© 2016 IJSRSET | Volume 2 | Issue 3 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

An Approach Based on SVM Classifier to Detect SQL Injection Attack

Ritu Awasthi, Dharmendra Mangal

Information Technology, Medi-Caps Institute of Technology and Management, Indore, Madhya Pradesh, India

ABSTRACT

The query process allows the attacker to achieve uncertified access to the back-end server and database, and remove or change sensitive information. It could be threatened because of interaction of code and data. SQL-Injection, cross-site scripting (XSS), cross-site request forgery (XSRF) are some examples of vulnerabilities. Injection Attacks exploit vulnerabilities of Web pages by inserting and executing malicious. We are proposing SVM (Support Vector Machine) for grouping and prediction of SQL Injection attacks. SQL Injection attack identification or detection accuracy is considerably better among the existing SQL-Injection detection techniques. The proposed framework reduces radically the runtime monitoring overhead. It is focusing only on SQL query conditions and program fragments that are vulnerable to injection attacks.

Keywords : SQL Injection, Support Vector Machine, Malicious Code, Information Security.

I. INTRODUCTION

The query process allows the attacker to achieve uncertified access to the back-end server and database, and remove or change sensitive information. Mobile phone apps, browser extensions are widely popular third-party written weakly-vetted pieces of code [9, 11]. They expand the functionality of computer browsers by interacting with browser level events and data. Cross-context scripting attacks are caused when crafted inputs from the content pages (web pages, RSS feeds). It is presented in the privileged interpretation of the web browser [3]. Another reason of XSS attack is injecting malicious code into the extensions, which run with the same privileges as the web browser. SQL injection and XSS vulnerabilities, both provided a legitimate flow of information from the non-trusted sources to the trusted application [3, 8]. The inputs from the un-trusted sources force the trusted applications to behave that are not intended by the application developers [17]. A user inserts inputs that produce different SQL queries with structures that the programmer intended. Injection Attacks exploit vulnerabilities of Web pages by inserting and executing malicious code (for example, database queries, java script functions) in unsuspecting users' computing environment or on a Web server [4, 8]. These types of attacks compromise user's information and system resources, and pose a serious threat to personal as well as to business assets. There are required to explore data mining technique, especially association rule mining [14] as one of the probable way to predict the behaviour of attacker.

II. METHODS AND MATERIAL

A. Literature Survey

Goseva et al. [5] proposed their system based on three machine learning method to classify the attacker activities. It uses three tier honey pots system for collecting datasets. It uses multiple learners and compares their performance on four datasets. A feature selection method is used to effectively separation between attack session and vulnerability session. E. Athanasopoulos et al. [2] have applied ISR to separate legitimate client-side code from potential attacks using a framework that applies to the browser environment Isolation Operators (IO) and Action Based Policies. M.Stephen et al. [6] have developed an Enhanced XSS Guard Algorithm, E- Guard. This passive detection system positioned between the Web server and browser applies to XSS. The goal of the algorithm is to list a

Website on a blacklist, white list or grey list for sites. M.Ruse et al. [7] have presented a SQLIV query level tool that develops a model of the query capturing the dependencies of sub-queries. The model is analyzed with a concolic testing tool to automatically generate inputs. S. Artzi et al. [10] proposed that "Apollo" is a technique which combines concrete and symbolic execution (concolic testing) and explicit-state model checking. It is used to detect vulnerabilities created by runtime errors and by malformed HTML, which could include SQLI and XSS.

B. SQL Injection Attacks

SQL injection attacks occur when the inputs to Web applications are used to attack the back-end database layers of the Web servers. Various bypassing techniques and attacks [16] have been developed to gain unauthorized access like SQL Injection attack. SQL field manipulation fields are where fielding, Insert Filed, Error message generation field, which retrieve useful information and destroys web applications? Flexibility is strength of SQL queries. Similar to a true programming language, Structured Query Language (SQL) allows the addition of inline comments within the code. These queries allow pattern and regular expression (RE) matching of strings [12]. SQL also enables users to concatenate and join different values or characters (such as from a field or column of a data record) to form a complete SQL string. It will be shown that, it is the stretch ability of SQL syntax which primarily makes signature based identification challenging. SQL query also has a "UNION" construct which allows the rows from different tables to be selected simultaneously, as long as the data table columns are of compatible types [15]. This is a powerful feature that opens up a new dimension in data table selection and enables more complicated formation or aggregation of SQL queries.

E.g.: consider a query

Select * from *tablename* where employeename = 'x' and pwd = 'y',
The attacker enters ' OR 5=5 -- in employeename field and leaves the pwd field blank. The query would become:

Select * from *tablename* where employeename =' ' OR 5 =5 -- and pwd = ''

The single quote entered by the user disables the opening quote and OR 5=5, which is a tautology, satisfies "where" clause for all records [13]. Hence, when this query is executed it will return information about the entire employee in the table.

C. Proposed Work: SQL Injection Detection Through SVM Classifier

The system discussed is called An Approach based on SVM classifier to detect SQL injection attack.

Support Vector Machines (SVM) is a supervised learning method. It is used for classification of query string. Classification of Suspicious query is done by analyzing the datasets of Original query and suspicious query. Classifier learns the dataset and according to learning procedure, it classifies the queries. Input feature vectors are mapped into a higher dimensional space by SVM using kernel function [5]. A maximal separating hyperplane is built in the transformed space by considering a two class problem. Creation of a good class separating hyperplanes the selection of the kernel function plays an important role. The Radial Basis Function (RBF) as a kernel function is used based on [5]. The flow chart of the proposed framework is depicted in Figure-1.

New technique for *Query Maping* has been implemented in the proposed work. Here, one to one mapping function MAP (N) has been designed to classify SQL queries,

f(O)=Class of Original and authentic queries.
f(S)=Class of Malicious and Mal-functioned queries.

The propose concept is based on the tokenization of queries. Here different tokens are created of both the classes, and compared by their F(N) function. SQL Injection is not present If mapping function is equal for both the queries, but if no matches are found then it would mean that the attack has occurred on system.

*Original Query: select * from college where department_id= "cs123"*

*Malicious Query: select * from college where department_id= " "OR 1=1;*

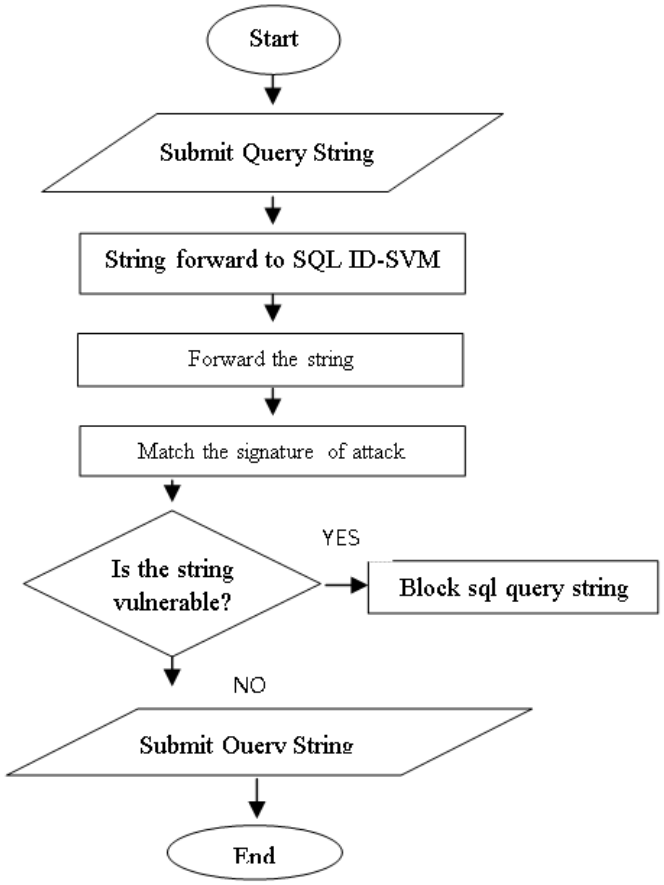

Figure 1: Flow chart of the proposed framework

SQL injection is detected by comparing original and suspicious model to detect SQL injection as depicted in Figure-2. Suspicious query detection algorithm is used for comparison. The result of comparison is a character value if SQL query string is malicious. False value indicates that SQL statement is malformed hence could be possible SQL injection attack and prevents SQL statement from being executed.

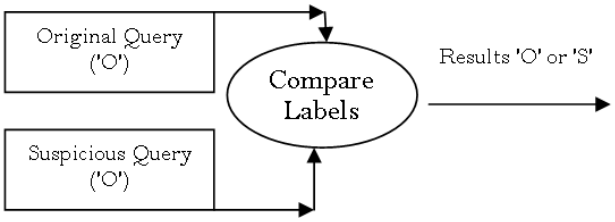

Figure 2: Comparing Original and Suspicious model to detect SQL injection attack

Algorithm for Suspicions Query Detection- Step 1. Input dataset of SQL Queries,

Step 2. Select Reasonable amount of training data.

Step 3. Do classification based on SVM classifier.

Step 4. Compare the U(N), with mapping Function MAP (T).

Step 5. If f(O)=U(N),Then there is no SQL-Injection and SQL-Poisoning attack.

SHOW MESSAGE" No Attack Found"

Step 6. If f(S)=U(N) ,then there is SQL-injection Attack if found.

SHOW MESSAGE "Attack Found"

Step 7. Repeat the steps 2 to steps 4 for different dataset size.

Step 8. Calculate Accuracy based on parameters TPR, TNR, FPR, FNR, and Training Time and Detection Time.

Where,

U(N)=User Input SQL-query.

f(O)= Authentic Query(Original and Safe Query Class).

f(S)=Suspicious Query(Malicious Query Class).

MAP(T)=Mapping function.

III. RESULTS AND DISCUSSION

The presented graph shows the training performance of the proposed work.

Where-

Training Time- Time required training the system from SQL-Query dataset.

Detection Time-Time required to detect the attack.

Accuracy-measurement of accurate attack detection

Accuracy=TP+TN / TP+FN+FP+TN

TP=True Positive, TN=True Negative

FN=False Negative, FP=False Positive

Figure-3 shows the Performance Analysis based on training time and detection time. Training time is the time to train the system and the detection time is the time that is taken by the system to detect the attack i.e. given SQL query string is OQ or SQ.

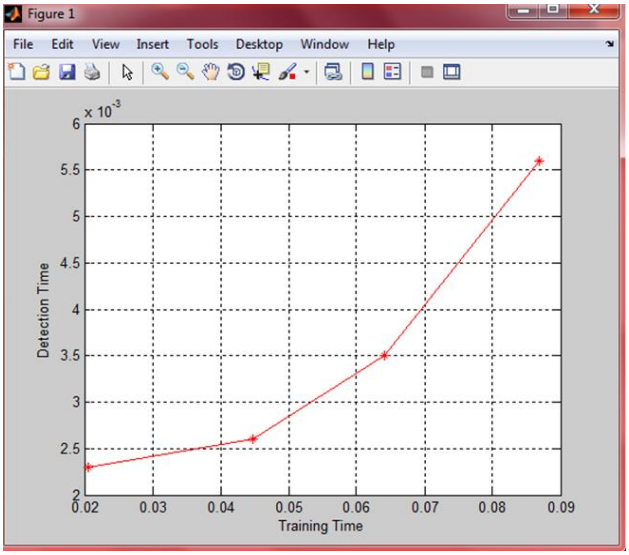

Figure 3: Performance analysis based on Detection Time and Training Time

Figure-4 shows the Performance Analysis based on detection time and accuracy. Detection time is the time that is taken by the system to detect the attack i.e. given SQL query string is OQ or SQ. Accuracy is the measurement of accurate attack detection i.e. how accurately system detect the SQL injection attack. Accuracy is depends upon TN, TP, FP, FN.

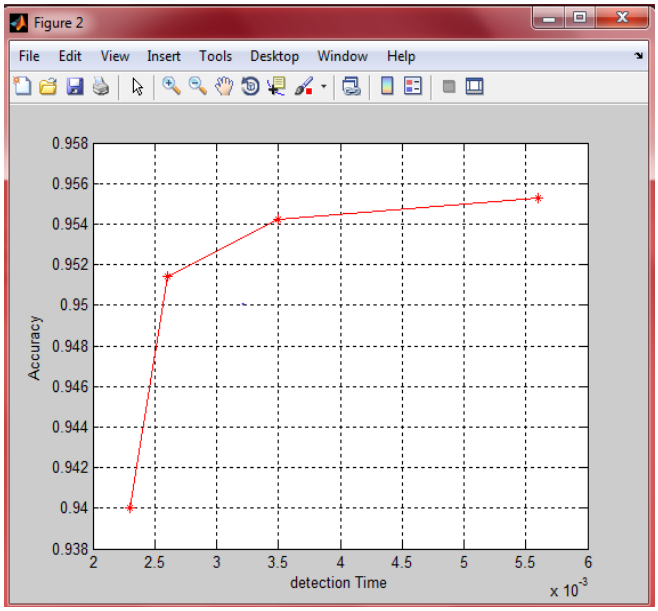

Figure 4: Performance analysis of Graph of Accuracy and Detection Time

IV. CONCLUSION

We presented detection mechanism to protect web application system from SQL Injection attack. SQL injection attacks are frequently carried out by hacker to retrieve unauthorized information. These attacks typically alter the syntactic structure of query.

We can conclude that the strategy of attacks that are targeted on web systems depends upon searching process. Based on this fact we developed a security system which uses SVM to analyze SQL statement. It prevents SQL injection attacks. Proposed work involves tracking strings from their source to the web application's output. We proposed using strings at the level of the program's data to represent delimiters. However, new strings at the level of the program's data may change the program's behavior. A central contribution of this dissertation is that web applications can be analyzed effectively as meta-programs. Gradually to design efficient decision procedures important efforts are made for these theories. We tested our security system for different cases and it worked correctly to prevent SQL injection attack. SQL-Injection attack detection accuracy is good compared to that of others. It is the highest among the existing SQL-Injection detection techniques.

V. REFERENCES

[1] A. Tajpour, M. Massrum and M. Z. Heydari, "Comparison of SQL Injection Detection and Prevention Techniques", 2nd IEEE International Conference on Education Technology and Computer (ICETC), 2010.

[2] E. Athanasopoulos, V. Pappas, A. Krithinakis, S. Ligouras, E. P. Markatos, and T. Karagiannis, "xjs: practical xss prevention for web application development", In Proceedings of the 2010 USENIX conference on Web application development, Berkeley, CA, USA: USENIX Association, 2010, pp. 13-19.

[3] G. J. William and A. Orso, "AMNESIA: Analysis and Monitoring for Neutralizing SQL-Injection Attacks", In International Conference on Automated Software Engineering (ASE), 2005.

[4] Kiezun VA., P. J. Guo, K. Jayaraman, and M. D. Ernst, "Automatic creation of SQL injection and

cross-site scripting attacks", 31st IEEE International Conference on Software Engineering (ICSE), 2009, pp. 199-209.

[5] K. G. Popstojanova, G. Anastasovski, and R. Pantev, "Classification of malicious Web sessions", 21st IEEE International Conference on Computer Communications and Networks (ICCCN), 2012.

[6] M. Stephen, P. P. Reddy, C. D. Naidu, and C. Rajesh, "Prevention of cross site scripting with E-Guard algorithm", International Journal of Computer Applications, Vol. 22, No. 5, pp. 30-34, 2011.

[7] M. Ruse, T. Sarkar, and S. Basu, "Analysis & detection of sql injection vulnerabilities via automatic test case generation of programs", In Proceedings of the 10th IEEE/IPSJ International Symposium on Applications and the Internet, Washington, DC, USA: IEEE Computer Society, 2010, pp. 31-37.

[8] I. Balasundaram, & E. Ramaraj, "An Efficient Technique for Detection and Prevention of SQL Injection Attack using ASCII Based String Matching", In Proceedings of the IEEE International Conference on Communication Technology and System Design, 2012, pp. 183-190.

[9] Peter Scherer, Martin Vicher, Jan Martinovic, "Using SVM and clustering algorithms in IDS systems", In Proceeding of Dateso, 2011, pp. 109-119.

[10] S. Artzi, A. Kiezun, J. Dolby, F. Tip, D. Dig, A. Paradkar, and M. D. Ernst, "Finding bugs in web applications using dynamic test generation and explicit-state model checking", IEEE Transaction Software. Engineering, Vol. 36, No. 4, 2010, pp. 474-494.

[11] Tiwari V., Lenka S.K. & Gupta S., "Performance Evolution of Java Remote Method Invocation and Mobile Agent Techniques in Context of Distributed Environment", IEEE International Conference on Networking and Information Technology (ICNIT), Manila, Philippines, 2010.

[12] V. Shanmughaneethi, Y. Pravin, Ra., E. Shyni & S. Swamynathan, "SQLIVD - AOP: Preventing SQL Injection Vulnerabilities Using Aspect Oriented Programming through Web Services", Communications in Computer and Information Science, Vol.169, 2011, pp. 327-337.

[13] X. Fu , K. Qian, "SAFELI-SQL Injection Scanner Using Symbolic Execution", In Proceedings of the ACM 2008 workshop on Testing, analysis, and verification of web services and applications, 2008, pp. 34-39.

[14] Tiwari V., V. Tiwari, S. Gupta, R. Tiwari., "Association Rule Mining: A Graph based approach for mining Frequent Itemsets", IEEE International Conference on Networking and Information Technology (ICNIT 2010) Manila, Philippines, IEEE.

[15] Y. Minamide, "Static Approximation of Dynamically Generated Web Pages", In International Conference on World Wide Web (WWW), 2005.

[16] Tiwari V., Gupta S., & Mishra R., "Computational Study of .NET Remoting and Mobile Agent in Distributed Environment", International Journal of Computing, Vol. 2, Issue 6,PP. 34-39, DBLP, 2010.

[17] Nema A., Basant T., & Vivek T., "Improving Accuracy for Intrusion Detection through Layered Approach Using Support Vector Machine with Feature Reduction", In Symposium ACM Women in Research, Indore, 2016.