ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

Prevention of SQL Injection Attack Using Positive Tainting

Sujata S. Wakchaure[1], S. M. Chaware[2]

PG student, Dept. of Computer Engineering, TSSM's BSCOER, Narhe, Pune, Savitribai Phule Pune University, Maharashtra, India

Professor and Head, Dept. of Computer Engineering, TSSM's BSCOER, Narhe, Pune, Savitribai Phule Pune University, Maharashtra, India

ABSTRACT: Security of network frameworks is getting more essential as users confidential and private data are being controlled online and get hacked consistently. The protection of a machine structure is exchanged off at the point when an interruption happens as it may bring about data thievery or developer making the machine structures more vulnerable. There are numerous algorithms which are utilized for the seeking the results on web. Pattern matching system is one of them. Few models consider the detection of obscure assaults with decreased false positives and confined overhead. This paper portrays a system to maintain this kind of control and consequently kill vulnerabilities of SQL Injection. This paper also proposed a discovery and balancing activity strategy for checking SQL Injection Attack (SQLIA) using Aho–corasick pattern matching computation. Main focus of this paper is on positive tainting so detection makes it simple. The rule objective is intrusion detection. Investigations exhibit that proposed system has higher recognition rate than existing structure.

KEYWORDS: SQL injection, database security, pattern matching, dynamic pattern, static pattern.

I. INTRODUCTION

Pattern matching is the system of checking an obvious gathering of string in vicinity of the inclusion of a small number of patterns [1]. As not quite the same as pattern acknowledgment, the match ordinarily must be accurate. The illustrations all around have the structure groupings of pattern coordinating join yielding the ranges of a case inside a series of string, to give way some fragment of the pattern matched, and to put the pattern matched with some other series of string. Pattern matching thought is used as a piece of various applications like web crawler, NLP and spam channels [1].

Web applications use the database at the backend for securing data and SQL (Structural Query Language) for insertion and bringing of data. There are a couple of spiteful codes that can be uniting with the SQL called SQL Injection. SQL Injection is the kind of assault that adventures uncalled for coding of our web applications that allows programmer to infuse SQL orders into say a login structure to allow them to become acquainted with the data held inside the database. SQL injection is a hacking framework which tries to go orders through a web application to execute it with the assistance of backend database [1].

SQLIA has highest need in web based security issues. It is extremely hard to distinguish the SQLIAs. Tautology assault, union queries, timing assault, piggybacked queries, blind SQL injection assaults are the most essential sorts of SQLIA. The current strategies can't distinguish a wide range of assaults they have restrictions like the improvement of the assault rule library, less detection ability furthermore trouble of distinguishing all wellsprings of client input. So there is a need to shield framework from all these assault.

This paper divided into section as: Section II speaks about related work analyzed until now. Section III provides implementation details, algorithm used, statistical design and experimental setup maintained to by this document. Section IV represents results and discussion part. Sections V draws conclusions and provides future perform.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

II. RELATED WORK

Intrusion Detection System (IDS) is one of the techniques to prevent SQLIAs but, it do not support service oriented applications. It only support network based attacks because it is working at lower layer [1]. Intrusion detection can termed as of detecting actions that attempt to threat the privacy, reliability and accessibility of the resources of a system [2].

AMNESIA was able to stop all of the attacks without generating any false positive for the legitimate accesses. But the primary limitation in AMNESIA is that the technique is dependent only on the accuracy of its static analysis for building query models for successful prevention of SQL injection [3]. Difficulty of identifying all sources of user input is the main limitation of all existing techniques for SQLIA. They need additional improvements in two main aspects: the detection capability and the development of the attack rule library.

The employment of web application is incrementing detailed. The web programs are being applied for looking information on the internet. String algorithm represents an important part for this. Different individuals are getting a taken at development and equipment levels to make example quick looking. By implementing various algorithms in different programs the surmised best algorithms for diverse programs is settled [10].

Several current systems, for instance, separating, information circulation, protecting programming can identify and keep a part of embellish that immediate SQLIAs. In this region, there is list the most significant techniques like Ali et. al. as. Plan [6] holds the hash value strategy to further improve the client authentication technique. They implement the client name and security password principles of hash SQLIPA (SQL Injection Protector for Authentication) design were designed with a particular end objective to test the framework. The client name and security password hash features are created and determined at runtime for the first time the particular customer record is created.

Thomas et. al. [7] suggested a automated organized explanation era computation to leave SQL injection weaknesses. They perform their study utilizing four free tasks. In light of the trial results, their organized connection code had the capacity effectively replace 94 percent of the SQLIVs in some free tasks.

In excessive computerized technique recognition and prevention of SQLIAs occurs. Usually, our technique satisfies objectives by acknowledging “trusted” string in an application and allowing just this reliable string to be utilized to make several aspect of a SQL query, for instance, keywords or administrators. The general element that we utilization to perform this technique is targeted around dynamic tainting, which postage stamps and paths specific information in a venture at runtime.

The recognition of web-based attacks has as of delayed got amazing attention as a outcome of the inexorably basic aspect that web-based organizations are playing. For example, in [8] the specialist display a structure that looks into web records looking for some design of well-known strikes. A different type of evaluation is conducted in [9] where the location technique is synchronized with the web server system itself.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

III. IMPLEMENTATION DETAILS

A. System Architecture:
Following Fig. 1 shows the proposed system architecture.

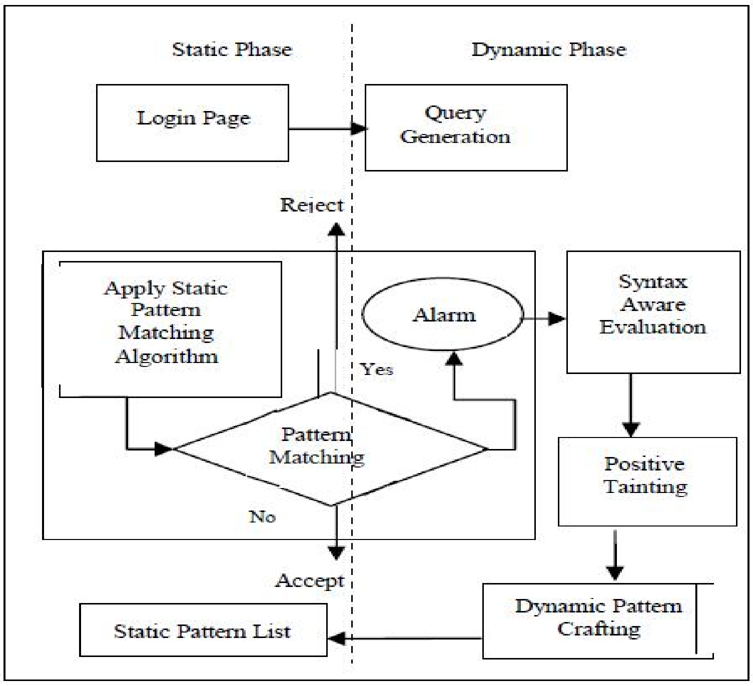

Figure 1. Proposed System Architecture

This Application proposed a detection and preventive action system for avoiding SQL Injection Attack (SQLIA) using pattern matching algorithm and positive tainting. It has two modules, Static Phase and Dynamic Phase.

1. Static Stage:
In this, a static pattern list is maintained. And keep up a list of known anomaly patterns. The client generated SQL queries are checked by applying the Static Pattern Matching Algorithm.

2. Dynamic Stage:
In Dynamic Stage, for unknown attack query if static stage failed then query is checked in the dynamic stage. First, query is converted into tokens then by using anomaly score untrusted tokens are removed by using trusted tokens the query is generated and executed.

B. Algorithm:
Algorithm 1for Static Pattern Matching Algorithm

```
Step1: SPMA (Query, SPL [ ])
INPUT: Query → User Generated Query
SPL [] → Static Pattern List with m Anomaly Pattern
Step2: For j = 1 to m do
Step3: If (AC (Query, String .Length (Query), SPL[j] [0]) ==))
Step4:
```

$$\text{Anomaly}_{\text{score}} = \frac{\text{Anomaly}_{\text{score}}(\text{Query},\text{SPL}[j][0])}{\text{String Length}(\text{SPL}[j])} * 100$$

```
Step5: If ($Anomaly_{score}$ ≥ Threshold value) then
Step6: Return Alarm → Administrator
Else
Step 7: Return Query → Accepted
End if
Step 8: Return Query → Rejected
End if
End For
End Procedure
```

Algorithm 2 for Aho - Corasick Algorithm

```
Step 1: Procedure AC (y, n, q0)
Step 2: Set of all Queries.
Step 3: For All Queries i = 1 to n do
Step 4: Check with Static pattern matching
Step 5: If (Detected (True)) show result
Step 6: Else Send For Dynamic Pattern Matching
Step 7: Tokenize the query.
Step 8: Convert token into pattern matching syntax by using syntax aware
Step 9: For each token match with patterns
Step 10: Detect anomaly score for the query
Step 11: If (Anomaly Score < Threshold)
Step 12: Reject Query
Step 14: Else Start Positive Tainting
Step 15: Remove the attack pattern tokens
Step 16: After token removal combine all tokens
Step 17: Execute Query
Step 18: End for
Step 19: End Procedure
```

First the input for the algorithm is the SQL queries there may be probability of existence of attacker design in the query. To eliminate the attacker design and to identify unlawful query we are using two algorithms, static pattern matching algorithm and dynamic pattern matching algorithm. Initial the query is examined in static pattern Matching (SPM) algorithms. SPM is the known type of attacks that is formerly recognized pattern and the present query is matched with past query pattern. If the query is not recognized in SPM then the query is sent to Dynamic Pattern Matching (DPM).

In DPM query is separated into number of tokens and then tokens are turned into patterns i.e. format aware assessment and then we are determining the anomaly score for the whole query. If the ranking is lower than threshold then we decline that query and if the ranking is higher than threshold then we are doing beneficial tainting, which is a procedure of elimination of attacker pattern i.e. un trusted pattern from the query and to perform the query with trusted pattern.

C. Mathematical Model:

Let S, be a system such that,
S = {s, e, X, Y, T, fme, DD, NDD, ffriend,
MEMshared, CPUCoreCnt, ϕ}
Where,
S- Proposed System

s- Initial state at T (init) i.e. constructor of a class i.e. User Query pattern () function.
e- End state of destructor of a class i.e. Stored Query pattern () function.
X- Input of System i.e. marked patterns.
Y- Output of System i.e. Trusted Patterns.

T- Set of serialized steps to be performed in pipelined machine cycle. **-** User Query pattern, Stored Query Pattern, Marked Pattern, Anomaly Detection, Trusted Query pattern.
fme- Main algorithm resulting into outcome Y, mainly focus on success defined for the solution i.e. Pattern Matching Algorithm.
DD- Deterministic Data helps identifying the load store function or assignment function. i.e. Marking patterns i.e. identifying trusted or untrusted queries.
NDD- Non Deterministic Data of the system is time required to match user's query with static pattern list and mark whether it is trusted or untrusted.
Ffriend- Set of random variables.
MEMshared- Memory required processing all these operations, memory will allocate to every running process.
CPUCoreCnt- More the number of count doubles the speed and performance.
ϕ− Null value if any.

D. Experimental Setup:

The system is manufactured utilizing Java framework (version JDK 8) on Windows platform. The Netbeans (version 8) is utilized as development tool. The framework doesn't require any particular hardware to run; any standard machine is equipped for running the application.

For the segments setup and algorithm execution Mysql database is useful and to apply SPM and DPM algorithms java language and Netbeans IDE is useful and more flexible. Query Submit Component posting the query to the Mysql database using Msql format for the query creation. Before the entirely presented and implemented by Mysql data source it is analyzed by SPM and DPM. In static pattern matching algorithm the entirely first turned into the regular expression by using design transformation classes in java. After the transformation it is matched with previous attack pattern stored in Mysql database.

IV. RESULT AND DISCUSSION

A. Dataset:

In this work numerous queries are applied. For generation result diverse query inputs to the system are useful and diverse set of outputs are generated for different inputs.

B. Results:

Table 1 shows the predicted outcomes of detection accuracy from static and dynamic pattern matching algorithm. Dynamic pattern matching with beneficial tainting enhances the detection precision of SQL injection attack. And the accuracy for the static and dynamic pattern matching is developed as;

$$AC_{SPM} = \frac{\text{No of Queries attack Detected}}{\text{Total No of queries submitted}} \times 100 \qquad \text{…. eq. (1)}$$

AC_{SPM} is the precision for the static pattern matching algorithm which developed as for how many queries the attack is accurately recognized is separated by complete queries presented by user. And for dynamic pattern matching if the query is not detected by SPM then it is sent to the DPM for attack detection and hence it is developed as follows,

$$AC_{DpM} = AC_{SPM} + \frac{\text{No of Queries attack Detected}}{\text{Total No of queries submitted}} \times 100 \qquad \text{…. eq. (2)}$$

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

As per the results DPM with positive tainting gives more accurate results in addition to SPM.

Table 1. Detection Accuracy

Sr. No.	Detection Accuracy in %		
	No. of Query	*Static pattern matching*	*Dynamic Pattern Matching(with Positive Tainting)*
1	1	100	100
2	2	50	50
3	3	67	100
4	4	50	75
5	5	60	80
6	6	67	100
7	7	71	86
8	8	63	88
9	9	78	89
10	10	70	100

In the above table 1 analysis of the system is shown. For each diverse input the detection accuracy is evaluated. In proposed system outcome gets enhanced than existing system in terms of accuracy rate. Following figure 2 describe graph for detection accuracy according to table 1.

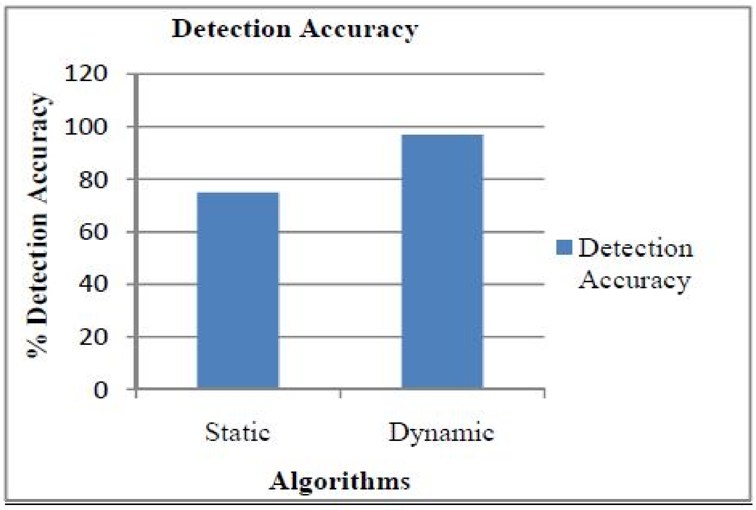

Figure 2. Detection Accuracy Graph

Table 2 shows the results of prevention accuracy. And this accuracy is calculated as,

$$PAcc_{SPM} = \frac{\text{No of SPM Queries}}{\text{Total No of queries}} \times 100 \qquad \text{…. eq. (3)}$$

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

$PAcc_{SPM}$ is the prevention accuracy for the static pattern matching algorithm which shows how many queries are prevented accurately out of total number of queries. And for dynamic pattern matching $PAcc_{DPM}$ is calculated as follows,

$$PAcc_{DPM} = \frac{\text{No of DPM Queries}}{\text{Total No of queries}} \times 100 \qquad \text{…. eq. (4)}$$

Similarly, for altered queries $PAcc_{Alt}$ is calculated as below,

$$PAcc_{Alt} = \frac{\text{No of Altered Queries}}{\text{Total No of queries}} \times 100 \qquad \text{…. eq. (5)}$$

Table 2. Prevention Accuracy

Prevention Accuracy			
Users	**SPM**	**DPM (Positive Tainting)**	**Altered Query**
User 1	90	56	40
User 2	45	75	63
User 3	80	60	40
User 4	54	95	30

In the above table 2 prevention analysis of the system is shown. For each user input the prevention accuracy is evaluated. Following figure 3 describe graph for prevention accuracy according to table 2.

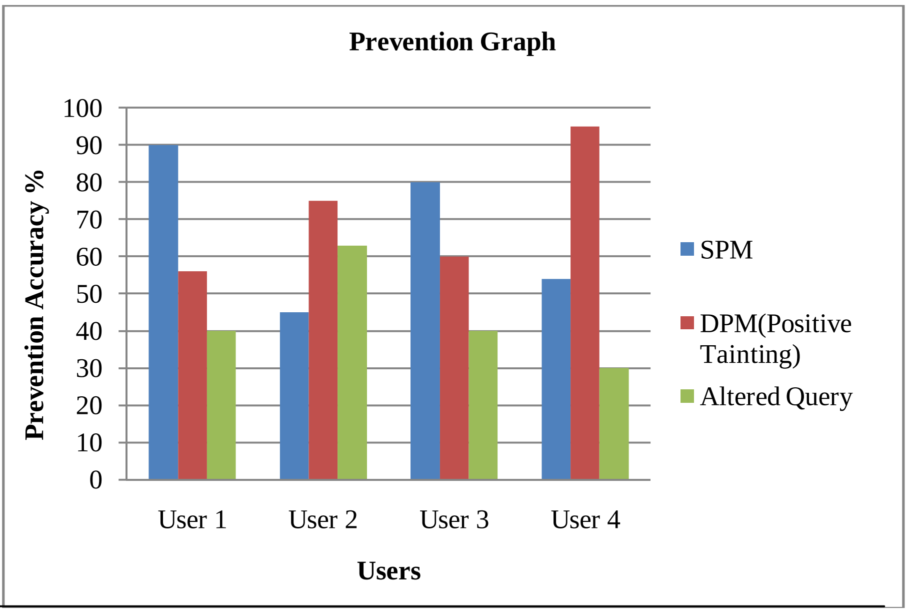

Figure 3. Prevention Accuracy Graph

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Vol. 3, Issue 6, June 2015

V. CONCLUSION

This structure avoids attacks like SQL control and also visible SQL injection. This paper furthermore recommend beneficial tainting changes from traditional tainting, regardless of the way that it is focused around the recognition, checking, and emulating of trusted, rather than non-trusted, information. Furthermore syntax-aware assessment is using the taint marks to understand authentic from harmful queries. This paper also present a strategy for preventive and recognition action of SQL injection attacks using Aho corasick pattern matching algorithm and Positive tainting technique. In future it is possible to use graphical passwords for login, so that it will also not get hacked by attacker and can provide more secure authentication. Also it will be useful to study alternative prevention technique for SQL Injection Attack to make the application more efficient.

REFERENCES

[1] M. A. Prabakar, M. KarthiKeyan, K. Marimuthu, “An Efficient Technique for Preventing SQL Injection Attack Using Pattern Matching Algorithm”, IEEE Int. Conf. on Emerging Trends in Computing, Communication and Nanotechnology, 2013.

[2] S. F. Yusufovna., Integrating Intrusion Detection System and Data Mining, International Symposium on Ubiquitous Multimedia Computing, 2008

[3] G. Vigna, W. Robertson, V. Kher, and R.A. Kemmerer. “A Stateful Intrusion Detection System for World-Wide Web Servers”, In Proceedings of the Annual Computer Security Applications Conference
(ACSAC 2003), pages 34–43, Las Vegas, NV, December 2003.

[4] Halfond, W. G. and Orso, A , AMNESIA: Analysis and Monitoring for Neutralizing SQL-Injection Attacks, in Proceedings of the 20th IEEE/ACM international Conference on Automated Software Engineering, 2005

[5] Nimisha Singla, Deepak Garg,”String Matching Algorithms and their Applicability in various Applications”, International Journal of Soft Computing and Engineering (IJSCE), Volume-I, Issue-6, January 2012.

[6] Shaukat Ali, Azhar Rauf, and Huma Javed “SQLIPA: An authentication mechanism Against SQL Injection”.

[7] S. Thomas, L. Williams, and T. Xie, “On automated prepared statement generation to remove SQL injection vulnerabilities”, Information and Software Technology 51, 589–598 (2009).

[8] M. Almgren, H. Debar, and M. Dacier, “A lightweight tool for detecting web server attacks”, In Proceedings of the ISOC Symposium on Network and Distributed Systems Security, San Diego, CA, February
2000.

[9] M. Almgren and U. Lindqvist, “Application-Integrated Data Collection for Security Monitoring”, In Proceedings of Recent Advances in Intrusion Detection (RAID), LNCS, pages 22–36, Davis, A, October 2001. Springer.

[10] Georgy Gimel'farb,”String matching Algorithms”, COMPSCI 369 Computational Science.

[11] William G.J. Halfond and Panagiotis Manolios,WASP: Protecting Web Applications Using Positive Tainting and Syntax-Aware Evaluation, IEEE TRANSACTIONS ON SOFTWARE ENGINEERING, VOL. 34, NO. 1, JANUARY/FEBRUARY 2008

BIOGRAPHY

Sujata S. Wakchaure has received the Bachelor’s degree in Computer Engineering from Amrutvahini College of Engineering, Sangamner, India in 2013 and perusing master’s degree in Computer Engineering from TSSM’s Bhivrabai Sawant College of Engineering, Pune, India. Her research interest is networking, security and data mining.

Prof. S. M. Chaware is a full time Professor and Head at Department of Computer, TSSM’s BSCOER, Narhe, Pune, India. He has completed Ph.D in Computer Engineering and his research interests are Information & Network Security, Natural Language Processing (NLP) and Information Retrieval (IR).