A Naïve Bayes Based Pattern Recognition Model for Detection and Categorization of Structured Query Language Injection Attack

Article *in* International Journal of Cyber-Security and Digital Forensics · July 2018

DOI: 10.17781/P002396

CITATIONS
0

READS
230

4 authors:

Morufu Olalere
Federal University of Technology Minna
23 PUBLICATIONS **84** CITATIONS
SEE PROFILE

Abdullahi Egigogo Raji
Federal University of Technology Minna
10 PUBLICATIONS **2** CITATIONS
SEE PROFILE

Ismaila Idris
Federal University of Technology Minna
42 PUBLICATIONS **326** CITATIONS
SEE PROFILE

Jimoh Rasheed Gbenga
University of Ilorin
34 PUBLICATIONS **92** CITATIONS
SEE PROFILE

International Journal of Cyber-Security and Digital Forensics (IJCSDF) 7(2): 189-199
The Society of Digital Information and Wireless Communications (SDIWC), 2018 ISSN: 2305-001

A Naïve Bayes Based Pattern Recognition Model for Detection and Categorization of Structured Query Language Injection Attack

[1]Morufu Olalere [2]Raji Abdullahi Egigogo, [3]Ojeniyi Joseph A. [4]Ismaila Idris, [5]Rasheed Gbenga Jimoh

[1,2,3,4]Department of Cyber Security Science, School of Information and Communication Technology,

Federal University of Technology, Minna, Nigeria.

[5]Department of Computer Science, Faculty of Information and Communication Technology,

University of Ilorin, Nigeria.

[1]lerejide@futminna.edu.ng, [2]raji.pg610868@st.futminna.edu.ng, [3]ojeniyija@futminna.edu.ng,
[4]ismi_idris@futminna.edu.ng, [5]jimoh_resheed@yahoo.com

Abstract— In the recent times, information sharing and delivery of services is done over the internet through different platform of web applications and various attacks are performed against these applications such as Cross Side Script (CSS), Denial of Service (DoS) and Structured Query Language (SQL) injection attacks among others. SQL injection is one among the ten top threats and vulnerabilities against web applications airming backend database. Researchers have proposed many approaches of SQL injection attack, either for the detection/categorization or both, many of the proposed approaches only detect few attack types among the seven most popular attack types and poor training of dataset. In this study, a Naive bayes based pattern recognition model for detection and categorization SQL injection attack type is proposed. The proposed model was trained and evaluated with 16,050 instances of dataset which comprises vulnerable and non-vulnerable web pages. Our experimental results showed detection and categorization accuracy of 98% and 99% respectively. The comparison of the performance of our model with the performance of existing techniques revealed that our model outperformed the previous techniques.

Keywords: Detection, Categorization, Machine learning, SQL injection attack, Naïve Bayes, web application

1 INTRODUCTION

Internet and web applications are essential in today's world as several human activities such as e-banking, e-business, e-reservations and surfing rely on the internet and web application for the proper running of day to day activities [1]. Web application is the development system using various scripting and programming languages such PHP and JAVA and for numerous purposes [2]. These developed systems worked alongside to store information in a database such as organizations confidential data, credit card numbers, social networking and bank details. Acc ording to [3], attackers compromise the security of the web application by taking the advantages of the loopholes through unauthorized access and intuitively ploy the backend database so as to read, write and update records. Hereafter, attainin g complete control which destroyed online web application system and databases through SQL injection attacks [4]. Figure 1 below illustrated the working principle of web application.

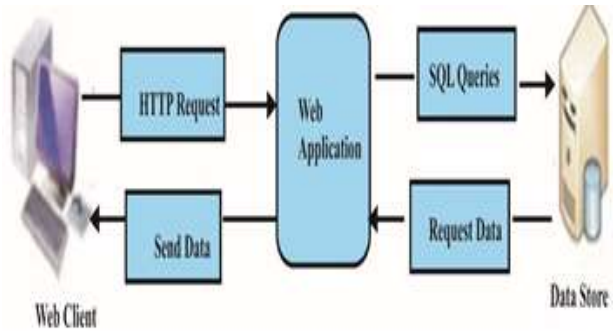

Figure 1: Working principle of web application, [5].

Structured Query Language Injection Attack (SQLIA) is a way used by intruders to inoculate an input into the structure of a query with the intent of altering the structure of the query and gaining access to the database which causes

modification or erasure of data [6]. This occurs as a result of non-standard error reporting, improper construction of SQL statements and insufficient input validation. Furthermore, SQL injection is categorized by the Open Web Application Security Project (OWASP) in 2010, 2013 and 2017 as one among the top ten threats and vulnerabilities in web applications that targeted backend databases. Even though many existing approaches are available for the detection and categorization of SQL injection, vulnerabilities are inherent in web applications and reports of compromised applications are recurrently circulated. Several existing techniques only detect few among the seven most popular injection attack types (such as tautology, illegal/logical incorrect queries, piggy blocked, union, stored procedure, inference and alternative encoding), extracting features inappropriately as well as the use of few datasets to evaluate the efficiency [7] and poor training of dataset which lead to generation of false positive rate [5]. As a result of these pitfalls, a Naive bayes based pattern recognition model is proposed to detect SQL injection attack and categorize SQL injection attack type.

2 RELATED WORKS

In order to prevent and detect SQL injection, [8] proposed a neural neetwork approach which offers a new scheme for securing an information so as to avoid difficulty in transmission over network. The authors created a dictionary consisting of malicious and input data of Uniform Resource Locator (URL) with parameters such as URL keywords, toxic factor based on auto downloading links and number of download links. Matrix Laboratory (MATLAB) was used as simulation tool. Also, [9] proposed a Neural Network(NN) model to detect and classify SQL injection. The model has three element such as Uniform Resource Locator (URL) generator, URL classifier and an NN model. These were used to generate, classify and detect thousands of malicious and benign URLs. This was implimented with MATHLAB.

The author [10], conducted a research on SQL injection attack and user behavior detection using query tree, fisher score (for selection of features) and Suport Vector Machine (SVM) (for detection). The proposed approach used simple dataset to test the scheme with several queries. In their findings, redundant features were removed. The authors [11] developed a scanner to prevent SQL injection attacks which reflected Cross Site Scripting (CSS) in web environment. The approach verifies, scans and input entry points with respect to an array vulsig () array. The scanner has two modules: ispot identification and instrumentation.

Similarly, [12] developed SQL Attack Scanner (SQLAS) for detection and prevention of SQL injection attack in web applications. The tool does its scanning offline, with reduced time and manual effort and runtime overhead. This is so, because, only fragments that are exposed to attacks are focused. A test-bed was developed using JAVA and was ran on XAMPP client server. Another work by [13] on detection and prevention of SQL injection attack uses fuzzy parameters and set of rules were implemented. The approach was not evaluated.

An algorithm for detection and mitigation of SQL injection was developed by [1], which authenticate the webpages through URL/HTML controls. The technique also does reconstruct queries automatically.

In order to prevent SQL injection attack, [14] proposed an encryption algorithm based on randomization to prevent SQL injection attack. The authors use this approach to transform the input into a cipher text integrating the idea of cryptographic salt.

The study of [15], focused on prevention of SQL injection attacks through defensive coding. The defensive coding approach was utilized to mitigate the major underlying driver of inadequate input validation. The defensive coding is executed utilizing key accepted procedures like input type checking, encoding the data sources, positive input matching and identification of all input sources among others.

AMNESIA is an apparatus that identifies and averts SQL infusion attacks by consolidating static examination and runtime checking. The Experimental assessment has demonstrated that AMNESIA is both compelling and effective against SQL injection [16]. These authors [17], worked on various studies of SQL-Injection web vulnerabilities detection and prevention techniques with their characteristics, and proposed an authentication technique to avoid such attacks which was implemented in stored procedures using Regular Expression.

An automatic query sanitization technique called Automatic and Static SQL Injection Sanitization Tool (ASSIST) was proposed by [18] to automatically eliminate SQL injection vulnerabilities in code. The procedure utilizes a blend of static examination and the program change to automatically instrument web applications with sanitization code. The implementation of the method was done in JAVA and test assessment evaluation indicated that method was compelling against SQL injection vulnerabilities and has a low overhead

In another vain, [19] proposed a method based on query tokenization for identification and prevention SQL injection attacks. This method checks client inputs whether they cause changes in query's expected results. After tokenizing, two arrays are made by all tokens and the lengths of gained clusters are compared. The outcome demonstrated that if there is variety in the lengths, an injection attack is detected. The authors [20], have proposed a framework in light of abnormality detection strategies to identify the pernicious behavior of database application programs. The research work received and made a fingerprint of an application program in light of SQL injection and afterward takes the benefits of association rule mining strategies to remove useful rules from these fingerprints. These rules delineate the normal behavior of the database application. The outcome demonstrated that dynamic queries checks against these rules to detect injection attacks.

The research work [5], proposed two unique systems to prevent and identify SQL injection in web application. For the aversion and identification purpose, the researchers designed SQL Meta Character Filter (MCF) and a network based vulnerability scanner respectively. The result showed that the scanner was able to detect and prevent SQL injection effectively though not 100% efficient. Addendum, [21] proposed a SQL injection vulnerability scanner and filter to prevent and detect injection attacks. The scanner was proven to discover vulnerabilities in a web application with high efficiency of counter attack method. Similarly, [22] proposed a novel approach to detect and block SQL injection in web application. The approach was based on request, receiver and analyzer. The implementation was carried out with RAT which is believed to be cost effective in terms of usage of resources, applicability and time against several methods of SQLIA.

A specialized agent in the detection of SQL injection attacks was presented by [23], the agent incorporates a Case-Based Reasoning Engine (CBRE) that is equipped with a learning and adaptation ability with respect to the classification of vindictive codes alongside with advanced algorithms in the reasoning cycle stages. The reuse stage utilizes a creative classification model in light of combination of a neural system together with a Support Vector Machine so as to classify the retrieved SQL queries in the most solid way. The testing and validation was done in real world traffic. The Support Vector Machine (SVM) was proposed for classification and prediction of SQL-Injection attack with an accuracy of 96.47%. The authors claimed that the approach has the highest detection accuracy among the existing techniques [24]. Similarly, [25] proposed a framework that uses Support Vector Machine (SVM) to group and predict SQL Injection attacks. The approach focuses only on fragments that are vulnerable to injection attacks.

Furthermore, Two-Class Support Vector Machine (TCSVM) is explored to detect and prevent SQL injection attack. The generated dataset contained

extraction from known attack patterns which are SQL tokens and symbols present at injection points. As a test case, a web application was built and expects dictionary word list as vector variables to demonstrate massive quantities of learning data [26].

In order to prevent SQL injection attack, [27] proposed a framework that uses machine learning and a compiler platform. The framework is targeted to prevent piggy-backed queries, Illegal/logically incorrect queries and Union queries on server-side scripting. The authors used four machine learning models which are Support Vector Machine (SVM), Boosted Decision Tree (BDT), Artificial Neural Network (ANN), and Decision Tree (DT). 1,100 samples of vulnerable SQL commands were generated for training of the models. The study of [28], proposed a genetic fuzzy classifier system for detection of SQLI. The authors used initial rules, parameters, an enhancing function and data-dependent so as to create the rules that will have high generalization capabilities and modifies the rule evaluation measures. The model treated SQL statement as a feature vector that characterizes the SQLI attack keywords. The system was evaluated using a number of well-known datasets. The results showed a significant enhancement in the detection procedure. Furthermore, [7] proposed a classifier for the detection of SQL Injection attacks. The proposed model uses combination of Role Based Access Control mechanism and Naïve Bayes machine learning algorithm for detection. It detects tautology, comments and union SQLIA attacks and the test cases were derived from these attacks with an accuracy of 93.3%.

In another research work by [29] a decision tree classification was proposed to prevent the SQL injection attacks. Based on the attack signatures, the model filters the Hypetex Transfer Protocol (HTTP) request sent using a decision tree classification. As such, the model was tested on synthetic data which give satisfactory results [30] introduced a novel approach to dissect the HTTP traffic and inspect complex SQL injection attacks. A Hybrid Injection Prevention System (HIPS) and web application firewall architecture were used. The former uses both a machine learning classifier and a pattern matching inspection engine based on reduced sets of security rules while the latter aim to optimize detection performances by using a prediction module that excludes legitimate requests from the inspection process. Based on the TCR results, the effectiveness of the classifier by tuning its values in order to reduce false negatives was revealed and false positives did not impact the overall performances of the system.

Furthermore, [31] proposed Bayes classification for detection of SQL injection. The authors use keywords rather than statement of SQL query. The Bayes theorem was applied on the keywords which improves the accuracy and performance of the detection. The new approach used injection patterns as its features in detecting and categorizing the injection attack. The authors [32] investigated and organized different detection and prevention of SQL injection attack approaches and proposed an algorithm that focuses on detection and correction using machine learning technologies. A novel approach for learning SQL statements which make use of machine learning techniques, such as one class classification was proposed so as to detect malicious behavior between the application and database. The technique integrates query value similarity, tree structure of SQL queries and input parameter as characteristics to differentiate benign from malicious queries [33].

3 THE PROPOSED NAÏVE BAYES MODEL

Naïve bayes is a simple supervised machine learning model for building a classifier which allocates class labels to problem instances, represented as vectors of feature values, where the class labels are drawn from some finite set. It is not a single algorithm for training such classifiers, but a family of algorithms based on a common principle. The classifiers assume that the value of a particular feature is independent of the value of any other feature, given the class variable [34].

The derivation of Naïve bayes model for this study is as follows:

Let $A=\{s_1,s_2,...,s_n\}$ be a set of features that signifies attack pattern/signature as illustrated in table1 where n symbolizes the total number of the features respectively.

Let each instance $A=\{X_1,X_2,...,X_n\}\in\{0,1\}=P$ where *P* is the total number of instances, 0 denotes non vulnerable and 1 as vulnerable be represented by the set A

is the total number of instances, 0 denotes non vulnerable and 1 as vulnerable be represented by the set of A

Let $C_1\in\{C_1,C_2,...,C_m\}=C$ be a class to which an instance X_j belongs, where $C\in\{V,NV\}$.

In other to detect/predict an instance X_j based on the assumption that the element of set A assumes their values independently on one another.

Also

$$C_i=\arg\max C_i\in P(Ci|s_1,s_2,...,s_n) \quad (1)$$

$$P(C_i|\ s_1,s_2,...s_n)=P(s_1,s_2,s_3...,s_n)\times P(C_i)\div P(C_i|\ s_1,s_2,...s_n) \quad (2)$$

Where $P(s_1,s_2,...,s_n)\neq 0$

Assuming the uniformity of $(s_1,s_2,...,s_n)$, the equation (2) can be simplify into the following

$$P(C_i|\ s_1,s_2,..s_n)=P(s_1,s_2,s_3...,s_n|C_i)\times P(C_i) \quad (3)$$

Using chain rule we have:

$$P(s_1,s_2,s_3...,s_n|C_i)\times P(C_i)\ \times\prod_{k=1}^{n}P(s_k|C_i) \quad (4)$$

Therefore, the web application is classified to a particular class C_i which gives

$$C_i=\arg\max C_i\in P(C_i)\prod_{k=1}^{n}P(s_k|C_i) \quad (5)$$

Where the probability $P(C_i)$ is estimated by the frequency of instances belonging to C_i in the training dataset.

Table 1 below depicts the attack type and patterns/signatures

Table 1: Attack type and Patterns/Signatures

Attack pattern			Attack type
‘	incorrect logics	--	Tautology
or	and	ascii ()	Illegal/logical incorrect queries
=	Orderby	bin ()	Piggy blocked
like	;	hex ()	Union
select	union	unhex ()	Stored procedures
convert	union select	base64 ()	Inference sql attack
int	shutdown	dec ()	Alternative encoding
char	exec	rot13 ()	Non-vulnerable
Varchar	xp_cmdshell ()	*	
Else	if	Waitfor	
Varchar	sp_execwebtask ()		

4 EXPERIMENTATION

To effectively evaluate and compare the efficiency of the proposed naïve bayes based pattern recognition model for detection and categorization of SQL injection attack with existing studies, a machine learning algorithms was adopted. The Figure 2 below shows the experimentation flow chart of our proposed study.

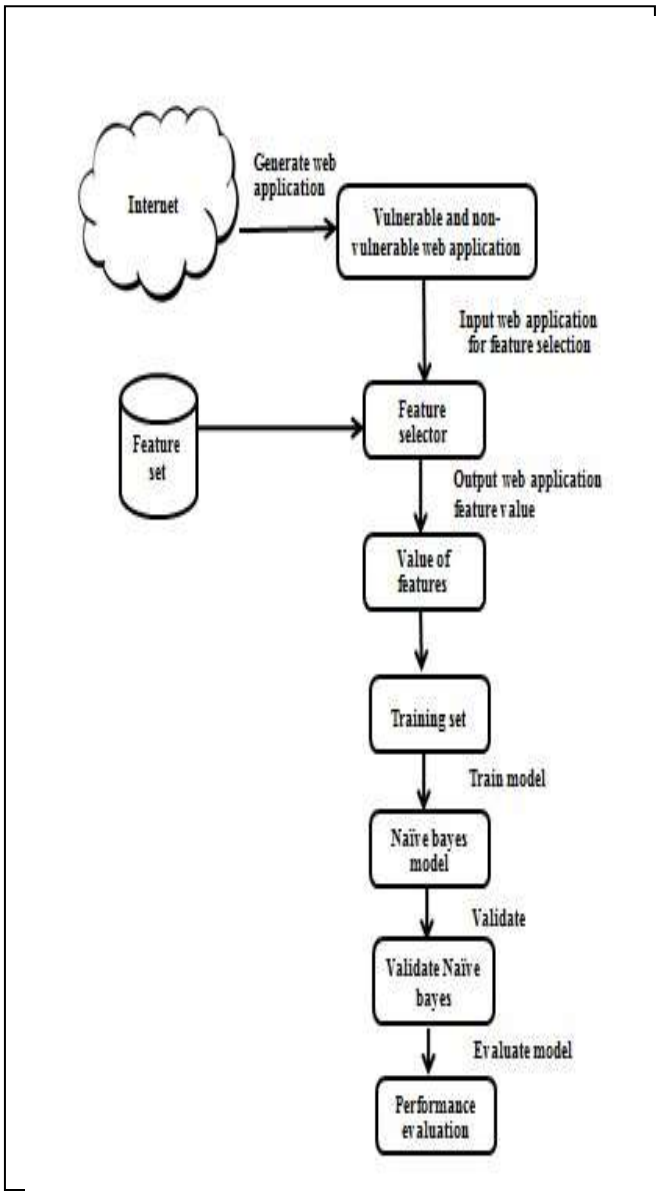

Figure 2: Experimentation flow chart

4.1 Data Collection

As a result of unavailability of SQL injection attack dataset, 16,050 webpages/URLs were generated from dmoz. Dmoz is an open source directory project that houses good web pages/URLs which made it to be popular and contains several classes of webpages/URLs topics. In order to generate these webpages/URLs, a web crawling tool called link klipper was used.
Out of 16,050 instances of dataset, 7,000 are vulnerable and 9,050 are non-vulnerable. Seven most popular injection attack types were considered with their signatures/patterns comprising of 31(see table 1).

4.2 Feature Selector

The features of the data were obtained from the seven most popular injection patterns/signatures (see Table 1). These signatures/patterns are the features of the dataset.

4.3 Value of the Features

The features were appended on these web applications which were labeled Sign_1, "Sign_2,…,$Sign_n$_31" with a class labeled V and NV that is vulnerable and non-vulnerable and the other with "TYP_1,TYP_2,TYP_3,TYP_4,TYP_5,TPY_6,TYP_7 and TYP_0" that is tautology, illegal/logically incorrect queries, union query, piggy-backed queries, stored procedures, inference, alternate encodings and zero type. The instances were converted to strings of (1, 0). The one indicates vulnerable and the zero represents non-vulnerable.

4.4 Training Set

The dataset for this study was prepared using machine learning approach for data pre-processing. The dataset was clean filtered so as to remove all the noises. It was converted to comma delimited and then Attribute Related File Format (ARFF) format to suite the extension of the experimental tool used. Weka was the tool used for experimentation. For proper validation, 90% of the dataset was used for training and 10% for testing the proposed model.

4.5 Validation of Naïve Bayes Model

In order to produce a robust model, stratified cross validation was chosen with ten (10) folds which used 90% of the dataset for training and 10% for testing in each fold test. Also the dataset was divided into 10 random seeds (1, 2, 3, …, 10) so that the seed can produce a different set of cross-validation folds thereby reducing the variance of the estimation [35].

4.6 Performance Evaluation

The model was evaluated using the following performance measures True Positive rate, False Positive rate, Precision, F-measure and Recall and Accuracy.

$$TP = \frac{TP}{TP + FN} \quad (6)$$

$$TN = \frac{TN}{TN + FP} \quad (7)$$

$$p = \frac{TP}{TP + FP} \quad (8)$$

$$r = \frac{TP}{TP + FP} \quad (9)$$

$$fmeasure = \frac{2pr}{p + r} \quad (10)$$

$$Accuracy = TN = \frac{TP + TN}{TP + TN + FP + FN} = p = \frac{TP + TN}{N} \quad (11)$$

5 EXPERIMENTATION PERFORMANCE EVALUATUION OF THE PROPOSED NAÏVE BAYES MODEL

After the model has been tested, the performance of the model was evaluated using True positive (TP) rate, False Positive (FP) rate, Accuracy (A) Precision (P) rate and Recall (R) rate which are similar to the studies of [7], [9], [10], [24] and [36]. The proposed naïve bayes based pattern recognition model for SQL injection attack was evaluated with a data set of 16,050 instances which constitutes vulnerable and non-vulnerable variables web application. 90% of the dataset was for training and 10% for testing in each fold test.

Figure 3 shows the true positive and false positive rate. Both the true positive and false positive rate is higher at random seed 3 and 10 with 0.988 and 0.008 rates when compare with other seeds.

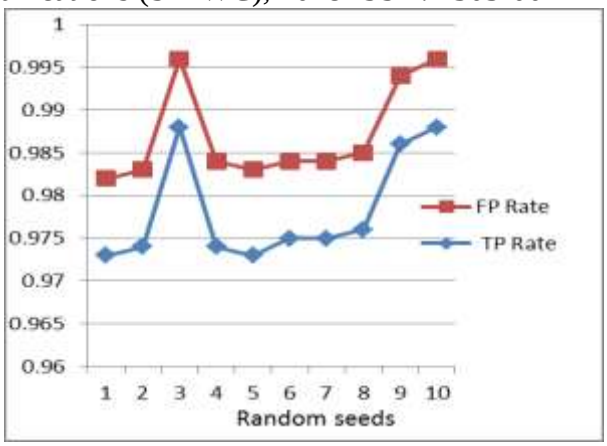

Figure 3: True positive and false positive

Figure 4 below compared the performance the naïve bayes model for detection of SQLIA on three parameters namely: precision, recall and f-measure on the random seeds. the result when compared the other two measures ranging from seed 1 to 10 indicated that a higher precision rate from seed 1 to 9 an at seed 10, the three measures has the same detection rate. Figure 5 below shows the comparison of performance of Precision, Recall and F-measure on the random seeds.

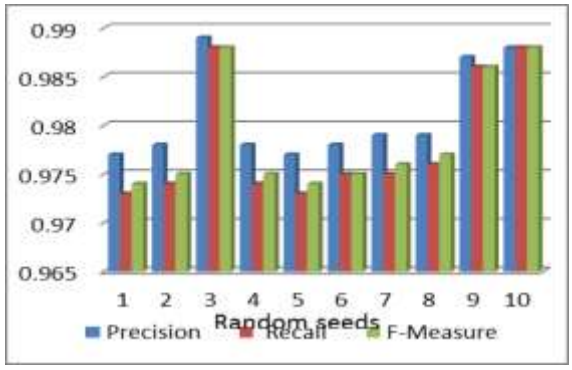

Figure 4: Comparisons of Performances of Precision, Recall and F-measure on the random seeds

The Figure 5 below shows the categorization accuracy of the naïve bayes model on different random seeds. At seed 10 the model performed better when compared with other seeds.

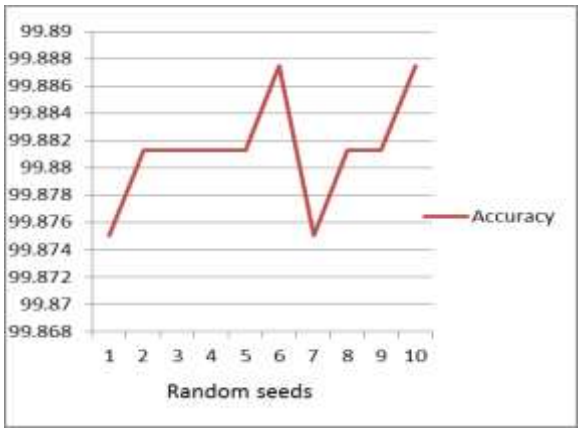

Figure 5: Categorization Accuracy of Naive bayes model using random seeds on SQLIA Dataset

6 COMPARISON OF PERFOMANCE EVALUATION WITH THIS STUDY WITH PREVIOUS STUDIES

Table 2 below shows the comparative analysis in terms of Machine Leaning model (ML) used, Number of attack type (NAT), Detection and Categorization (D & C) while Table 3 shows True positive (TP), False Positive (FP), Precision (P), R ecall (R) and Accuracy (A%) of this study with ot her related study.

Table 2: Comparison of this Study with other Works

Author(s) & year	MLU	NAT	D&C
[7]	RB & NB	3	D
[9]	NN	8	D&C
[10]	QT, FS & SVM	*	D
[24]	SVM	*	D&C
Proposed model	NB	8	D&C

Table 3: Comparison of this Study with other Works

Author(s) & year	TP	FP	P	R	A%
[7]	*	*	1.00	0.890	93
[9]	0.960	0.100	*	*	96
[10]	0.941	0.108	0.949	0.941	94
[24]	*	*	*	*	96
Proposed model	0.980	0.008	0.989	0.988	98

* = not stated

It can be inferred from Table 2 above that the model used in this study is different from that of [24]; in terms of detection and categorization of injection attacks, the authors also detected and categorized SQL injection. The detection accuracy of this study is better when compared with the study of [24]. [7], study is similar to this study in terms of model used. The authors focused on detection of SQL injection and detected only three (3) types of attack while this study detected seven (7) and focused on both detection and categorization. The P rate, R rate and detection A of this study as shown in table 3 is better when compared with the study of [7]. This study outperformed the study of [8] in terms of TP rate, FP rate, detection A with 0.980, 0.008, 98% against 0.960, 0.100 and 96% respectively. Furthermore, the study of [10] used different model for detection and categorization od SQL injection; the model of this study showed better detection A, R rate, P rate, FP rate. Thus, this study has better performance when compared with the existing studies. The Figures 6, 7, and 8 below depicted the TP rate, FP rate and accuracy of the model of this study with other existing studies.

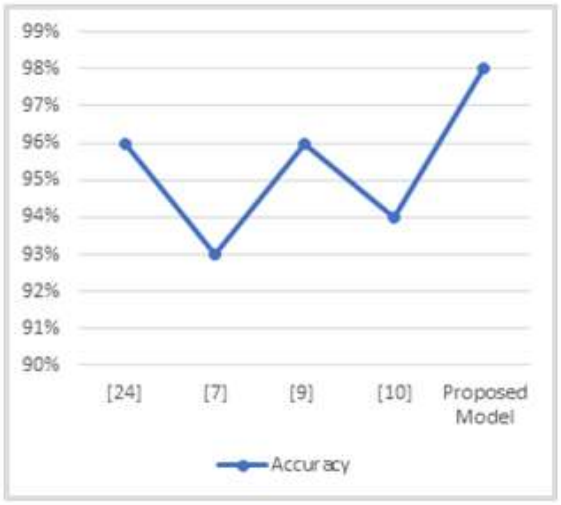

Figure 6: Comparison of Accuracy of this Study and other Related Studies.

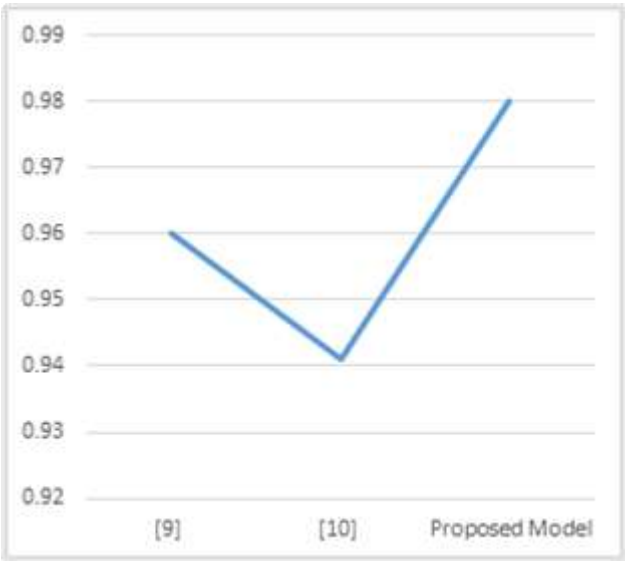

Figure 7: Comparison of the True Positive rates this Study and other Related Studies.

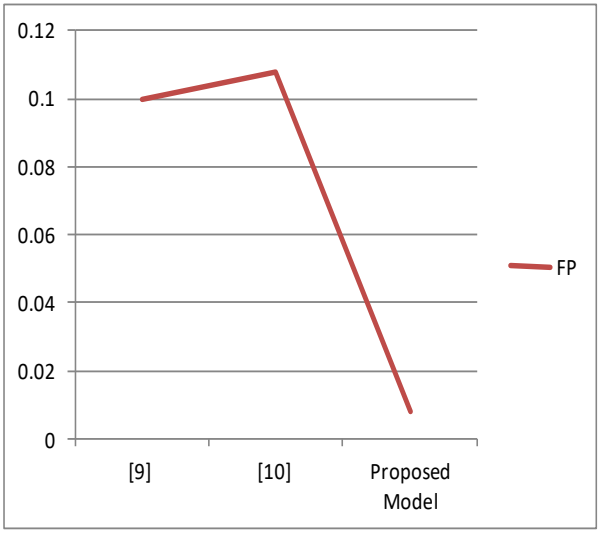

Figure 8: Comparison of the False Positive rates this Study and other Related Studies.

7 CONCLUSION

Web applications have been the platforms that convey numerous activities such as e-payment, e-business and seat reservations across the internet. These application are associated with weakness that cybercriminal take the advantage of, among them are DoS, CSS and SQL injection.

This study proposed a naïve bayes pattern recognition model for detection and categorization of SQL injection attack. 16,050 webpages/URLs were generated from dmoz. 7,000 were assumed to be vulnerable to seven most popular SQL injection attack type and 9,050 were labeled as non-vulnerable accounting to 16,050 instances based on the patterns/signatures. The validation of the model was done using 10 folds stratified cross validation with 1-10 random seeds. The findings showed that the model proposed by this study performed better with 98% and 99% for detection and categorization respectively when compared with the existing studies.

REFERENCES

1. George, T. k., Poulose, J.: A proposed architecture for query anomaly detection and prevention against SQL injection attacks *International Journal of Computer Applications*. 88 (137) 0975. 2016.
2. Diallo, A. K., Al-sakib, K. P.: A survey on SQL injection:vulnerabilities, attacks, and prevention techniques.2011. Retrieved from http://irep.iium.edu.my/769/1/ISCE2011_paper323.pdf and accessed on 10th June, 2017.
3. Dharam, R., Shiva, S. G.: A framework for development of runtime monitors. *International Conference on Computer & Information Science (ICCIS)*, 2(1). 953-957. 2012.
4. Pankaj, S., Rahul,, J., Sarma, S. S.: Combined approach to prevent XSS attacks and SQL injection. 2012 .Retrieved from http://csss.gh.h and accessed on 20th January 2017
5. Avinash, K. S.: Detection & Prevention of SQL Injection Attack in Web Application. 2011 Retrieved from http://avido.yp/bhand accessed on 25th January 2017
6. Kanika. E., Prabhjot, E. k.: A dynamic approach to detect & preventsql injection attack to overcomewebsite vulnerability *International Journal of Innovative Research In Science, Engineering And Technology* 4(12). 2015.
7. Anamika, J., Geetha V.: SQL Injection Detection using Machine Learning. International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT). IEEE. 2014. Retrieved from http://ieeexplore.ieee.org/document/6993127/ accessed on 10th march 2017
8. Varuna, H. Preeti, G.: A Practical Approach of Neural Network Based Detection and Prevention of Malicious URL using MATLAB.*International*

Journal of Advanced Research In Computer And Communication Engineering. 5(7). 2016.

9. Naghmeh M. S.: Employing Neural Networks for the Detection of SQL Injection Attack 2014. Retrieved from https:// www.researchgate.net/publication/264799665 and accessed on 4th April 2017
10. Aniruddh, L., Phalke D. A.: SQL Injection Attack and User Behavior Detection by Using Query Tree, Fisher Score and SVM Classification. *International research journal of engineering and technology (IRJET)* 3 (6). (2016)
11. Pankaj, S., Rahul,, J., Sarma, S. S.: Combined approach to prevent XSS attacks and SQL injection. 2012 .Retrieved from http://csss.gh.h and accessed on 20th January 2017
12. Vandana, D., Himanshu Y., Anurag J.: SQLAS: tool to detect and prevent attacks in PHP web applications *International Journal of Security, Privacy And Trust Management(IJSPTM)*4(1) .2015
13. Kanika, E., Prabhjot, E. k.: A dynamic approach to detect & preventsql injection attack to overcomewebsite vulnerability *international journal of innovative research in science, engineering and technology* 4(12). 2015.
14. Avireddy, S., *et al*.: Random4: an application specific randomized encryption algorithm to prevent sql injection," Trust, Security and Privacy in Computing and Communications (TRUSTCOM), 2012 *IEEE 11th international Conference*. pp.1327,1333, 25-27 June 2012.
15. Tajpour, A., Massrum, M, Heydari. M. Z. : Comparison of SQL injection detection and prevention techniques, *2nd International Conference on Education Technology and Computer* (Icetc). 2012.
16. William, G., Halfond, J., Alessandro, O.: Preventing SQL injection attacks using AMNESIA. 2005 Retrieved from www.ghyrew.com and accessed on 20th January 2017.
17. Mohammad, A., Anurag, J., Neeraj, S.: Web application vulnerabilities monitoring & avoiding techniques, *International Journal of Advanced Research in Computer Science* 4(8), May-June 2013.
18. Raymond, M., Phyllis, F.: Preventing SQL Injection through Automatic Query Sanitization with ASSIST. 2010. Retrieved from http://puitut.hj/of and accessed on 25th January 2017
19. Lambert, N., Lin, K. S.: Use of Query Tokenization to detect and prevent SQL Injection Attacks, in *Proc. 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT)*, pp. 438-440. 2010.
20. Bertino, E., Kamra, A., Early, J. P.: Profiling Database Applications to Detect SQL Injection Attacks, *in Proc. IEEE Internation Conference on Performance, Computing, and Communications* (IPCCC). 449-458. 2007.
21. Sangita, R., Avinash, K. S., Ashok S. S.: Detecting and Defeating SQL Injection Attacks *International Journal of Information and Electronics Engineering*,1 (1). 2011.
22. Nausheen, K.: Detection and Prevention of SQL Injection Attacks by Request Receiver, Analyzer and Test Model. 2011.
23. Cristian, N., *et al*.: CBRid4SQL: A CBR Intrusion Detector for SQL Injection Attacks. 2010. Retrieved from http:// ght.lk/jj/? and accessed on 25th January 2017
24. Romil, R., Shailendra, K. S.: SQL injection attack Detection using SVM. *International Journal of Computer Applications*.42(13). 2012. Retrieved from http://research.ijcaonline.org/volume42/number13/pxc3877043.pdf and accessed on 15th March 2017.
25. A. Ritu & M. Dharmendra. An Approach Based on SVM Classifier to Detect SQL Injection Attack *IJSRSET*,2 (3) 2395-1990. 2016.
26. O. U., Solomon. & J. B. William, (2017). Applied Machine Learning Predictive Analytics to SQL Injection Attack Detection and Prevention. *IEEE*. Retrieved from http://www.napier.ac.uk/~/media/worktribe/output-687590/applied-machine-learning-predictive-analytics.pdf and accessed on 26th April 2017.
27. Krit, K., & Chitsutha. S.: Machine Learning for SQL Injection Prevention on Server Side Scripting Soomlek.2016. Retrieved from http://ieeexplore.ieee.org/document/7859950/?denied and accessed on 15th March 2017
28. Christine, B., Ahmed, E., Saad, D.: Detection of SQL Injection Using a Genetic Fuzzy Classifier System. 2016
29. Hanmanthu, B., Raghu, B. R., Niranjan, P.: SQL Injection Attack Prevention Based on Decision Tree Classification *IEEE Sponsored 9th International Conference on Intelligent Systems and Control (ISCO)* 2015.
30. Abdelhamid, M., Youcef, B., Ahmed, S.: (Improving Web Application Firewalls to detect advanced SQL injection attacks. *Information Assurance and Security (IAS), 2014 10th International Conference*.2014 Retrieved from http://ieeexplore.ieee.org/document/7064617/ and accessed on 20th April 2017

31. Amit, B., Tushar, V.: SQL Injection detection using Baye's Classification. *3rd International Conference on Resent Innovation of Science Engineering and Management.* 2016.
32. Garima, S., Dev, K., Unique, G. & Akhilesh, P. S.: SQL Injection Detection and Correction Using Machine Learning Techniques. *Emerging ICT for Bridging the Future - Proceedings of the 49th Annual Convention of the Computer Society of India (CSI)* 1(1) pp 435-442. 2015 Retrieved from
33. Yi Wang, Zhoujun, L.: SQL Injection Detection with Composite Kernel in Support Vector *Machine International Journal of Security and Its Applications* 6(2). 2012.
34. Olalere, M. Abdullah, M. T., Mahmod, R. & Abdullah A.: Proposed Discriminative Lexical Features for Real-time Detection of Malware Uniform Resource Locator *Indian Journal of Science and Technology, 9(46).* 2016.
35. Ian, H. W., Eibe F.: (2005) data mining: A Practical Machine learning tools and techniques. 2nd edition Morgan kaufmanna Publishers
36. Olalere, M., Abdullah, M. T., Mahmod., R., Abdullah. A.: Identification and Evaluation of Discriminative Lexical Features of Malware URL for Real-Time Classification. *International Conference on Computer & Communication Engineering*. 2016.