Applied Machine Learning to Predict the SQL Injection Attacks

THOTA BHAVANI [#1], K.VENKATESH [#2]

[#1] MCA Student, Master of Computer Applications,

D.N.R. College, P.G.Courses & Research Center, Bhimavaram, AP, India.

[#2] Assistant Professor, Master of Computer Applications,

D.N.R. College, P.G.Courses & Research Center, Bhimavaram, AP, India.

ABSTRACT

In current days for any small scale or large scale enterprise companies, their data plays one of the important assets and this is very important for each and every enterprises. As we all know that as data is increasing day by day there are a lot of attackers who try to create some sort of attacks on that data by injecting some traits. One among the several attacks in enterprise level is SQL injection attack, which will be injecting the fake contents inside the business server and try to alter the queries based on intruder choice. In general it is very difficult for the network administrator to trace the difference between the normal SQL query and Injected SQL query which is triggered for the enterprise server. In this paper we try to design SQL injection detection based on SQL tainting method, which can greatly identify the intruders who try to create SQL attacks on the employee tuples and try to differentiate the genuine query and abnormal query easily in the run time. Our Experimental results clearly state that this approach is best in identifying the attackers dynamically and detect the type of injection strings applied to gather the sensitive information from the business server.

Key Words: SQL Injection, SQL Tainting, Network Administrator, Attacks, Employee Tuples, Intruders.

1. Introduction

SQL injection is term refereed to injecting the un-usual patterns or strings into the SQL queries and tries to gain the illegal access from the business database and retrieve the information in un-authorized manner [1]. This is mainly injected into the database at the time of substituting the username and password for user authentication. During the process of user authentication or registration of user details these SQL queries are injected into the database and they will try to control the database by injecting these type of queries[2]. Here the attacker tries to inject some extra string (txtUserId) for that input query and this variable will be fetched from the input query based on the statement like (getRequestString):

FOR EXAMPLE

A network administrator is asked with a task to retrieve the information about an Employee [3]who is present in that employee tuple and he is holding with an ID : 1234.In order to retrieve that employee record from the EMPLOYEE tuple ,we will construct a SQL query as Follows:

```
SELECT * FROM EMPLOYEE WHERE EID = 1234 ;
```

Here EID = Employee Id Attribute which is declared on the EMPLOYEE Tuple

This Query will try to execute successfully by providing all the information related to the EID =1234 and the resultant data will be displayed as result for the end user.

The Result for the above query is displayed as follows:

EID	EmpName	Department	Contact No	Email Id
1234	Eeshan	IT-Developer	1234567890	isha@gmail.com

Here we can see the employee name “Eeshan” details are retrieved because the EID is matched with this corresponding employee Id.

Now if any attacker[4] try to attack SQL queries by injecting some special strings into the Employee Tuple, he can able to receive the same data in this manner for that corresponding users records.

```
SELECT * FROM EMPLOYEE WHERE EID = 1234 Or 1=1 ;
```

The Result for the above SQL Injection query is displayed as follows:

EID	EmpName	Department	Contact No	Email Id
1234	Eeshan	IT-Developer	1234567890	eeshan@gmail.com

Here the above query will try to execute and display the records which are matched with the EID=1234 because the special string like 1=1 is originally true factor and this will be executed he original purpose of the code was to create an SQL statement to select a user, with a given user id.In the process of SQL Injection, the operator ‘=’ is always true operator and this will be executed as positive parameter by the SQL query analyzer and they try to display the output which is relatively matched with that query keyword[5].

In some cases a hacker might try to inject an empty strings with the operators like " OR ""=" ,so that the access for the username and password is granted by that sql query analyzer for that appropriate queries because of the true parameters 'OR' and '=' present in the SQL query analyzer.

In current days a SQL injection, also known as SQLI is most common attack which is used by the attacker in order to inject the malicious code from the backend and try to change the content which is present in the database in a manipulated manner. Here the data which is present in the enterprise server may some time contain employee personal information, annual turnover information, salary, profit and loss information, import and export details, clients and their confidential MOU's and a lot more[6]. Till now no approach is able to identify these SQL attacks in run time and identify in which way the query is impacted for the enterprises. All the approaches try to identify these types of attacks in later stage rather than identifying on the spot situation[7].

2. LITERATURE SURVEY

Literature survey is that the most vital step in software development process. Before developing the tool, it's necessary to work out the time factor, economy and company strength. Once this stuff is satisfied, ten next steps are to work out which OS and language used for developing the tool. This literature survey is mainly used for identifying the list of resources to construct this proposed application.

MOTIVATION

In general for accessing the login form ,all the traditional web sites require username and password form for entering into the application but for the email id field of that page,there is an alternate link provided by the web developers in order to forget my password and retrieve the password at the time of need[8].

In order to resend the mail id after verifying the forget answers which are entered by that concern user; we need to check in the user database table where the queries and answers are initially feeded into the system. If the details are found as correct the system will collect all the details and send the details to that corresponding user mail id, if the same user details are not matched then we may get message as email address is not found, it wasn't going to send **me** anything[9].

Initially we try to text a single quote in any of various SQL databases and then try to enter a single quote additionally while inserting the data. At the time of submitting the form with a quote in the email address, sometimes we may get a 500 error (I.e. It indicates the server is failed) and so the data cannot be retrieved due to that broken query[10].

For Example:

The following is the query which is submitted for retrieving the details based on E-Mail Field.

SELECT fields From Table Name where field =’$Email’ :

Here, **$EMAIL** is the address submitted on the form by the user, and the larger query provides the quotation marks that set it off as a literal string.

Immediately we will get this result with the help of that SQL query

EID	EmpName	Department	Contact No	Email Id
1234	Eeshan	IT-Developer	1234567890	eeshan@gmail.com

When the above query is executed in the SQL command prompt, the SQL parser will try to find the query and remove those extra quotes and then manifest the query according to the field ‘Email Id’ and then try to execute the query with the search keyword Email Address.

In another way the same query can also be called to the SQL database by the intruder by adding some extra strings in the place of Email Field. This can be possible as follows:

By entering **anything' OR 'x'='x**, the resulting SQL is:

SELECT fields From Table Name where field =’ **anything' OR 'x'='x**’

If we run this query in the sql window the data will be displayed in same way like how it executed for the normal query.

EID	EmpName	Department	Contact No	Email Id
1234	Eeshan	IT-Developer	1234567890	eeshan@gmail.com

In this way the SQL injections are creating a lot of problem for the end users to access the sensitive information illegally from the enterprise database and they are creating a huge problem for the companies by modifying the important fields and gaining the illegal access about main contents which are present in the database.

So in this proposed application we try to use the SQL tainting method to identify such SQL injection attacks and try to provide an facility for the user to identify the type of query whether it is executed normally or it is executed in Abnormal manner. If this is found we can able to know the exact condition about these SQL queries.

3. PROPOSED SQL TAINTING METHOD FOR IDENTIFYING SQL INJECTION QUERIES IN RUNTIME

In this section we will mainly discuss about the proposed SQL Tainting approach for identifying the SQL Injection queries in runtime from the MNC company database. Now let us discuss about this proposed model in detail as follows:

MOTIVATION

In general it is very hard for the end user to identify the data contains any injection or abnormality present inside the content.

Almost all the data will be having the information in binary manner.I.e Either true or false and the data user can able to identify with this two cause. If there is any attacker present inside the data, the data will be appeared as Attacked and this can be identified in two circumstances:

1. One is Static Taint Analysis
2. Second one is Dynamic Taint Analysis

Normally this dynamic tainting method is identified with three parameters like:

A) Taint Seed
B) Taint Tracker
C) Taint Assert

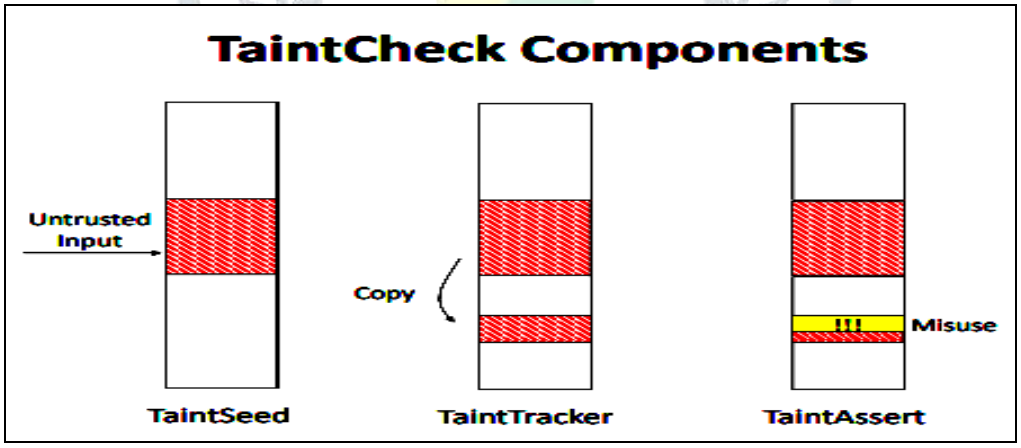


Figure.1. Represents the Flow of SQL Tainting in Run Time

From the above figure 1, we can clearly identify the taint check components which are present in the process of dynamic tainting for identifying the sql injection attacks which is present in the sample input dataset.

TAINT SEED

This is first component in the process of SQL injection attacks and this is used to monitor the input via system calls and also this will be identified the frequent untrusted inputs which are present in the sample dataset.

TAINT TRACKER

This is the component which is used to identify the data movements which is present inside the data. This will basically generate the operations like e.g., move, load, store, etc. This will also identify some Unicode and arithmetic operations which are present in the sql query like add, xor, mult, etc. Once if any such special characters or symbols are present in the sql query then this is identified as sql query injection.

TAINT ASSERT

This is the final function which is used for identifying the intruders who try to create sql injection attack. This will be identified based on trained data which is been misused by the end users by injecting some informal strings inside the queries and all these kinds of data is identified and marked by the taint assert. This method will be invoked automatically if there is any attack present inside the network.

4. IMPLEMENTATION PHASE

We have implemented the proposed concept on Java programming language with JEE as the chosen language in order to show the performance this proposed model. The front end of the application takes JSP, HTML and CSS, and as a Back-End Data base we try to use the input from sample dataset which is collected by the user at the time of application deployement. The proposed application is divided into mainly 3 modules; now let us look about them in detail as follows:

1) SQL Query Analyzer Module

In this module we try to write the sql queries by following the database scheme syntax and this will try to connect with the database which is installed in our PC.Once the query is written the user need to choose analyze button so that this query will be verified internally with the dataset which is loaded previously inside the application.

2) SQL Injection Module

In this module the intruder try to inject the SQL attacks by using the WHERE clause component. This where clause will try to determines the type of access level and which type of attack he want to inject.In

real time this SQL query can be validated based on the username and password credentials and the attacker want to create some sort of attacks at this point.

3) Result Analysis Module

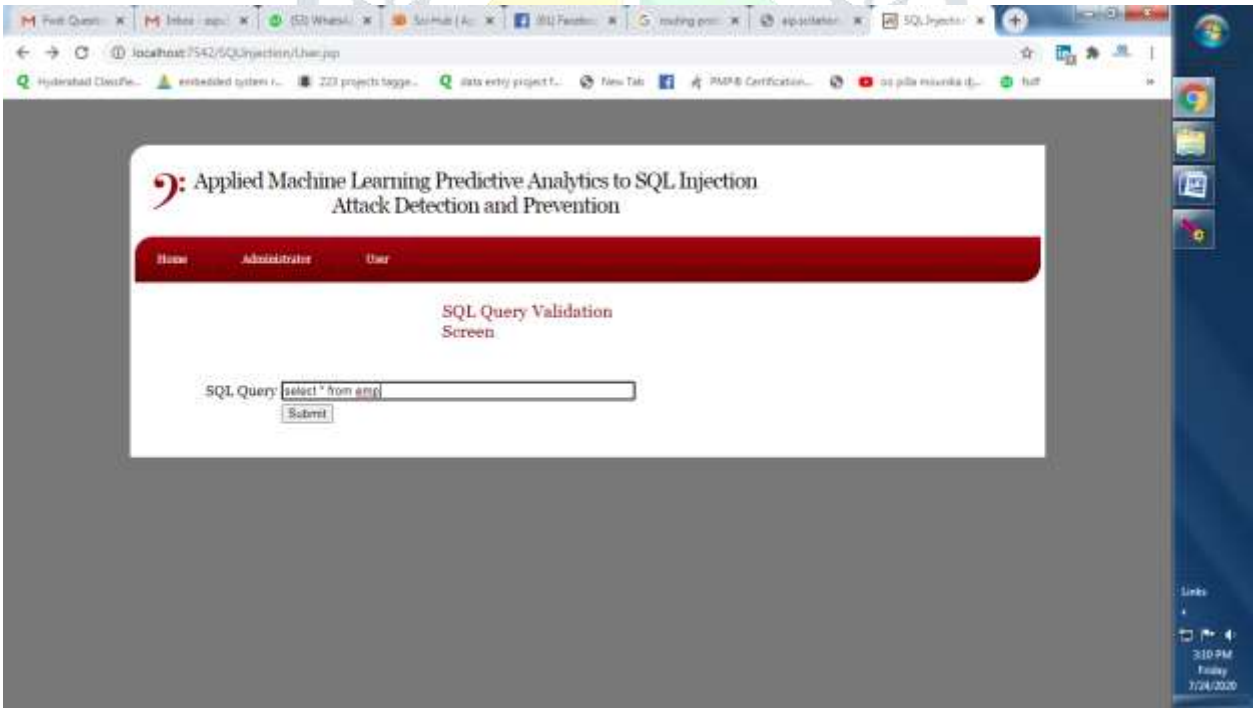
In this current module we will try to display the result by gathering the SQL query which is passed by the query user and then it is tested on the SQL queries sample dataset. Here if the query contain no special characters or keywords, then it is identified as normal query and the data will be returned as Normal.If the same query is identified as Negative then it is labeled as query executed with some abnormal nature.

In this way we can able to identify the SQL injection attacks in run time rather than in later stage.

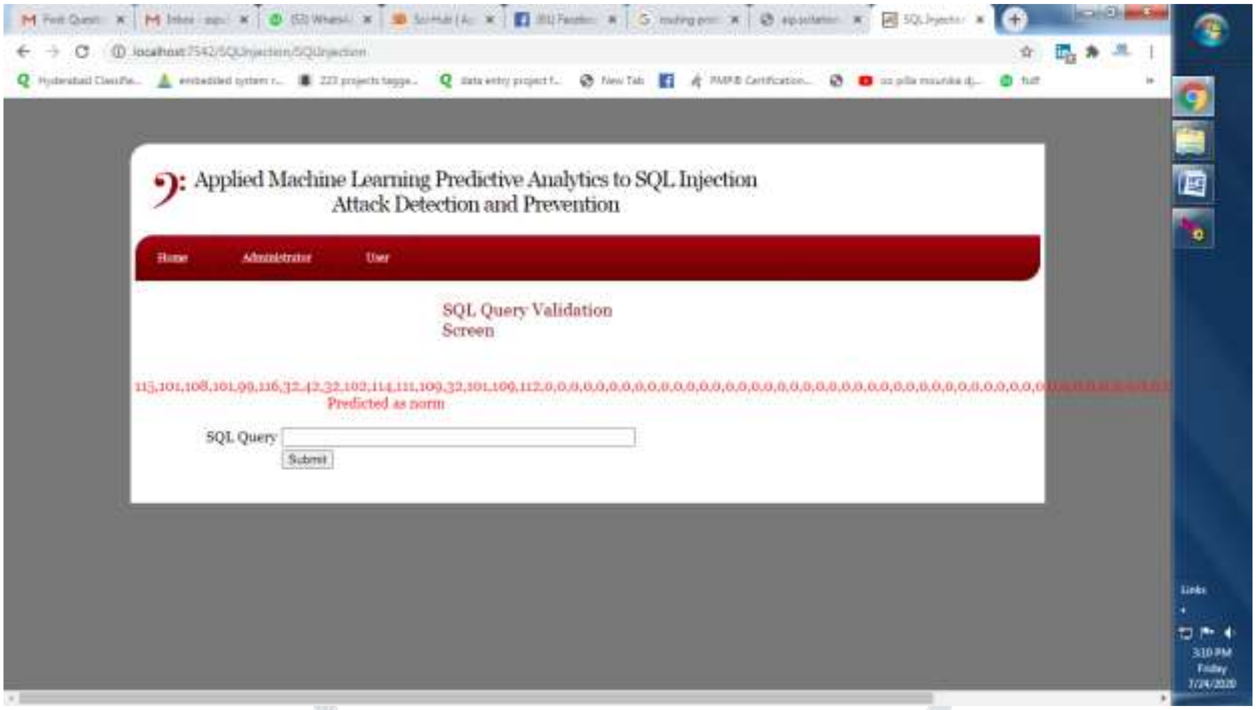
5. EXPERIMENTAL REPORTS

We have implemented the proposed concept on Java programming language with JEE as the chosen language in order to show the performance this proposed model. The front end of the application takes JSP, HTML and CSS, and as a Back-End Data base we try to use the input from sample dataset which is collected by the user at the time of application deployment.

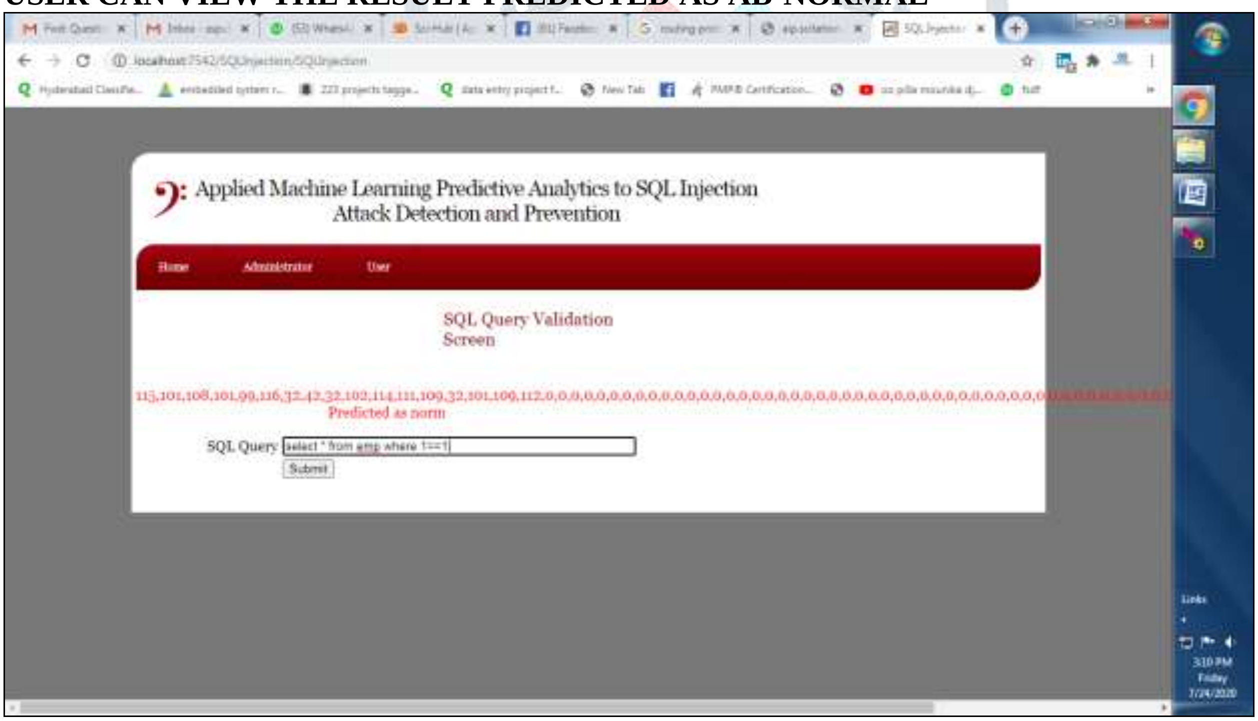
USER SUBSTITUTE THE QUERY



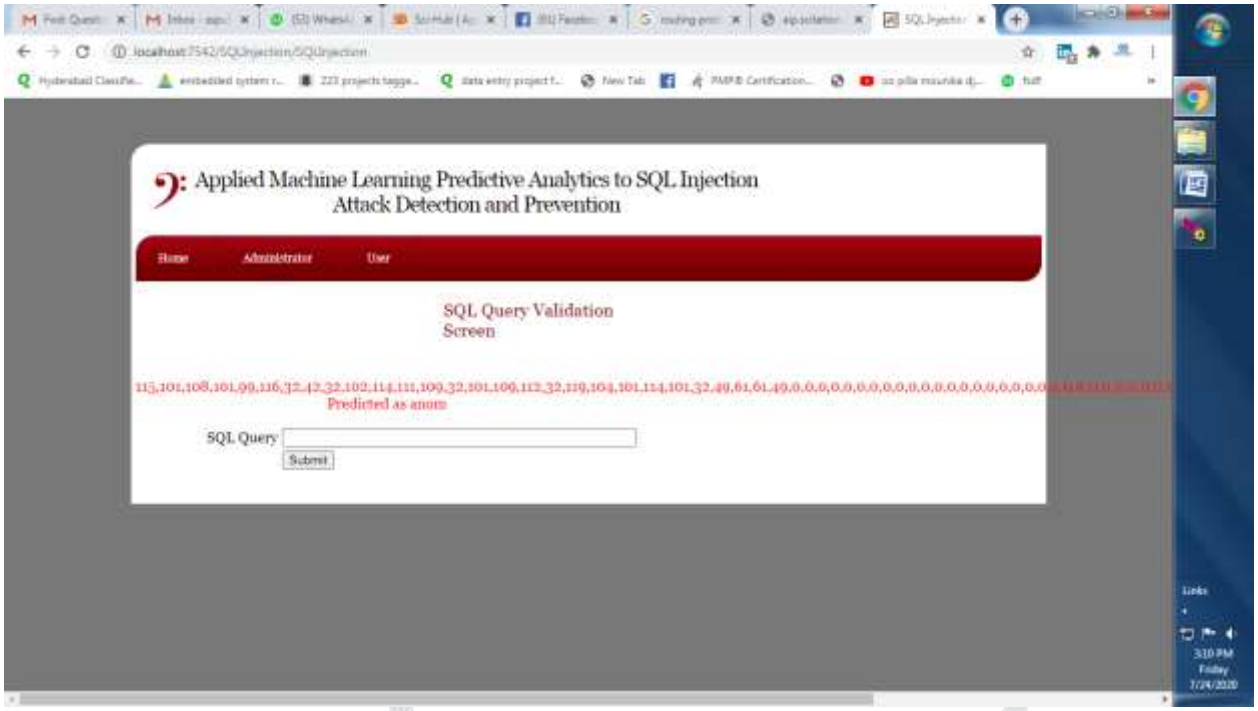
USER VIEW THE RESULT PREDICTED AS NORMAL QUERY



USER CAN VIEW THE RESULT PREDICTED AS AB-NORMAL



USER CAN VIEW THE QUERY AS AB-NORMAL



6. CONCLUSION

In this paper, we for the first time designed a model for SQL injection detection based on SQL tainting method, which can greatly identify the intruders who try to create SQL attacks on the employee tuples and try to differentiate the genuine query and abnormal query easily in the run time. Our Experimental results clearly state that this approach is best in identifying the attackers dynamically and detect the type of injection strings applied to gather the sensitive information from the business server.

7. REFERENCES

[1] Two Well-known authors,E. M. Yuniarno, and M. Hariadi, has written a paper on ``Large scale text classification using map reduce and Naive Bayes algorithm for domain specified ontology building," Published in Aug. 2015, pp. 428_432.

[2] Two Well-known authors ,U. Zahoora, and A. S. Qureshi, has written a paper on ``A survey of the recent architectures of deep convolutional neural networks," Published in 2019.

[3] Two Well-known authors, C. Caballero-Gil, and P. Caballero-Gil, has written a paper on ``Collaborative SQL-injections detection system with machine learning," Published in *Proc. 1st Int. Conf. Internet Things Mach. Learn.*, 2017, Art. no. 45.

[4] *The Ten Most Critical Web Application Security Risks*, Top OWASP 10, Toronto, ON, Canada, 2013.

[5] Two Well-known authors, M. Alshraideh, and K. E. Sabri, has written a paper on ``Detecting and preventing SQL injection attacks: A formal approach," Published in *Proc. Cybersecur. Cyber-forensics Conf. (CCC)*, Aug. 2016, pp. 123_129.

[6] Two Well-known authors, W. J. Buchanan, and L. Fan, has written a paper on ``Applied machine learning predictive analytics to SQL injection attack detection and prevention," Published in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manage. (IM)*, May 2017,pp. 1087_1090.

[7] Two Well-known authors, P. R. McWhirter and B. Askwith, has written a paper on ``SQL injection attack classification through the feature extraction of SQL query strings using a gap-weighted string subsequence kernel," Published in Jun. 2018.

[8] Two Well-known authors ,K. Kamtuo and C. Soomlek, has written a paper on ``Machine Learning for SQL injection prevention on server-side scripting," Published in Dec. 2016.

[9] A Well-known author, S. M. Darwish , has written a paper on ``Machine learning approach to detect intruders in database based on hexplet data structure," Published in 2016.

[10] Two Well-known authors, B. R. Ram, and P. Niranjan, has written a paper on ``SQL injection attack prevention based on decision tree classification," Published in Jan. 2015.

[11] Two Well-known authors ,N. Singh and S. Kumar, has written a paper on ``SQL injection: Types, methodology, attack queries and prevention," Published in Mar. 2016.