLIAs. These mechanisms have been proven effective and efficient when tested, but they do have their own limitations. Some of these mechanisms are for the use during static analysis (McClure and Kruger (2005); Halfond *et al.* (2006); Fu *et al.* (2007)), where the analysis are carried out during coding of the application to identify any vulnerabilities of the application.

Besides static analysis, a number of mechanisms have been developed to be used during both static and dynamic analysis (Halfond and Orso (2005), Muthuprasanna *et al* (2006), Wei *et al.* (2006) and Kosuga *et al.* (2007)). These mechanisms analyze codes during the static analysis, and validate the coding during dynamic analysis, which is the execution of attacks towards the analyzed codes.

Halfond *et al.* (2006) mentioned that the cause of SQL injection vulnerabilities is relatively simple and well understood; which is insufficient validation of user input. Alfantookh (2004) and Lin and Chen (2006) have developed a mechanism that does the input validation before it is being sent to the web server.

From the works of Halfond *et al.* (2006), an analytical comparison and countermeasures have been carried out to compare existing techniques or mechanisms that exist. The techniques are divided into *detection-focused,* techniques that detect attacks during runtime, and *prevention-focused*, techniques that statically identify vulnerabilities in the code. It is found that a *combination* of static and dynamic analysis *detects* more attack types instead of other techniques. For the purpose of this study, two mechanisms (AMNESIA by Halfond and Orso (2005), and SQLGuard by Buehrer *et al.* (2005)) have been selected to be analyzed and implemented in a new environment. However, Halfond *et al.* (2006) stated that they did not take precision into account in this evaluation. Many of the techniques considered are based on some conservative analysis or assumptions that can result in false positives.

According to Halfond *et al.* (2006), future evaluation work should focus on evaluating the techniques' precision and effectiveness in practice. There has not been an effectiveness and efficiency evaluation carried out to ensure the techniques are fully effective and efficient in practice, although some are claimed to be 100% effective in their experimental result. Empirical evaluations would allow for comparing the performance of the different techniques when they are subjected to real-world attacks and legitimate inputs.

1.3 Problem Statement

The purpose of this thesis is to provide an experimental analysis and comprehensive comparison of defense mechanisms based on the selected mechanisms. The key factors to be evaluated in the comparison are the correctness, effectiveness and the efficiency of the mechanisms. The main problem in preventing and detecting any

type of SQL injection attacks is the development of a defense mechanism which guarantee no false positives in detecting the attacks (high precision) while ensuring the actual attacks injected are detected (high recall). The problem statements emphasizing the goal of this study are: *which SQL injection attack detection approaches is effective and efficient to successfully detect SQL injection attacks?*

1.4 Project Aim

The aim of this project is to determine the effective and efficient detection technique that is able to detect the SQL injection attacks.

1.5 Objectives

In order to accomplish the hypothesis of the study, three objectives have been identified as stated below:

1. To investigate and classify SQL injection attacks by types of attacks.
2. To study the techniques of both mechanisms and create the algorithms.
3. To investigate the effectiveness and efficiency of both the SQL injection attacks detection techniques.

1.6 Project Scope

The scopes of this project are defined as follows:

1. The techniques are developed in a PHP language.
2. The mechanisms that will be compared in this study are AMNESIA and SQL Guard.
3. The web applications involved for experiment are *mekar* and *myMarket.*
4. The SQL query keyword that will be affected in this study is SELECT statements only.

1.7 Significance of the Project

This project will investigate the effectiveness and efficiency of detection approaches in order to spot an incoming of injected attacks through web applications. The analysis will be carried out and to compare which technique works better and more effective. The result of this study can be used to verify the effectiveness and efficiency of the tested approaches and will contribute in future works for future improvements.

1.8 Report Organization

This thesis is divided into five chapters. Chapter 1 will introduce the basic information of SQL injection attacks, problem background, problem statement, objectives and scope. Chapter 2 is a literature review and will explain in-depth on SQL injection attacks and their prevention or detection approaches. Chapter 3 will explain the way this project is conducted, and the methodology that is used to handle the evaluation and the testing. Chapter 4 will show the experiment results from what has been done, and Chapter 5 will summarize and conclude on the thesis.

REFERENCES

Alfantookh A. A., (2004). An Automated Universal Server Level Solution for Sql Injection Security Flaw 2004. *IEEE,* pg 131- 135

Anley C., (2002). Advanced SQL Injection In SQL Server Applications. White paper, Next Generation Security Software Ltd., 2002.

Bertino E., Kamra A., and Early P. J., (2007). Profiling Database Applications to Detects SQL Injection Attacks. *ISBN 1-4244-1338-6/07.IEEE*

Buehrer G. T., Weide B. W., and Sivilotti P. A. G., (2005). Using Parse Tree Validation to SQL Injection Attacks. In *International Workshop on Software Engineering and Middleware (SEM), 2005.*

Fu X., Lu X., Petsverger B., Chen S., Qian K., and Tao L. (2007). A Static Analysis Framework for Detecting SQL Injection Vulnerabilities. *31st Annual International Computer Software and Applications Conference (COMPSAC 2007). IEEE*

Halfond W. G., and Orso A., (2005). AMNESIA: Analysis and Monitoring for NEutralizing SQL-Injection Attacks. In *Proceedings of the IEEE and ACM International Conference on Automated Software Engineering (ASE 2005).* Long Beach, CA, USA, Nov 2005.

Halfond W.G., Viegas J., and Orso A., (2006). A Classification of SQL Injection Attacks and Countermeasures. In *Proceedings of IEEE International Symposium Secure Software Engineering,* Mar. 2006.

Halfond William G. and Orso A., (2008). WASP: Protecting Web Applications Using Positive Tainting and Syntax-Aware Evaluation, *IEEE Transactions On Software Engineering,* Vol. 34, No. 1, January/February ,2008 pg 65-81

Howard M. and LeBlanc D., (2003). Writing Secure Code. Microsoft Press, Redmond, Washington, second edition, 2003.

Huang Y., Yu F., Hang C., Tsai C. H., Lee D. T., and Kuo S. Y., (2004). Securing Web Application Code by Static Analysis and Runtime Protection. In *Proceedings of the 12th International World Wide Web Conference (WWW 04)*, May 2004.

Kosuga Y., Kono K., Hanaoka M., Hishiyama M., and Takahama Y., (2007). Sania: Syntactic and Semantic Analysis for Automated Testing against SQL Injection, *23rd Annual Computer Security Applications Conference,* pg 107 - 116

S. Labs. SQL Injection. White paper, SPI Dynamics, Inc., (2002). http://www.spidynamics.com/assets/documents/WhitepaperSQLInjection.pdf.

Lin J., and Chen J., (2006). An Automatic Revised Tool for Anti-malicious Injection In *Proceedings of The Sixth IEEE International Conference on Computer and Information Technology (CIT'06)*

Livshits V. B. and Lam M. S., (2005). Finding Security Errors in Java Programs with Static Analysis. In *Proceedings of the 14th Usenix Security Symposium,* pages 271–286, Aug. 2005.

Mackay C. A., (2005). SQL Injection Attacks and Some Tips on How to Prevent Them. Technical report, The Code Project, January 2005. http://www.codeproject.com/cs/ database/SqlInjectionAttacks.asp.

McClure R., and Kruger I., (2005). SQL DOM: Compile Time Checking of Dynamic SQL Statements. In *Proceedings of the 27th International Conference on Software Engineering (ICSE 05),* pages 88–96, 2005.

Muthuprasanna M., Wei K., Kothari S. (2006). Eliminating SQL Injection Attacks - A Transparent Defense Mechanism. *Eighth IEEE International Symposium on Web Site Evolution (WSE'06).IEEE*

Spett K., (2003). Blind sql injection. White paper, SPI Dynamics, Inc., 2003. http://www.spidynamics.com/whitepapers/ Blind SQLInjection.pdf.

Su Z., and Wassermann G. (2006). The Essence of Command Injection Attacks in Web Applications. In *The 33rd Annual Symposium on Principles of Programming Languages (POPL 2006),* Jan. 2006.

Thomas S., and Williams L. (2007). Using Automated Fix Generation to Secure SQL Statements. *Third International Workshop on Software Engineering for Secure Systems (SESS'07)*

Ullrich J. B., Lam J., (2008). Defacing websites via SQL injection. *Network Security Magazine*, pg 9-10

Valeur F., Mutz D., and Vigna G. (2005). A Learning-Based Approach to the Detection of SQL Attacks. In *Proceedings of the Conference on Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA),* Vienna, Austria, July 2005.

Wei K., Muthuprasanna M., Kothari S. (2006). Preventing SQL Injection Attacks in Stored Procedures. In *Proceedings of the 2006 Australian Software Engineering Conference (ASWEC'06). IEEE*