SQL Injection Detection and Neutralization using Neural Network

Doni Putra Purbawa, Azzam Jihad Ulhaq, Gusna Ikhsan

Abstract

SQL injection is a common Web attack, and it has always been a challenging network security issue, causing millions of dollars in financial losses worldwide every year, as well as the leakage of a large number of users' private data. This paper proposes a high-precision SQL injection detection method based on machine learning algorithms. We first obtain the real user URL access log data from the Internet service provider (ISP) to ensure that our method is authentic, effective and practical. Then, we conduct statistical parameters research on normal data and SQL injection data.

Introduction

Global internet users reached 4.2 billion in 2018. The International Telecommunications Union (ITU) expects approximately 50% of the world's population to be connected to the Internet by the end of 2018. Similar growth of internet services in harmony. With such a massive increase in the number of Internet users. Many online services have been introduced, including e-commerce, business support, and large data repositories. Many other innovative services are coming soon. As the number of online services increases, security threats increase, and they have become a significant concern for both internet users and online service providers. By having the organization's resources available online and applying mild security measures, unauthorized persons can access confidential and sensitive data. Nowadays, database applications have become the main target of this type of unauthorized access.

Cyber attacks cost the economy an average of almost $ 50 billion a year, more than a fifth of which is caused by SQL injections. According to a study of 300,000 attacks worldwide in a given month, 24.6% were SQL injections. The 2014 Global Threat Intelligence report by NTT Corporation pointed out the cruel fact that average care expenditure following a small-scale SQL attack by a company typically exceeds $ 196,000 (over 1.2 million yuan). SQL injection attacks will not cease in a short time and will continue along with the development cycle of computer technology. It is an immediate and severe, costly threat. Therefore, implementing a strategy to protect against web application attacks, including SQL injection attacks, is a necessary and essential security task, which is crucial to protect enterprise data and user privacy data.

The core of any web application is a relational database supported by statements written in a particular language called Structured Query Language (SQL). Structured Query Language Injection is an injection technique used to attack sites where an attacker inserts SQL characters

or keywords into SQL statements using unconstrained user input parameters to modify the intended query logic. Every time a request is made from the end of the user, a query is applied. The query contains user input that may be malicious. Our web applications need to learn that user input comes from an external source and can be hostile, so we need to process it before it gets done.

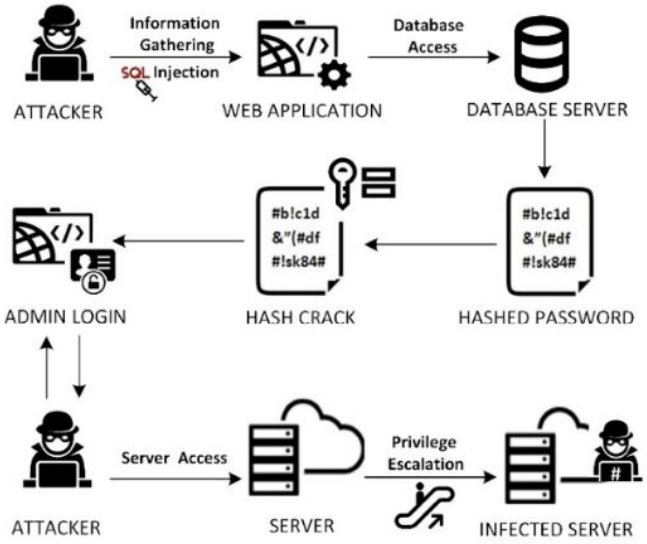

Fig. 1 The SQL injection attacks

Using SQL injection, attackers can alter the SQL statement by replacing the user-supplied data with their data, as shown in Figure 1. In this way, attackers can directly access the database server to retrieve sensitive information. However, the effects of SQL injection attacks are far-reaching and vary by the database application. Some of the consequences of a successful SQL injection attack include authentication bypass, information disclosure, compromised data integrity, limited data availability, and remote command execution. Authentication bypass allows an attacker to gain access to the database application by using a fake username and password. Disclosure of information enables an attacker to obtain sensitive data from the database. Damaged data integrity can help an attacker change some of the data contained in the database. Damaged availability data allows an attacker to delete certain information from the database. Remote command execution enables an attacker to influence the host operating system.

The traditional method of defense by SQL injection uses blacklist filtering, which stores a blacklist and regular expressions to filter on keywords or illegal strings. However, this method is unable to filter out keywords and strings outside the blacklist, making web programs vulnerable to new SQL injection attacks. The article presents a simple and efficient method of SQL injection detection based on machine learning technique, consisting of three parts: data preprocessing, feature extraction and model training.

Related Research

Several studies have been conducted before. In the first study, to prevent SQL injection attacks, machine learning with the heuristic algorithm method was used. Because the traditional algorithms used to detect SQL Injection failed to prevent the attack. There are 616 training data and 23 different classification methods for machine learning. Of the 23 classification methods, the best 5 were taken based on their accuracy and then created a User Application Graphic (GUI). From the heuristic algorithm testing that has been done, it shows that the algorithm is able to detect SQL Injection attacks with accuracy results (93.8%)[1].

The second study revealed that attacks on websites are network problems that cause financial loss and leakage of user privacy data. This problem is still a challenge every year. The method for detecting SQL Injection uses neural networks. The data used comes from the URL access logs of Internet Service Provider (ISP) users. From the data obtained, statistics were performed on normal data and indicated data for SQL injection. From the statistical results the researcher designed 8 types of features and trained them using the MLP model. The accuracy obtained from training the model is more than 99%. These results are compared with other methods such as LSTM for example, the results show that the accuracy of using MLP is superior to the methods that have been compared[2].

The third study discusses how the SQL injection attack also attacks new technologies such as intelligent transportation that integrates advanced sensors, inter-network communication, data processing and automatic control. SQL injection attacks cause huge losses so a method to detect these attacks is needed. The detection result is highly dependent on the accuracy of feature extraction. This research proposes a method of detection of SQL injection attacks based on short-term memory, which can automatically study effective data representation and large and complex data sizes. This study also proposes an injection-based sampling method. This method is formatted to model SQL injection attacks. The results of the study indicate that using machine learning methods (SVM, KNN, Decision Tree, NB, RF) and deep learning methods (CNN, RNN, MLP) can improve the detection accuracy of SQL injection attacks[3].

The fourth research discusses methods to improve web security by detecting SQL Injection attacks. The proposed method is to modify the serve code to minimize vulnerabilities and reduce fraudulent activity. Unlike most approaches, the proposed method is quite simple but shows very effective results. The results obtained are promising with a high level of accuracy for detecting SQL injection attacks[4].

Method

This study aims to classify data into two classes (safe and harmful) through data collected from the Kaggle website. Therefore, we used several classification methods and compared the

results based on their accuracy values. Before implementing our proposed model, we do the preprocessing phase to make sure all data is clear and ready to process.

Tokenization is a method of breaking text into sequential tokens (this is the term or word representation in NLP). The primitive tokenization process usually only breaks the text with whitespace as the divider, then converts it to lowercase so that it is uniform. For example, the text "SELECT * FROM users" will become "SELECT", "*", "FROM", and "users", but problems that arise when there are other queries such as "WHERE id = 1" will become "WHERE", "Id = 1". Therefore we need a method to separate the punctuation marks in a query.

After the data is tokenized, then a method is used to obtain the intensity of the appearance of each word that has been tokenized. In information retrieval (IR) cases, tf * idf is used to give weight to a text document to find out its relevance in a corpus (collection of documents). Term Frequency (tf) is the value of the frequency of tokens appearing in a document and Inverse Document Frequency (idf) is the size of the token spread in the corpus. Then, cosine similarity is a way of measuring the similarity of two vectors of inner product space by using the cosine of the angle between the two vectors. The weighting calculation formula can be seen in Eq. 1.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right) \tag{1}$$

Stopword removal is also required in data processing before it is processed using the classification model. This process eliminates a number of conjunction classes or a large number but does not affect the overall document content. This is usually done to improve system performance so that the system can be used effectively to process content that is really considered important. Words like "from", "which", "at", and "to" are some examples of high frequency words that can be found in almost every document (known as stopwords). Removing this stopword reduces the index size and processing time. Apart from that, it also reduces the noise level.

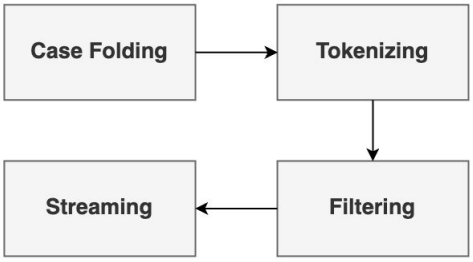

Fig. 2 Text preprocessing flow

In general, the Text Preprocessing stage is the stage where the application selects data to be processed in each document. This preprocessing process includes (1) case folding, (2) tokenizing, (3) filtering, and (4) stemming that can be seen **Fig. 2**.

The Support Vector Machine (SVM) model forms a hyperplane to separate the data based on its categories. In the calculation, the hyperplane has the formula :

$$w^T x+b=0 \quad (2)$$

The formula wT and b is a bias to be searched, and x is the feature used. In determining the hyperplane, sometimes it is necessary to add a dimension called the kernel. This process has several methods which will later become parameters that will regulate its implementation.

The ensemble method is an algorithm as the best solution finder compared to other methods, because this method performs several models that are interconnected with one another to improve accuracy. Random forest is an ensemble method with a decision tree as a basic model. This method combines several decision tree models into one model. The advantage of using a random forest over a decision tree is that it increases accuracy and avoids overfitting. Making the decision tree itself is done based on the calculation of Information Gain (eq. 4) or Gini Impurity (eq. 5).

$$\text{Entropy}(S) = -P_{(+)}\log_2 P_{(+)} - P_{(-)}\log_2 P_{(-)} \quad (3)$$

$$\text{Gain}(S,A) = \text{Entropy}(S) - \Sigma_{v \in \text{Values}(A)} \frac{|Sv|}{|S|} \text{Entropy}(S_v) \quad (4)$$

$$GI = 1 - [(P_{(+)})^2 + (P_{(-)})^2] \quad (5)$$

This calculation method is a parameter set in the random forest modeling. The leaves in the decision tree are a predictable class.

Voting Classifier is one of the ensemble learning methods that combine several models based on voting. The predictions of each model will be considered with the majority voting to produce the final prediction. Prediction results from a model can be favored by adjusting the weight. Other classification methods that were tried were Logistic Regression, Gaussian Naive Bayes, and Decision Tree. But because in the end only focus on the models described previously, these models are not discussed further.

PCA is added before the data is given to the model to reduce dimensions. Grid Search CV is used to find the optimal hyperparameter in the model used. Parameter Grid, which contains a set of parameters to try, a predefined pipeline, and the Stratified K-Fold validation method are used as parameters in this search. The number of components in PCA was tested from 2 to 8 and the model parameters used can be seen in **Table 1**.

Table 1 ***Hyperparameter Tuning***

Model	Parameter
SVM	C : [1, 10, 100, 0.1, 50] gamma : [0.1, 0.01, 0.001, 1.0] kernel: ['linear', 'poly', 'rbf', 'sigmoid']

LDA	solver: ['svd', 'lsqr', 'eigen'] n_components: [i for i in range(1, 9)]
Random Forest	n_estimators: [400,100,200, 70, 60] criterion: ['gini', 'entropy']

Convolutional Neural Network (CNN) is a type of neural network that is commonly used for image data. CNN can be used to detect and recognize objects in an image. CNN is a technique that is inspired by the way mammals - humans produce visual perception like the example above. Broadly speaking, the Convolutional Neural Network (CNN) is not much different from the usual neural networks. CNN consists of neurons that have weight, bias and activation function. The convolutional layer also consists of neurons arranged in such a way as to form a filter with length and height (pixels).

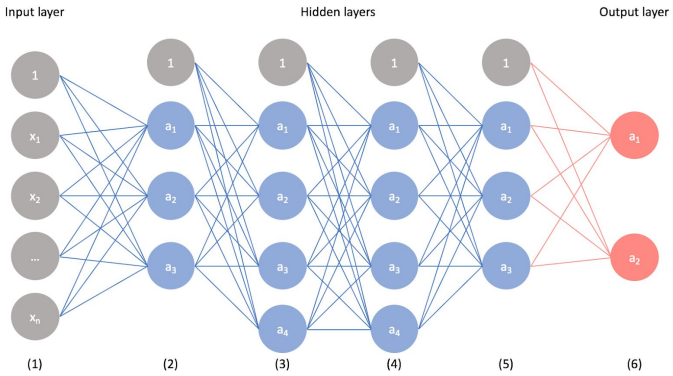

Fig. 3 Convolutional neural network (CNN)

Overall architecture can be seen at **Fig. 4**, hence the next result and analysis part will be explained as the result of our research, analysis and evaluation for our proposed system.

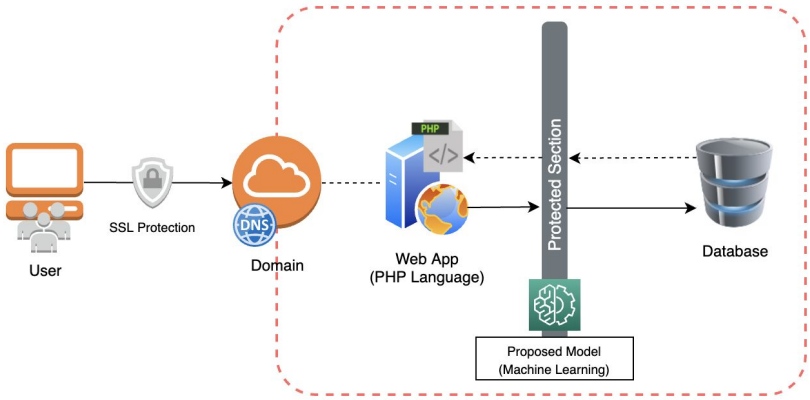

Fig. 4 Overall architecture of our proposed system

Result and Analysis (On Progress)

1. Text Pre-processing

The first preprocessing stage that is carried out is eliminating all null values or blank data in the dataset. Then each sentence is tokenized to taste each word. Furthermore, the CountVectorizer process is carried out to calculate the frequency data that appears on each data, it can be seen in **Fig. 6**.

	Sentence	Label
0	a	1
1	a'	1
2	a' --	1
3	a' or 1 = 1; --	1
4	@	1
5	?	1

Fig. 5 Sample of original dataset

	Sentence	Label	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	a	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	a'	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	a' --	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	a' or 1 = 1; --	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	@	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	?	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Fig. 6 Vectorized SQL dataset using count vectorizer

2. Classification (on progress)
   -
3. Analysis (on progress)
   -
4. Evaluation (on progress)
   -

Conclusion (On Progress)

-

References

[1]. Hasan, M., Balbahaith, Z. and Tarique, M. (2019) “Detection of SQL Injection Attacks: A Machine Learning Approach”, 2019 International Conference on Electrical and Computing Technologies and Applications (ICECTA)

[2]. Tang, P., Qiu, W., Huang, Z., Lian, H., Liu, G.. (2020, Jan) “Detection of SQL injection based on

artificial neural network".
[3]. Qi Li, Fang Wang , Junfeng Wang , Member, IEEE, and Weishi Li. (2019, May) "LSTM-Based SQL Injection Detection Method for Intelligent Transportation System".
[4]. Sarjiyus, O. and El-Yakub, M. B. (2019, May) "Neutralizing SQL Injection Attack on Web Application Using Server Side Code Modification".