A Survey on Authentication Mechanism against SQL Injection in XML

Preshika Tiwari
PG Scholar, CSE
N.I.I.S.T Bhopal , India

Ashish Kumar Srivastava
Associate Professor, CSE
N.I.I.S.T Bhopal, India

ABSTRACT
SQL Injection Attacks (SQLIAs) are very serious intrusion attacks on database driven web application because such attacks can lacks the confidentiality and integrity (security) of data (information) in databases. In reality, unauthorized person intrudes to the web database and then after accordingly, enter in the data. To prevent this type of attack various approaches are proposed by analysts but they are not sufficient because most of implementing techniques will not stop all types of attacks. This paper presents the different kinds of SQL Injection attacks on the web based XML data and on the various present SQLIAs prevention techniques . This paper shows the existing SQLIAs prevention techniques which will demand the client side data, one by one and then validate Which will make typical the developer's job to write various different validation codes for each data page which is receiving in the server side. This paper reviewed the various security threats and XML database and analyze the available security mechanism to protect against the above attacks. This paper also presents the various research scopes in XML SQLIAs.

Keywords
SQL Injection, XML, Stored Procedure.

1. INTRODUCTION
The tremendous use of the Web as a quick means of data dissemination and various other transactions including those having financial drawbacks (consequences), has essentially made it a key component of today's Internet architecture. These applications and their underlying data store keep necessary secure data. The small mismanage will lead to the millions dollar loss which could greatly affect the users. Hence, it is necessary to protect these applications from targeted attacks now a day's many activities are done by dynamic web application. For example several people pay their bills, book the hotels (restaurants) or and give the online exam through the dynamic websites instead of spending time for commuting. It is very necessary that the private information of the user should follow the CIA trade (i.e. Confidentiality, Integrity and Authentication) information of people must be kept secret and confidentiality and integrity of them must be provided by developer of web application but unfortunately there is no any guarantee for preserving the underlying databases from current attacks. Web applications are vulnerable (insecure) to outside attacks, in which unauthorized person easily threaten the application's underlying database. The Structural Query Language Injection (SQLI) attackers were hired when an attacker used to change the logic or semantics or the syntax Of a SQL query by inserting new malicious Structured Query Language keywords or operators. When there are no input validation phenomena SQL Injection Attacks are hired . In reality, attackers can shape their illegal input as a component of final query that will operate by databases. Web applications or undisclosed information systems can be the victims of this vulnerability because attackers can abuse this vulnerability can threat their authority, integrity and confidentiality. So, programmers should use some different coding patterns to avoid this vulnerability but they are not sufficient enough. SQLIAs are also capable of escaping traditional tools such as firewalls and various Intrusion Detection Systems (IDSs) because they are hired through workstation ports used for regular (permanent) web flow (traffic) usually are open in firewalls . On the other side many IDSs aims on the network layer and IP layers of the security but SQLIAs occupies their place at the application layer. Many analysts have proposed a range of technique to get a rid of from these attacks through defensive coding style [4], [5]. SQL-Injection Attack (SQLIA) constitutes as one of the great attacks against web applications.. Due to the deficiency in input validation, an attacker can be able to directly access the data store. Any web application exposed on the WWW or even in the simple intranet could be vulnerable to SQLIAs. Although the major causes that lead to SQLIAs are well known, but they still exist because of lack of effective mechanism for detecting and preventing them. Secure way of programming could, protect against various types of SQLIAs. This entire process for protecting Several approaches have been proposed in the studies to prevent SQLIAs in the application layer, which will first combine the static study of application level programs and then the runtime justification of dynamically generated SQL-queries with addition of user inputs. Although these approaches will prevent SQLIAs at the application layer, very small importance is given on securing objects kept in the database layer such as stored procedures which are also largely vulnerable to SQL Injection Attacks. Stored procedures occupies an important place in the present -day relational database stores. They are responsible for adding the extra layer of abstraction level into the design of a software system, i.e. This extra layer, at the particular degree, secures some design confidential from the potentially unauthorized personnel's, such as relational table designs. By using the mechanism of stored procedures, one can be ensured that the information is stored in the data store safely. In these databases, the developer use dynamic SQL queries i.e., SQL statements are build at runtime according to the different user inputs. For example, in SQL Server, EXEC (varchar (n) @SQL) could execute arbitrary SQL statements. On the basis of this flexibility feature degree to construct SQL statements according to different requirements can be easily managed ,

but have a a threat from SQL Injection Attacks, the problem is the impractical nature of various present techniques because they could not address all attack types or have not been implemented yet. Also some of them requires to modify web application code or extra working source. However, the main aim of this paper is to introduce all types of SQL injection attacks and to evaluate present approaches which will detect and then after preventing these attacks.

2. LITERATURE REVIEW

2.1 Vulnerabilities

This section shows the vulnerabilities commonly found the web applications defined in[15].

TABLE I: Types of vulnerabilities

Types of Vulnerabilitiy	Description of Vulnerability
1st Type	Delay of operation analysis till the runtime phase where the current variables are considered rather than the source code expressions.
2nd Type	Lack of clear distinction between data types accepted as input in the programming language used in the Web application development.
3rd Type	The validation of the user input is not well defined or sanitized . Inputs are not checked correctly.
4th type	Weak concern of type specification in the design a number can be used as a string or vice-versa.

2.2Various SQL injection attack types:

This section presents the type of SQL Injection Attacks and Its working methodology defined in[2].

Table II: Types of SQL injections

S: NO	Types of Attack	Working Method
1	Tautologies	In such category of attack the intruder injects SQL tokens (malicious keywords) to the conditional query statement which will evaluate always true. This type of attack used to bypass security control and to gain access to information by avoiding vulnerable input field which use the WHERE clause in the SQL.
2	Union Query	The UNION keyword is used in such queries which will join the malicious query with the simple one
3	Piggybacked Queries	The query delimiter mechanism is used in this type of attack, intruder makes the breach in security of database by appending extra other query to the original query
4	Stored Procedure	Many databases have built-in stored procedures. The attacker executes these built-in functions using malicious SQL Injection codes.
5	Inference -Blind Injection -Timing Attacks	The error generated messages which will provide the needful information to unauthorized persons to hire the attack will be hide by developer here generic page will be shown to intrude. The intruder will collect information by asking a series of true/false questions. An intruder can find some information due to delays in response from the database
6	Alternate Encodings	Alternate coding mechanism such as hexadecimal , ASCII and Unicode etc. will be used in injection query by using alternate encoding

.

3. RESEARCH SCOPE

The WSIVM (web service input validation model) consist of XML schema and module for just performing the input validation according to the schema. WSIVM focus on data validation allowing only valid entries to be accepted. Since it is based on a whitelist approach in which only predefine values are accepted and others are considered invalid.[1]

This paper presented the comparison of SQL injection detection and prevention techniques. This will first identify the various types of SQLIA then investigate on SQL injection detection and prevention techniques and after that it will compare the techniques in terms of the ability to stop SQLIA. Regarding the results some current technique's are imposed for stopping SQL injection attacks. [2]

This paper provides the penetrating testing tools for the web service specific attacks known as the WS – attacker. It shows the design decision which enabled to construct a general framework extensible with web service specific attack plugins. [3]

As the deploy of defensive coding or OS hardening but it is not enough to stop SQLIAs to web applications because of that researchers have proposed some of the techniques to assist developers. The paper proposes WAVES, a black box technique for testing web applications for SQL injection deficiencies the tool identifies all points a web application that can be used to insert SQLIAs. [8]

It constructs attacks that aim these points and observe the application how response to the attacks by utilize machine

learning IDBC-Checker was not invented for the purpose of detection and prevention As most of the SQLIAs contains the of syntactical and type correct query format so this technique would not catch more general forms of these attacks. [9]

This paper proposed Tautology Checker that uses static analysis to stop tautology attack. The important limitation of this technique is that its scope is limited to tautology and cannot detect or prevent other types of attacks. [10]

This paper proposed the design of a static analysis framework, called SAFELI for identifying SQLIA vulnerabilities at compile time. SAFELI statically monitor the MSIL (Microsoft Symbolic intermediate language) byte code of ASP DOT NET Web applications using symbolic execution. SAFELI can analyze the source code and will be able to identify delicate vulnerabilities that cannot be discovered by black-box vulnerability scanners. The main drawback of this technique is that this approach can discover the SQL injection attacks only on Microsoft based product. [11]

The paper modifies web applications written in Java through a program transformation. This tool dynamically mines the programmer-intended query structure on any input and detects attacks by comparing it against the structure of the actual query issued. It's natural and easy procedure turns out to be very efficient for detection of SQL injection vulnerabilities. [4] [6]

In SQL Guard and SQL Check queries are checked at runtime based on a model which is expressed as a grammar which inputs legal requests only. SQL Guard determines the pattern of the query prior and after the addition of user- input based on the model. During SQL Check, the model is specific independently by the developer. These mechanisms use the secure secret key to bind the user input during parsing check phenomena done by runtime checker, so the security of the approach depends on the inability of the intruder to detect the key. With these mechanisms developer should modify the code to use a special intermediate library or physically inserts special keywords into the code where the value which is input by the user is composed with a dynamically generated query. [12] [13]

This paper combines static analysis and runtime monitoring. Queries are intercepted before they are sent to the database and are checked against the statically built models, in dynamic phase. Queries that violate the model are prevented from accessing to the database. [14], [15].

In this proposal SQL injection prevention techniques verifies authenticity of each data element individually .Consider an example, if any user Inputs to a login form than the various servers will verify the given username and given password separately. In present scenario we can see lot of complex HTML form that user has to fill up and submit. For example, a simple medical store needs to enter various product details into one form. Hence, we can easily understand the complexity of data validation into the server side. In a simple web application each individual page has different types of HTML form which are essential because each form is dedicated for different work, such as user registration, product creation, product purchase etc. Now existing SQL injection prevention techniques handle each application separately.. The developers have to write code for the validity checking of each html-form separately. Since each form has different types of complexity it is currently not possible to make the validating process as generic for all the forms. This also makes the maintenance job difficult as the data validation policy is different in each from submitting server code Different web applications manage the validation rules differently., Also some framework available which allows automated server-side validation and the developer can do the very less work to manage it. But complex validation requirements cannot be satisfied by all those automatic validating systems. Also for most of these validating systems, developers have to enter validation rules separately. Sometimes we need to send multiple information to the server. For example, customer can purchase 10 items online at a time. In that case 10 insertions required in the corresponding database table. Now instead of sending each product information separately we can make one XML file before sending it to the server. On receiving the XML file server will parse the XML file and extract data from it. This is the best way to send large and complex information to the server. Also the middle- man attacker cannot easily guess the structure of data. This XML based data passing is not a new concept and is already supported by almost all server side scripting languages. Also most of the databases have a very powerful tool for parsing XML files. So we can use the databases stored procedures to receive XML file and parse it to get all information's. The stored procedure then processes the extracted data one by one as required.

4. FUTURE WORK

We would like to develop a countermeasure that can efficiently tackle the innovative SQL Injection attacks and fix as much vulnerability as possible. Hackers are in reality very innovative and as the time is passing by, new attacks have been launched that may need new ways of thinking about the solutions we currently have on our hands.

5. CONCLUSION

Database driven web application is threatened by SQL Injection Attacks (SQLIAs) because this type of attack can compromise confidentiality and integrity of information in databases. In this paper we have done a survey of the different kinds of SQL Injection attacks and also the existing SQLIAs prevention techniques available. Though many approaches and frameworks have been identified and implemented in many interactive Web applications, security still remains a major issue. SQL Injection prevails as one of the top-10 vulnerabilities and threat to online businesses targeting the backend databases. This paper, has reviewed the most popular existing SQL Injections related issues.

6. REFERENCES

[1] Rafael Bosse brinhhosa, Carla Merkle Westphall and Carlos Becker Westphall,"Proposal and development of the web services input validation model" 978-1-4673-0269-2/12/$31.00@2012 IEEE.

[2] Atefeh Tajpour, Maslin Massrum and Mohammad zaman Heydari."Comparison of SQL Injection detection and prevention techniques," in proceeding of 2nd international conference on education technology and computer(ICETC)

[3] Christian mainka,juraj somorovsky and jorg schwenk,"Penetrating testing tool for web services security,"2012 IEEE eighth world conference on services

[4] P. Bisht, P. Madhusudan, and V. N. Venkratakrishnan, "CANDID: dy- namic candidate evaluations for automatic prevention of SQL injection attacks," ACM Trans. Inf. Syst. Secure., vol. 13, no. 2, pp. 1–39, 2010.

[5] K. Kemalis and T. Tzouramanis, "SQL-ids: a specification-based approach for SQL-injection detection," in Proceedings of the 2008 ACM symposium on Applied computing, sir. SAC '08. ACM, 2008, pp. 2153–2158.

[6] S. Bandhakavi, P. Bisht, P. Madhusudan, and V. N. Venkatakrishnan, "Candid: preventing SQL injection attacks using dynamic candidate valuations," in Proceedings of the 14th ACM Conference on Compute and communications security, sir. CCS '07, 2007, pp. 12–24.

[7] X. Jin and S. L. Osborn, "Architecture for data collection in database intrusion detection systems," in Proceedings of the 4th VLDB conference on Secure data management, sir. SDM'07, 2007, pp. 96–107.

[8] Y. -W. Huang, S. -K. Huang, T. -P. Lin, and C. -H. Tsai, "Web application security assessment by fault injection and behavior monitoring," in Proceedings of the 12th international conference on World Wide Web,ser. WWW '03, 2003, pp. 148–159.

[9] G. Wassermann, C. Gould, Z. Su, and P. Devanbu, "Static checking of dynamically generated queries in database applications," ACM Trans. Softw. Eng. Methodol., vol. 16, no. 4, Sep. 2007.

[10] G. Wassermann and Z. Su, "An analysis framework for security in web applications," in In Proceedings of the FSE Workshop on Specification and Verification of Component-Based Systems (SAVCBS 2004, 2004, pp. 70–78).

[11] X. Fu and K. Qian, "Safeli: Sql injection scanner using symbolic execution," in Proceedings of the 2008 workshop on Testing, analysis, and verification of web services and applications, ser. TAV-WEB '08, 2008, pp. 34–39.

[12] G. Buehrer, B. W. Weide, and P. A. G. Sivilotti, "Using parse tree validation to prevent sql injection attacks," in Proceedings of the 5th international workshop on Software engineering and middleware, ser. SEM '05, 2005, pp. 106–113.

[13] Z. Su and G. Wassermann, "The essence of command injection attacks in web applications," SIGPLAN Not., vol. 41, no. 1, pp. 372–382, Jan. 2006.

[14] W. G. J. Halfond and A. Orso, "Amnesia: analysis and monitoring for neutralizing sql-injection attacks," in Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering, ser. ASE '05, 2005, pp. 174–183.

[15] Diallo Abdoulaye Kindy and Al-Sakib Khan Pathan,"A survey on SQL injection:vulnerabilities,attacks and prevention techniques," in 2011 IEEE 15th international symposium on consumer electronics.