All About SQL Injection Attacks

Daniel Thomas Loughran
dtloughran1@buffs.wtamu.edu

Mayar Kefah Salih
Mksalih1@buffs.wtamu.edu

Vinitha Hannah Subburaj
vsubburaj@wtamu.edu

School of Engineering, Computer Science and Mathematics
WTAMU Box 60767, West Texas A&M University, Canyon, TX 79016

Abstract - With advancements in Internet technologies, there is an increasing growth of applications that are web based. With smaller software development cycles and faster delivery, security has become an important issue. There are many types of security attacks that are made on Web applications and SQL injection attack is one type of an attack. Recently, studies have shown that more and more web applications are getting attacked by different types of SQL injection attacks. To effectively detect and prevent these attacks, a deeper understanding on the different types of SQL injection attacks, the nature of the attacker, and the mechanism used is very important. This paper discusses details that one would need to understand all about SQL injection attacks. This paper presents a detailed study of most recent SQL injection attacks on web applications, SQL injection prevention and detection mechanisms. The classification of different types of SQL injection attacks, prevention and detection mechanisms discussed in this paper highlights the need for future improvements in the detection and prevention mechanisms to secure web applications from SQL injection attacks.

Keywords

SQL injection attacks, detection, prevention, web application

1 INTRODUCTION

SQL Injections (SQLI) are attacks in which an attacker sends SQL statements to a SQL database; the SQL statements allow the attacker to control what the server does. By doing so, an attacker can take full control over the server [1]. The attackers can send code by simply inputting a SQL statement in place of a username and password. For example, an attacker could simply provide the server a statement that looks like:

SELECT CustomerName FROM Customers;

Assuming no security measures was taken towards the creation of the web application, this SQL statement would list all customers in the Customer database. SQL statements can also allow attackers to gain administrator rights to the database as well, which means the attacker can add, edit, or delete data with nothing stopping them from doing it. However, the only way an attacker can use this exploit is by looking for inputs on the website in question. The login boxes or inputs are the places where the SQL statements are typed, which then sends the malicious code to be ran on the server hosting the database [23].

The resulting impact of SQL injection attacks can vary. For example, data could be destroyed, stolen, manipulated, the server hosting the database could be harmed, or the attacker could be waiting for users to logon on the web application to wait for more user input before doing any harm [22].

The rest of the paper is organized as follows. Section 2 discusses the different types of SQLI attacks made on web applications. The different types of hackers and their behaviors are discussed in Section 3. Section 4 discusses and tabulates the recent SQLI attacks made on web applications, the year the attack was made, the attacker's name, the consequences of the attack, and the purpose of the attack. Section 5 discusses the prevention techniques used to prevent against SQLI attacks. Section 6 discusses the detection techniques used to detect SQLI attacks. Section 7 concludes this paper with a discussion on the proposed future work.

2 TYPES OF SQLI ATTACKS

There are several types of SQL injection attacks that could be deployed. Depending on what the attacker's goal is, the SQL injections are sent to the server one after another or all at the same time. The different types of SQLI attack methods are tautologies, illegal/logically incorrect queries, union query, piggy-backed queries, stored procedures, inference, timing attack, and alternating encoding.

The first type of SQL injection is tautologies. Tautologies are attacks in which SQL statements are written to be always true [23]. Consequently, it bypasses the authentication the server does. The main element to get this attack to work correctly is the use of the WHERE statement. Attackers use the WHERE statement in a way that makes the outcome to be true when the server database runs the code. A possible outcome is the server displaying all the data to the attacker.

The next type of attack is Illegal/Logically Incorrect Queries. These queries enable attackers to gain information about the back-end of the database. This information includes the structure of the database and the type of database being used [10]. Essentially, this is an attacker's version of scouting before they perform the actual attack. This preliminary attack works when the attacker makes the server throw an error. If the error thrown contains any information on the code, it can assist the attacker by showing the vulnerable parameters [23]. Additionally, this type of attack can also provide the name of the table if the attackers provide the correct SQL statement, and the programmer did not hide that information.

The next type of attack is a Union Query SQL injection. The main part of this attack is the UNION SQL statement. UNION in SQL allows programmers to combine the result of two different SELECT statements. Atefeh Tajpour, et al [23] gives a good example of this attack. For example:

```
SELECT Name, Phone FROM User WHERE Id=1 UNION

ALL SELECT creditCardNumber, 1 FROM CreditCarTable
```

This example returns the result of the first query and every credit card user in the database.

Then, there are Piggy-backed Queries. These are queries where the attacker uses a ";" to add an extra query to the first one. The key to making this attack work is making sure that the first query the server sees is a legitimate one. The legitimate query is usually followed by illegitimate queries, which in turn causes the server to run all the illegitimate queries [23]. Thus, bypassing the need for the attacker to sign into the database as a user.

In addition, there is a SQL injection attack called stored procedures. A stored procedure SQL injection occurs is when an attacker attempts to use the stored procedures within the database they are attacking [18]. For this to work properly, attackers must find out what type of database is being used in the web application. This is where illegal/logically incorrect queries come in handy, since those attacks provide the attacker the information, a stored procedure requires to be effective.

Another type of SQL injection attacks an attacker can use is called inference attacks; there are two methods that fall under this category. The first is blind injection; blind injection is used when the attacker sends SQL statements that can return true or false values [23]. Essentially, the attacker is blindly looking for a weak point in the application; however, this only works when the application does not use input validation. The second method is called timing attacks. Timing attacks are usually carried out by attackers observing the timing delays in the database response time to gain more information from the database [23]. This is similar to blind injections in that the attacker is still guessing to find a weak spot. The only difference between them is that the attacker focuses on the response time rather than what the server returns.

The last SQLI attack type is called alternate encoding. This technique works by confusing the database with different encoding in the SQL statements. For example, attackers could use hexadecimal, ASCII, or Unicode in a SQL statement. By doing this, attackers will bypass any basic validation done by the application. Consequently, SQL commands could be encoded in hexadecimal and the validation would not catch it, which means the server would run any commands the attacker wants it to run.

3 TYPES OF HACKERS

Attacking websites became more popular due to the huge growth in networking systems. There are different types of network users who are classified based upon their use of Internet facility. Scholars, searchers, and other ordinary users who aim to get the benefit out of the Internet are generally called end users. On the other hand, we have extraordinary users who tend to make use of the Internet for unusual purposes.

Focusing on those users alongside with the amount of damage the website experiences, helps classify the skill of the attacker and the effectiveness of the cracker's attack. If we determine how skilled the attacker is, we can deflect any foreseen attacks in later times. Thus, prevention becomes easier.

Xu and Chenghong [26] categorize hackers into three different groups called "White-hat hackers," "Grey-hat hackers," and "Black-hat hackers." White hats also known as security analysts are very knowledgeable individuals who have professional hacker skills, but still use those skills for defensive purposes. On the other side of the spectrum, we have Black-hat hackers; black-hat hackers are the people who attack with malice in their heart. They use an application's vulnerability to steal or destroy data. Neutral individuals who have extraordinary skills are known as gray hats, they work both defensively and offensively in various times.

Specific reasons as to why white-hat hackers and black-hat hackers impose SQLI attacks are: finding vulnerable inputs, finding the type of database the web application is using, finding out a table's name, column names and data types, stealing data, change information, shutting down the server hosting the database, bypass security measures, and running commands remotely with elevated permissions [11].

Depending on the reason for attacking a web application, attackers will choose the SQL injection attack based on what their goals are. For example, if the attacker wants to find out the table name or if there is any vulnerable input, then they will probably use illegal/logically incorrect queries to get the information they want.

4 WEB APPLICATIONS ATTACKED BY SQLIA

Web developers have been using PHP for building web applications for many years now. Facebook and Wikipedia were originally developed using PHP [14]. A very simple programming language such as PHP, which is closer to natural languages more than it is to machine language, can be easily manipulated by the extraordinary web users. SQL injection is one way of causing damage to websites and web applications, which is believed to be first applied in programs written and constructed in PHP. SQL injection works as a dose that is given to a certain program causing it some damage and rarely to collapse.

4.1 Recent SQLI Attacks in Web Application

In Feb 2017, ePolicy Orchestrator database was attacked by sending a crafted HTTP POST request to the targeted system. The application company, indeed, experienced a hard time trying to fix the issue by creating a support file and later providing it to all users.

In December, 2016, National Assembly of Ecuador's website experienced an intensive SQL injection attack that led to stealing of 930 users' records. "Kapustkiy" is the one who performed the attack. The attacker is skillful and implemented many other attacks over several websites like Hungarian Human Rights Foundation, Eastern Indian Regional Council Server, and Italy Dipartimento della Funzione Pubblica. The outcomes of the attacks were, in some cases, huge like stealing users and worker's information, which should not be overlooked.

Some other websites were also attacked by other hackers like Russian Embassy in Armenia Database and US government servers in the .us top-level domain. Valuable information were stolen of such high secure servers.

Facebook and Instagram announced rewards to those who find bugs through their systems as a challenge in order to show how secure their systems were, which lately was proven to be wrong by Orange Tsai and Jani in 2016, who imposed SQL injection attacks on these applications.

The table below describes some of the recent web applications that were SQL injected.

Table 4.1 - Recent SQL Injected Web Applications

Website / Application	**Year**	**Attacker**	**Consequence**	**Purpose of the Attack**
Tesla	03/2014	Unknown	Research accessed to customer records and superuser areas of site.	Research Purposes
University of Sydney	02/2015	Unknown	Personal information of 5,000 students were looked over just because the attacker wanted to mess.	Illicitly view data
TalkTalk	10/2015	Unknown	Four million customers' information including: personal details, passwords, credit card, were stolen.	Stealing data from the TalkTalk servers

Table 4.1 - Recent SQL Injected Web Applications

Website / Application	**Year**	**Attacker**	**Consequence**	**Purpose of the Attack**
National Assembly of Ecuador website	12/2016	Kapustkiy	930 users' records were stolen	Stealing data
McAfee enterprise software console	12/2016	Unknown	Those systems which used McAfee security software back then were vulnerable	Taking control of the server via root control
Facebook	04/2016	Security researcher (Orange Tsai)	Employee password vulnerability was discovered by that researcher.	Looking for vulnerabilities in Facebook's software

Table 4.1 - Recent SQL Injected Web Applications

Website / Application	**Year**	**Attacker**	**Consequence**	**Purpose of the Attack**
Instagram	05/2016	Jani	The attacker showed that comments could be deleted.	Looking for vulnerabilities in Instagram's software
Florida Elections sites	05/2016	TALLAHASSE The young cybersleuth	Usernames and passwords were stolen.	Stealing data from the servers
Muslim Match - dating website	07/2016	Unknown	Certifications and profiles for 150,000 subscribers were leaked	Exposing user's data to the everyone
Bitcoin	08/2016	Hacker0	Hacker proved that one can steal bitcoin using sql injection.	Stealing data

Table 4.1 - Recent SQL Injected Web Applications

Website / Application	**Year**	**Attacker**	**Consequence**	**Purpose of the Attack**
Epic Games	08/2016	Unknown	80,000 users were warned to change their security information.	Stealing data from Epic servers
Dota 2 forum	08/2016	Unknown	19+ million players' record were exposed	Stealing data from the Dota forums
Hungarian Human Rights Foundation	11/2016	Kapustkiy and CyberZeist	20,000 accounts' personal information, Including some phone numbers and home addresses, were exposed.	Stealing data

Table 4.1 - Recent SQL Injected Web Applications

Website / Application	**Year**	**Attacker**	**Consequence**	**Purpose of the Attack**
US government servers in the .us top-level domain.	09/2016	Fear	Usernames alongside with passwords for every FTP server on a US domain were looked over.	Leak data from the servers
Eastern Indian Regional Council Server	11/2016	Kapustkiy	More than 17,000 students' data were exposed.	Showing system admins bugs in the applications
Italy Dipartimento della Funzione Pubblica	11/2016	Kapustkiy	More than 45,000 users' data were vulnerable including logins.	Leak data from the server

Table 4.1 - Recent SQL Injected Web Applications

Website / Application	**Year**	**Attacker**	**Consequence**	**Purpose of the Attack**
Russian Embassy in Armenia Database	12/2016	Cryptolulz	Admin credentials were examined.	Informing the admins of the bug and leak data
McAfee "ePO"	2017	Unknown	It costed the company to fix the issue by creating a support file and later on providing it to all user.	Leak and steal information
BSNL's website	03/2018	Elliot Alderson	An estimation of 47,000 employees' data were stolen.	Reporting the bug to the admins

5 PREVENTION TECHNIQUES

From the previous section, it is very evident that SQLI attacks are an important class of attacks made on web applications. Table 4.1 highlights the most recent ones and the complete list is huge. Any website that is web-based is vulnerable to SQLI attacks if not secured properly. Although there is significant amount of research done on the different types of SQLI attacks and their mechanisms, there exists insufficiencies in research addressing effective techniques for detecting and preventing them from happening. In this section, the prevention techniques used against the different types of SQLI attacks are described.

Web applications developed using HTML has always been vulnerable to SQL injection attacks. By entering a HTML command as input, attackers were able to manipulate the program to get what they wanted. SQL-DOM is one of those prevention methods that was developed to handle the injected HTML's commands. SQL-DOM is capable of turning HTML into structured data thus making it hard for the hackers to enter HTML commands as input.

Another prevention way is SQLrand. A very smart method that transforms the application Instruction-Set Randomization to a SQL language, and the result of the transformation will be appended by random number, with which the hacker who tries to perform any SQL injection will be unable of guessing the appended number. And by that, according to Boyd and Keromytis, "any malicious user attempting an SQL injection attack would be thwarted, for the user input inserted into the 'randomized' query would always be classified as a set of non-keywords, resulting in an invalid expression" [4].

Many other prevention ways like AMNESIA, SQLCheck, SQLGuard and CANDID have proven to be successful in terms of preventing SQL injection, but still cannot prevent against all of SQL injection types. All of the methods mentioned below are not successful in preventing the Stored Procedure attack. But the technique proposed by Halfond. et. al. [12] uses positive tainting and syntax-aware evaluation technique to prevent all types of SQL injection. According to

Manmadhan and Manesh [18], tainting is the best SQL injection prevention method. It is both precise and efficient because it is purely dynamic.

Table 5.1 - SQLI Prevention Techniques

Technique Name	**How It Works**	**Completely Prevent Against**	**Partially Prevent Against**	**Cannot Prevent Against**
SQL-DOM [20]	In this technique, SQL statements are constructed through manipulation of objects. This is done is two parts: the first is constructing the abstract object model and the second is running an executable that is executed against a database schema.	Tautology/ Illegal/ Piggy Back/ Union/Inference / Alternate encoding / SQLIA / SQLI + DNS Hijacking	None	Stored Procedure
SQLrand [4]	This technique uses the concept of instruction-set randomization to SQL. Whenever the attacker tries to inject the attack, the database parser caches and terminates it.	Tautology / Piggy Back/ Union / Inference	None	Illegal / Stored procedure / Alternate coding

Table 5.1 - SQLI Prevention Techniques

Technique Name	**How It Works**	**Completely Prevent Against**	**Partially Prevent Against**	**Cannot Prevent Against**
AMNESIA [11]	This technique uses static analysis and runtime monitoring of application code to detect and prevent SQLI attacks. The four main steps involved are: identifying hotspots, building SQL-query models, instrument application, and runtime monitoring	Tautology / Illegal / Piggy Back/ Union / Inference / Alternate encoding	None	Stored Procedure
Tainting [12]	This is a highly automated approach that uses positive tainting and the concept of syntax-aware evaluation to counter SQLI attacks. This technique	Tautology / Illegal / Piggy Back / Union/Inference /	None	None

Table 5.1 - SQLI Prevention Techniques

Technique Name	**How It Works**	**Completely Prevent Against**	**Partially Prevent Against**	**Cannot Prevent Against**
	makes use of trusted data sources and allows data only from these trusted data sources.	Alternate encoding / Stored Procedure		
SQLCheck [25]	This algorithm prevents command injection attack by using context free grammars and compiler parsing techniques.	Tautology / Illegal / Piggy Back / Union/Inference / Alternate encoding	None	Stored Procedure
SQLGuard [15]	This technique does a run time comparison of the user input with the parse tree of the SQL statement before and after inclusion of user input. The	Tautology / Illegal / Piggy Back / Union / Inference / Alternate encoding	None	Stored Procedure

Table 5.1 - SQLI Prevention Techniques

Technique Name	**How It Works**	**Completely Prevent Against**	**Partially Prevent Against**	**Cannot Prevent Against**
	technique detects and eliminates the SQLI attacks by juxtaposing the intended query structure with the instantiated query.			
CANDID [3]	This technique detects SQLI attacks by dynamically comparing the candidate inputs with the programmer-intended query structure.	Tautology	Illegal/ Piggy Back / Union /Inference / Alternate encoding	Stored Procedure

6 DETECTION TECHNIQUES

Attackers, who use SQL injection, aim to achieve several purposes. Getting to the database through using SQL injection is one of them. Once the attacker gets to the database, then they will be able to impose threats from several perspectives. The attacker may have access to very sensitive information and, therefore, can perform obliteration and alteration on that information.

However, one problem remains, which is how to know if the application was attacked. Detection is a way to discover the footprints of the attacker and also plays a profound part of preventing SQL injection. Learning whether you were attacked or not help solving the problem and overcome some expanded consequences. Some of the detection methodologies were created especially in order to search for specific vulnerabilities. Same for SQL injection, Fast Flux Monitor, Machine Learning, and Ardilla tools are methods to detect SQL injection attacks, while Noxes tool, SQLMap, and Session Shield are methods used to detect and prevent SQL injection simultaneously [2].

7 FUTURE WORK AND CONCLUSIONS

With the growing demand of web applications comes the challenge of addressing their security problems. SQLI attacks have been around for decade and have produced undesirable consequences. The classifications of recent prevention and detection techniques presented in this paper conclude that SQLI threats have not been fully overcome. Some of the findings that resulted out of this research effort are: understanding the nature of hackers is equally important for detecting and preventing SQLI attacks; looking at some recurring patterns of attacks and observing the footprints of hackers can be helpful; proper detection mechanisms are essential to expose the SQLI attacks; and techniques that does both detection and prevention on all types of SQLI attacks are scarce in the state of art.

For future work we are looking at the following goals. The first goal is to look at ways of improvising the efficiency of existing prevention and detection techniques to prevent all types of SQLI attacks; the second goal is to combine existing techniques and come up with hybrid versions to cover a broad spectrum of SQLI attacks; and the third is to propose a novel automated approach to build highly resilient web applications that are less likely to be attacked by any type of SQLI attacks.

REFERENCES

[1] Acunetix. "What Is SQL Injection (SQLi) and How to Fix It." Web.

[2] Alwan, Zainab S., and Manal F. Younis. "Detection and Prevention of SQL Injection Attack: A Survey." (2017).

[3] Bandhakavi, Sruthi, et al. "CANDID: preventing sql injection attacks using dynamic candidate evaluations." Proceedings of the 14th ACM conference on Computer and communications security. ACM, 2007.

[4] Boyd, Stephen W., and Angelos D. Keromytis. "SQLrand: Preventing SQL injection attacks." International Conference on Applied Cryptography and Network Security. Springer, Berlin, Heidelberg, 2004.

[5] Buehrer, Gregory, Bruce W. Weide, and Paolo AG Sivilotti. "Using parse tree validation to prevent SQL injection attacks." Proceedings of the 5th international workshop on Software engineering and middleware. ACM, 2005.

[6] "Dangerous Hole Found In Mcafee Epo Antivirus Central Management Suit". Cso.Com.Au, 2018, https://www.cso.com.au/article/613679/dangerous-hole-found-mcafee-epo-antivirus-central-management-suit/. Accessed 27 Mar 2018.

[7] Digital Trends, "The 'Dota 2' Forum Was Hacked In July, And We're Just Now Hearing About It". 2018, https://www.digitaltrends.com/computing/dota2-forum-hacked-two-mollion-sql-injection/. Accessed 28 Mar 2018.

[8] "Sqli Hall-Of-Shame - The Code Curmudgeon". The Code Curmudgeon, 2018, https://codecurmudgeon.com/wp/sql-injection-hall-of-shame/. Accessed 27 Mar 2018.

[9] "SQL-Injection Test Targets / Websites". Ma.Ttias.Be, 2018, https://ma.ttias.be/sql-injection-test-targets-websites/. Accessed 27 Mar 2018.

[10] Halfond, William G., Jeremy Viegas, and Alessandro Orso. "A classification of SQL-injection attacks and countermeasures." Proceedings of the IEEE International Symposium on Secure Software Engineering. Vol. 1. IEEE, 2006.

[11] Halfond, William GJ, and Alessandro Orso. "Preventing SQL injection attacks using AMNESIA." Proceedings of the 28th international conference on Software engineering. ACM, 2006.

[12] Halfond, William GJ, Alessandro Orso, and Panagiotis Manolios. "Using positive tainting and syntax-aware evaluation to counter SQL injection attacks." Proceedings

of the 14th ACM SIGSOFT international symposium on Foundations of software engineering. ACM, 2006.

[13] Jeremy Seth Davis, Senior Reporter et al. "Researcher Finds Backdoor That Accessed Facebook Employee Passwords". SC Media US, 2018, https://www.scmagazine.com/researcher-finds-backdoor-that-accessed-facebook-employee-passwords/article/528524/. Accessed 28 Mar 2018.

[14] Jones, Christopher, and Alison Holloway. "The Underground PHP and Oracle® Manual." Oracle (2007).

[15] Lee, Inyong et al. A Novel Method For SQL Injection Attack Detection Based On Removing SQL Query Attribute Values. 2018. Accessed 28 Mar 2018.

[16] Limited, Bharat. "-CUSTOMER VALUE-". -CUSTOMER VALUE-, 2018, http://www.bsnl.co.in. Accessed 28 Mar 2018.

[17] Mandalia, Ravi. "Researcher Finds SQL Injection Vulnerability On Tesla Website - Techie News". Techie News, 2018, http://www.techienews.co.uk/976931/researcher-finds-sql-injection-vulnerability-tesla-website/. Accessed 28 Mar 2018.

[18] Manmadhan, Sruthy, and T. Manesh. "A method of detecting sql injection attack to secure web applications." International Journal of Distributed and Parallel Systems 3.6 (2012): 1.

[19] Mavromoustakos, Stephanos, et al. "Causes and Prevention of SQL Injection Attacks in Web Applications." Proceedings of the 4th International Conference on Information and Network Security. ACM, 2016.

[20] McClure, Russell A., and Ingolf H. Kruger. "SQL DOM: compile time checking of dynamic SQL statements." Software Engineering, 2005. ICSE 2005. Proceedings. 27th International Conference on. IEEE, 2005.

[21] Osborne, Charlie. "Security Flaw In Mcafee Enterprise Software Gives Attackers Root Access | Zdnet". Zdnet, 2018, http://www.zdnet.com/article/security-flaw-in-mcafee-enterprise-software-gives-attackers-root-access/. Accessed 28 Mar 2018.

[22] Sharma, Chandershekhar, and S. C. Jain. "Analysis and classification of SQL injection vulnerabilities and attacks on web applications." Advances in Engineering and Technology Research (ICAETR), 2014 International Conference on. IEEE, 2014.

[23] Tajpour, Atefeh, Suhaimi Ibrahim, and Mohammad Sharifi. "Web application security by sql injection detectiontools." IJCSI International Journal of Computer Science Issues 9.2 (2012): 332-339.

[24] "SQL Injection Cheat Sheet & Tutorial: Vulnerabilities & How to Prevent SQL Injection Attacks." Veracode. 06 Oct. 2017. Web.

[25] "SQL CHECK Constraint". W3schools.Com, 2018, https://www.w3schools.com/sql/sql_check.asp. Accessed 28 Mar 2018.

[26] Xu, Zhengchuan, Qing Hu, and Chenghong Zhang. "Why computer talents become computer hackers." Communications of the ACM 56.4 (2013): 64-74.