*International Journal of Computer Applications (0975 – 8887)*
*Volume 42– No.13, March 2012*

# SQL injection attack Detection using SVM

Romil Rawat
Samrat Ashok Technological Institute
Vidisha, (M.P.), India

Shailendra Kumar Shrivastav
Samrat Ashok Technological Institute
Vidisha, (M.P.), India

## ABSTRACT

Web application has various input functions which are susceptible to SQL-Injection attack. SQL-Injection occurs by injecting suspicious code or data fragments in a web application.Personal information disclosure ,loss of authenticity, data theft and site fishing falls under this attack category. It is impossible to check original data code and suspicious data code using available algorithms and approaches because of inefficient and proper training techniques of dataset or design aspects. In this paper we will use SVM (Support Vector Machine) for classification and prediction of SQL-Injection attack. In our propose algorithm, SQL-Injection attack detection accuracy is (96.47% and which is the highest among the existing SQL-Injection detection Techniques.

**Keywords**
SQL Injection, Database Security, Authentication, SVM.

## 1. INTRODUCTION

Various databases are available for web application functionalities such as Oracle, MySQL, MS-Access, SQL-Serve. All databases have their own structure and functions for storing data applications. Excecution of database application occurs by SQL(Structured Query Language).

Ex:- Select * from ADMIN where uid='1'; (**Original Query**)

User input is supplied through web application interface, which then further executed through available modules or codes of databases. If proper input validation ,syntax validation ,secure coding framework, secure guideline for web designing is not followed, malicious code could be injected in database.

Ex:- Select * from ADMIN where uid=' ' OR 1=1;' (**Suspicious Query**)

Various system and approaches has been already defined for protecting the system from SQL-Injection attack such as IDS (Intrusion detection System Model)[18] using DOM-tree comparison of SQL queries, tools for detecting SQL-Injection attack such as Swaddler[19].

### 1.1 Working principal of web application is as follows

-Web application is requested through a web browser by a user.

-The HTTP protocol accepts a request of user and sent to the targeted web server.

-Server executes the request received

-Application program generates a output and sent back to the user via HTTP.

-Current states of User, Web server and their execution report are maintained by a special unit called cookies.

### 1.2 SVM (Support Vector Machine)

The term SVM[13] is typically used to describe classification with support vector methods and support vector regression is used to describe regression with support vector methods. SVM (Support Vector Machine) is a useful technique for data classification.

The classification problem can be restricted to consideration of the two-class problem without loss of generality. In this problem the goal is to separate the two classes by a function which is induced from available examples. The goal is to produce a classifier that will work well on unseen examples, i.e. it generalizes well. Consider the example in figure 1. Here there are many possible linear classifiers that can separate the data, but there is only one that maximizes the margin (maximizes the distance between it and the nearest data point of each class). This linear classifier is termed the optimal separating hyper plane. Intuitively, we would expect this boundary to generalize well as opposed to the other possible boundaries.

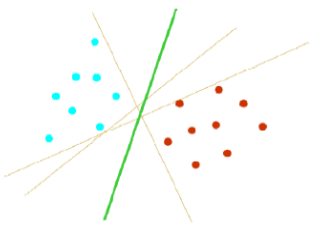

**Fig1. Optimal Separating Hyper Plane**

A classification task usually involves with training and testing data which consist of some data instances. Each instance in the training set contains one "target value" (class labels) and several "attributes" (features). The goal of SVM is to produce a model which predicts target value of data instances in the testing set which are given only the attributes.

To attain this goal there are four different kernel functions.

1. Linear: $K(x_i, x_j) = x_i^T x_j$
2. Polynomial: The polynomial kernel of degree d is of the form.

$$K(x_i, x_j) = (x_i x_j)$$

3. RBF: The Gaussian kernel, known also as the radial basis function, is of the form

$$K(X_i, X_j) = exp\ (-\frac{\|(x_i, x_j)\|}{2\sigma^2})$$

4. Sigmoid: The sigmoid kernel is of the form

$$K(x_i, x_j) = tanh(k(x_i x_j) + r)$$

The RBF kernel nonlinearly maps samples into a higher dimensional space, so it, unlike the linear kernel, can handle the case when the relation between class labels and attributes is nonlinear. Furthermore, the linear kernel is a special case of RBF show that the linear kernel with a penalty parameter C has the same performance as the RBF kernel with some parameters (*C, r*). In addition, the sigmoid kernel behaves like RBF for certain parameters.

In our research we have used a unique concept of determing the SQL-injection attack using SVM(support Vector Machine).classification of Suspecious query is done by analyzing the datasets of Original query and suspicious query. classifies learns the dataset and according to learning procedure ,it classifies the queries. Appropriate classification occurs in our system because of best learning approaches and by designing concerns.

## 2. RELATED WORK

The mechanism to keep track of the positive taints and negative taints is proposed by William G.J. Halfond, Alessandro Orso, Panagiotis Manolios [10], Defensive Programming [11][12] has given a approach for Programmers by which they can implement their own input filters or use existing safe API s that prevent malicious input or that convert malicious input in to safer input. Vulnerability pattern approach is used by Livshits and Lam [8], they propose static analysis approach for finding the SQL injection attack. . The main issues of this method, is that it cannot detect the SQL injection attacks patterns that are not known beforehand. Vulnerability patterns are described here in this approach.

AMNESIA mechanism to prevent SQL injection at run time is proposed by William G.J. Halfond and Alessandro Orso [9].It uses a model based approach to detect illegal queries before it sends for execution to database.

Static analysis framework (called SAFELI) has been proposed by Xiang Fu et al [5], for identifying SIA (SQL Injection attacks) vulnerabilities at compile time.. the source code information can be analyzed by SAFELI and will be able to identify very delicate vulnerabilities that cannot be discovered by black-box vulnerability scanners.

Scott and Sharp Proxy filter [1] [2] , this technique can be effective against SQLIA; they used a proxy to filter input data and output data streams for a web application ,although correctly specify filtering rules for each application is required by the developers to input.

The mechanism which filters the SQL Injection in a static manner is proposed by Buehrer et al [7]. The SQL statements by comparing the parse tree of a SQL statement before and after input and only allowing to SQL statements to execute if the parse trees match.

Novel-specification based methodology proposed by Konstantinos et al [6], they given a mechanism to detect SQL injection with. This approach utilizes specifications that define the intended syntactic structure of SQL queries that are produced and executed by the web-application.

Instruction–Set Randomization [1][3] defined a framework that allows developers to create SQL queries using randomized keywords instead of the normal SQL keywords.

Marco Cova et al [11], proposed a Swaddler which, analyzes the internal state of a web application and learns the relationships between the application's critical execution points and the application's internal state.

NTAGW ABIRA Lambert and KANG Song Lin[12] proposes a string tokenizer which, creates tokens of original query and sql-injected query, and creates array of tokens of both the original and injected query ,if length of arrays of both query is found equal ,that means no sql-injection., Otherwise there is injection.

## 3. PROPOSE TECHNIQUE

Our propose work contains the unique idea that compares SQL query strings and blocks suspicious sql-query and passes original sql-query.

Ex-

Original query=select * from admin where uid='1';

Suspecious query=select * from admin where uid=' ' OR 1=1;--'

Here, original query is passed and suspicious query is blocked.

Word-list contains the tokens of sql-query strings.

'O'-Original query

'S'-Suspecious query

Ex- ('O') select * from admin where uid = '1';

('S') select * from admin where uid = ' ' OR 1=1;--'

('O') select * from admin where uid ='1' && pwd ='abc';

('S') select * from admin where uid = ' ' OR 1=1;--'

Tokens:-t1='select',t2='*from',t3='admin',t4='where' ,t5='uid',t6='',t7='OR',t9='1',t10='&&',t11='pwd=',t12='abc',t13='='

Word-list contains various tokens of named t1…t13,which are listed above.

Vector of string is created, and classifier classifies the original and suspicious query.

### Algorithm

**Step 1**. Select a reasonable amount as the training set.

**Step 2**. Input the SQL-Query string.

**Step 3**.Feed the training set into the SVM-Train process to generate a model.

**Step 4**. Now we are ready to make prediction.

**Step 5**.Now classify the model using SVM classifier.

**Step 6**. Labeled output will give us the accuracy of our algorithm

**Step 7**.Repeat step 2 to 6 till the correct classification precision is achieved.

## .4. TESTING AND RESULTS

The proposed technique has been tested on a SQL-query string dataset. For testing dummy dataset has been created. The dataset has been populated with the records of Original SQL-query string('O') and Suspicious SQL-query string('S') and was tested ,whose results are shown in fig 2

Detection time(in seconds) is calculated by taking average of 100 queries of Original SQL-query string and 100 queries of suspicious SQL-injection qery string.

| | |
|---|---|
| Original Query | 0.01531 |
| Suspecious Query | 0.01519 |

**Fig 2- Detection Time**

Dataset of different size has been taken and accuracy is measured result is shown below in fig 3

**Comparison of accuracy and Datasize**

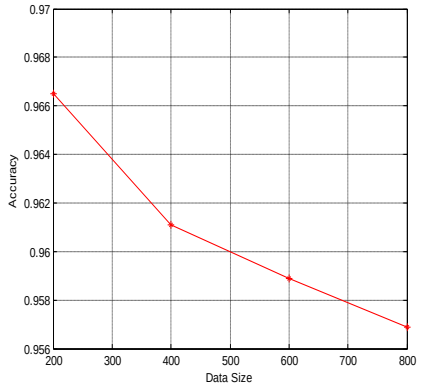


**Fig 3**

Different size of dataset is trained and their detection time is calculated ,result is shown below in fig 4

**Comparison of Detection time and Training time**

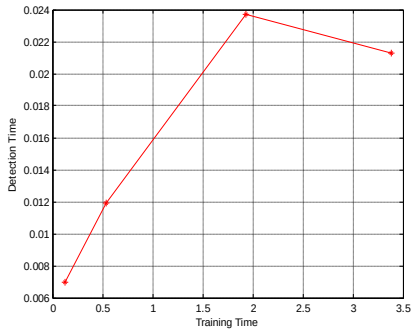


**Fig 4**

Accuracy and detection time is calculated when different sized dataset is used, result is shown below.fig 5

**Comparison of Detection time and accuracy**

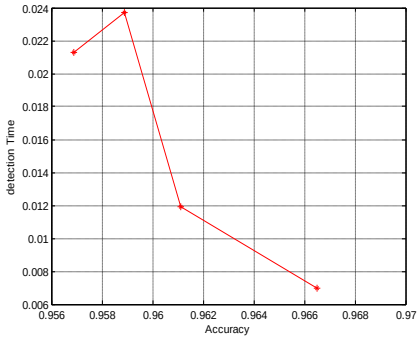


**Fig 5**

TPR,TNR,FPR, and FNR is calculated at different data size ,result is shown below.

**- Comparison of FNR and Datasize**

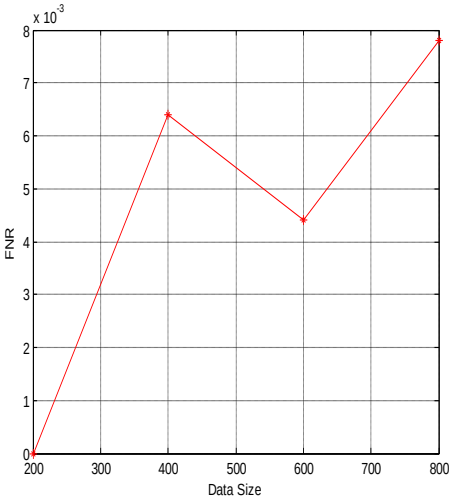


**Fig 6**

**Comparison of FPR and Datasize**

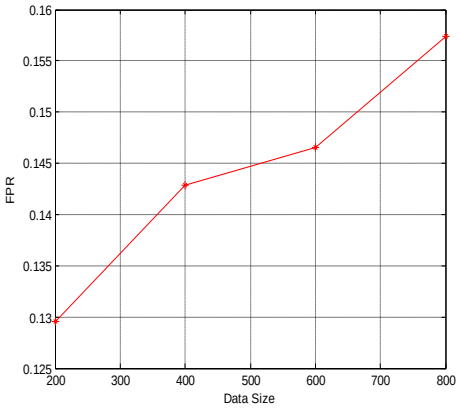


**Fig 7**

**-Comparison of TNR and Datasize**

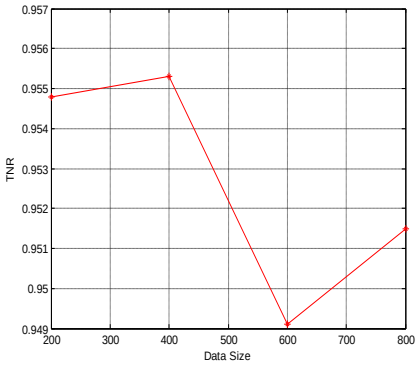


**Fig 8**

**-Comparison of TPR and Datasize**

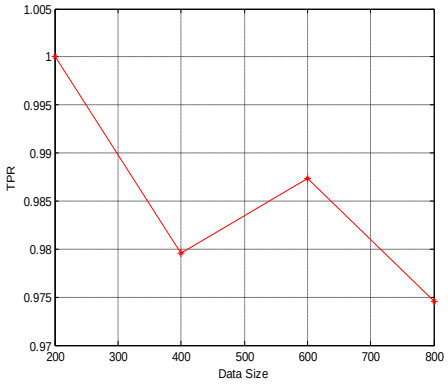


**Fig 9**

As from the result shown above ,the it is found that ,when data size increase detection time also increases but accuracy is increased.TPR,TNR,FPR and FNR also shows the accuracy and efficiency of our system**.**

| | |
|---|---|
| Accuracy | 96.47% |

**Fig 10.**

**Accuracy of our system is 96.47 % As shown in fig. 10 and which is the highest among the existing Sql-Injection detection techniques**

## 5. CONCLUSION

Our concept provides a secure application, based classification of original and suspecious query strings using SVM. Here dataset of different size is used for training and classification .different parameters like Accuracy, detection time ,training time ,TPR,TNR,FPR,FNR and the graphical description shows the performance of our system. Our system shows the best perforamance result in accuracy which is 96.47% and best among the existing systems..

## REFERENCES

[1] A Classification of SQL Inject ion At tacks and Countermeasures: William G.J. Hal Fond and Alessandro Orso, Col lege of Computing, Georgia Institute of Technology.Gatech.edu.

[2] D. Scott and R. Sharp, “Abstracting Application-level Web Security”, In Proceedings of the 11th International Conference on the World Wide Web (WWW 2002), Pages 396–407, 2002.Y. Huang, F. Yu, C. Hang, C. H. Tsai, D. T. Lee, and S. Y. Kuo.

[3] “Securing Web Application Code by Static Analysis and Runtime Protection”, In Proceedings of the 12th International World Wide Web Conference (WWW 04), May 2004.

[4] SQL Injection Attack Examples based on the Taxonomy of Orso et al.

[5] Xiang Fu, Xin Lu, Boris Peltsverger , Shijun Chen, "A Static Analysis Framework For Detecting SQL Injection Vulnerabilities", IEEE Transaction of computer software and application conference, 2007.

[6] Konstantinos Kemalis and Theodoros Tzouramanis, "Specification based approach on SQL Injection detection", ACM, 2008.

[7] G.T. Buehrer, B.W.Weide and P.A..G.Sivilotti, "Using Parse tree validation to prevent SQL Injection attacks", In proc. Of the 5th International Workshop on Software Engineering and Middleware(SEM '056), Pages 106-113, Sep. 2005.

[8] V.B. Livshits and M.S. Lam, "Finding Security vulnerability in java applications with static analysis", In proceedings of the 14th Usenix Security Symposium, Aug 2005.

[9] William G.J. Halfond, Alessandro Orso, Panagiotis Manolios, "WASP: Protecting Web Applications Using Positive Tainting and Syntax-Aware Evaluation", IEEE Transaction of Software Engineering Vol34Nol, January/February 2008.

[10] W.G. J. Halfond and A. Orso, "Combining Static Analysis and Run time monitoring to counter SQL Injection attacks", 3rd International workshop on Dynamic Analysis, St. Louis, Missouri, 2005, pp.1.

[11] Marco Cova, Davide Balzarotti , Viktoria Felmetsger, and Giovanni vigna, " Swaddler: An approach for the anamoly based character distribution models in the detection of SQL Injection attacks", Recent Advances in Intrusion Detection System, Pages 63-86, Springerlink, 2007.

[12] NTAGW ABIRA Lambert and KANG Song Lin ,” Use of Query Tokenization to detect and prevent SQL Injection Attacks”, IEEE,2010.

[13] Vipin Das 1, Vijaya Pathak2, Sattvik Sharma3,Sreevathsan4,MVVNS.Srikanth5,Gireesh Kumar T,” NETWORK INTRUSION DETECTION SYSTEM BASED ON MACHINE LEARNING ALGORITHMS”, International Journal of Computer Science & Information Technology (IJCSIT), Vol 2, No 6, December 2010.