Research on the Detection Method of SQL Injection Attack Based on Sequence Alignment

Xiaoxia Qi*, Tianyang Dai†

†College of Information Engineering, Anhui Xinhua University, Hefei 230088, China
*Corresponding author’s e-mail: lushitou710@163.com

Abstract. In order to improve the detection efficiency of SQL injection attack, this paper proposes a detection method of SQL injection attack based on sequence alignment algorithm. Needleman-Wunsch and its improved algorithm are used to carry out sequence alignment and then find out the best alignment results in the global scope. Experiments show that 100% of the SQL injection attacks tested are detected in the detection rate; the algorithm has the minimum response delay compared with other types of SQL injection attacks. The detection method proposed in this paper can improve the detection effect quickly and effectively, and reduce the complexity of time and space.

1. Introduction
With the development of computer technology and Internet, applications based on Internet have also been developed rapidly; security problems have attracted much attention [1]. SQL injection attack [2] is one of the most popular hacker script attacks on the Internet. In the web security threat assessment reports published by OWASP in 2013 and 2017[3-4], SQL injection attack ranked first. SQL injection attack has become the mainstream attack mode, and its harm is far greater than other attack types. SQL statements are flexible and changeable. They can compile, execute and store data in the three-tier application architecture of web through interaction. Because the interaction time of SQL is very short, it is difficult for common detection methods to effectively detect such attacks in a short time. Therefore, this paper proposes a SQL injection detection method based on sequence alignment algorithm [5]. Sequence contrast detection does not need to build syntax parse tree, which reduces the complexity of time and space to a certain extent, and improves the efficiency of SQL injection detection.

2. Related Works
In recent years, the proportion of information security problems caused by SQL injection attacks is increasing. Program developers are aware of the harm brought by the attack, and the research for this is also in-depth. The current research results mainly include: input filtering method, program analysis method, method based on machine learning, sequence alignment method, etc. In terms of user input filtering methods, Zhang et al. [6] proposed a SQL injection attack interception method based on HttpModule; Roy s et al. [7] developed an injection attack detection method based on URL filtering. In terms of program analysis, Qin et al [8] put forward a static analysis method to judge SQL injection attack by analysing the differences in syntax structure and semantic content of SQL statements; Li et al. [9] designed a detection method combining dynamic and static detection, which verified suspected loopholes through fuzzy testing technology and reduced false alarm rate. In the detection method based on machine learning, Li et al. [10] applied the SVM algorithm training into the detection model.
to improve the applicability and detection accuracy of the detection method; Kar D et al. [11] directly applied the bidirectional hidden Markov model to detect SQL injection attacks; Zhang et al. [12] presented an analysis model of SQL injection vulnerability based on artificial neural network.

Although the above research has achieved good results, most of them still have the problems of high rate of failed reporting and error reporting, which cannot achieve real-time dynamic detection. The SQL injection attack detection technology based on sequence alignment proposed in this paper uses Needleman-Wunsch algorithm [13] to analyses the SQL statement structure and judges whether there is an attack behavior, so as to reduce the detection time and improve the detection efficiency.

3. SQL injection attack detection based on sequence alignment

3.1. Needleman-Wunsch Algorithm

Sequence alignment was first used in the field of biological gene sequence analysis. It is of great significance in bioinformatics to analyse the maximum number of base matching between two or more sequences by a specific algorithm. According to the number of sequences, sequence alignment can be divided into pair-wise sequence alignment and multiple sequence alignment. According to the sequence range of alignment, sequence alignment can be divided into global alignment and local alignment. The most classical global comparison algorithm is Needleman-Wunsch algorithm, Smith-Waterman algorithm [14] is the representative of local comparison algorithm.

This paper applied the Needleman-Wunsch pair-wise sequence alignment algorithm. It can carry out sequence comparison analysis in the global-scope and find out the best comparison results, but its time and space complexity are higher than other sequence alignment algorithms, both of which are O(n*m) (where n and m represent the length of two sequences respectively). The main body of the algorithm can be described as follows: Firstly, the data matrix is constructed for two sequences with length of n and m respectively, and the last element of the matrix is the best score of the base comparison of the corresponding positions of the two sequences; then the best ratio pair result is obtained by backtracking.

This paper takes the following code as an example to detect SQL injection attack based on sequence alignment:

- SELECT user,password FROM user WHERE id=''
- Sequence p: S usr,psd F u W id=''
- SELECT user,password FROM user WHERE id='1'
- Sequence q1: S usr,psd F u W id='1'
- SELECT user,password FROM user WHERE id='-1' or 1='1'
- Sequence q2: S usr,psd F u W id='-1' or 1='1'

The specific implementation steps of the algorithm are as follows:

The two sequences are SqA = s1s2…sn and SqB = t1t2…tm. The data matrix S is constructed for two sequences, n and m represent the respective length, i represents the i-th character of the sequence SqA, j represents the j-th character of the sequence SqB (1<=i<=n,1<=j<=m), S(i,j) represents the best score of corresponding positions of the two sequences. The implementation of the algorithm is divided into three steps: initialization, filling and backtracking.

3.1.1. Initialization. The data matrix S(i, j) is constructed for two sequences:

\[
 w(i, j) = \begin{cases} 
 K_1, i = j; \\
 K_2, i \neq j; \\
 \text{gap, } i = - \text{ or } j = -; 
\end{cases} 
\]  

(1)

Among them, \( K_1 \) is the score of sequence matching, \( K_2 \) is the score of mismatch, \( \text{gap} \) is the situation of inserting vacancy (the penalty score of vacancy is 0, that is \( \text{gap} = 0 \)). At the same time, the match
score should be greater than the mismatch score, and the mismatch score should be greater than the vacancy score, so that the result is meaningful.

3.1.2. Filling. The filling process is a recursive calculation of $S(i,j)$. The values of the remaining vacancy in the matrix are optimally selected based on the values of the cells adjacent to them. Fill the cell with the largest value as its final value. The filling rules are as follows:

$$S(i, j) = \max\begin{cases} S(i - 1, j - 1) + w(i, j) \\ S(i - 1, j) + gap \\ S(i, j - 1) + gap \end{cases}$$

The value of $S(i, j)$ can be obtained from three paths, so the value of $S(i, j)$ needs to solve the maximum score of the three paths: From the diagonal direction of $S(i - 1, j - 1)$, there is no penalty point for vacancy, and the matching score is $w(i, j)$; from the vertical direction of column $S(i - 1, j)$, the penalty score for vacancy is $w(i, -)$; from the horizontal direction of row $S(i, j - 1)$, the penalty score for vacancy is $w(-, j)$.

3.1.3. Backtracking. Using backtracking technology to obtain the best comparison result according to the cost matrix. When all the values in the score matrix are calculated, the optimal global alignment result of the sequence is the path from the lower right corner of the score matrix to the upper left corner. As shown in Figure 1.

![Scoring matrix statement](image)

Figure 1. Scoring matrix statement. (a) Without SQL injection and (b) With SQL injection.

It can be seen from Figure 1 that if the last character of the p and q sequences is the same, they backtrack from the bottom right to the top left according to the diagonal principle; otherwise, move one character to the left and compare again until the characters are the same. Suppose that the comparison result obtained by backtracking is sequence T. As follows:

- **q1**: S usr,psd F u W id='1'  
- **q2**: S usr,psd F u W id='1' or 1='1'

- **T1**: S usr,psd F u W id='_'  
- **T2**: S usr,psd F u W id='_'

According to the comparison results, we can see that the user inputs string with length of 1 in sequence q1; the user inputs a number with length of 8 and a string with length of 1 in the sequence q2. In conclusion, sequence q2 is attacked by SQL injection.

3.2. Needleman-Wunsch algorithm optimization

The matrix filling process of Needleman-Wunsch algorithm has two cyclic structures: 1) Judge the value of $w(i, j)$; 2) Calculate the sum of the values of horizontal, vertical and diagonal directions and their respective penalty points, and finally select the maximum value from this three sums as the scoring matrix value. Needleman-Wunsch algorithm consumes a lot of time in the calculation of filling matrix, we optimize the filling matrix to reduce the time complexity.
The improved Needleman-Wunsch algorithm process differently according to the matching situation: When two sequence characters match, we only need to calculate the value $S(i-1, j-1)$ in the diagonal direction, plus the matching value $w(i, j)$, which is the current value $S(i, j)$. In this way, the calculation time can be reduced without calculating the horizontal and vertical values. Compared with the calculation process before the improvement, the time complexity is greatly reduced. The specific steps of the improved filling matrix are as follows:

1. If the $i$-th bit of sequence $p$ matches the $j$-th character of sequence $q$, so
   \[ S(i, j) = S(i-1, j-1) + K_1 \]  

2. If the $i$-th bit of sequence $p$ is mismatched with the $j$-th character of sequence $q$, so
   \[ S(i, j) = \max \left\{ S(i-1, j-1) + K_2, S(i-1, j) + gap, S(i, j-1) + gap \right\} \]  

4. Experiment and Analysis

4.1. Experimental Environment and Data
In order to obtain the user's login account and password, and then log in to the background database to obtain the data information. We used phpstudy, DVWA, and MD5 decryption website to build the experimental environment. The experimental data mainly consists of two parts: finding injection point and confirming injection.

4.1.1. Search for injection point. Enter "1", "1'", "1 and 1 = 1" and "1 and 1 = 2" in the website respectively and submit, the results are shown in Figure 2. It can be seen from the four inputs that there is an injection point, so it is inferred that the SQL query statement is:

Select the column name of First name and the column name of Surname from the table name where id = 'the id we entered'.

![Figure 2. Character test. (a) Input character 1. (b) Input character 1', error reporting. (c) Input character 1 and 1 = 1. (d) Input character 1 and 1 = 2.](image)

4.1.2 Detected injection. First, get the number of fields. Enter "1' order by 1", "1' order by 2" and "1' order by 3" one by one. The results of the first two inputs are correct; when the third was inputted, an error is reported, as shown in Figure 3. Therefore, the number of this field is 2. Then, enter the code "1' union select database(), user () #" to get the current database name. The obtained database name is DVWA, and the user name is root @ localhost, as shown in Figure 4. Next, enter the code "1' union select column_name,2 from information_schema.columns where table_schema= 'dvwa' and table_name= 'users'##", to obtain the data table information from the database, as shown in Figure 5(a).
Finally, select the user and password table to obtain all the user name and password information in the table, as shown in Figure 5(b). After decrypting the obtained password through MD5, you can log in to the background database.

![Figure 3](input_1_order_by_3_error_reporting.png)  ![Figure 4](obtain_database_information.png)  ![Figure 5](a_get_all_information_database_user_table_b_get_user_name_password_all_users.png)

**4.2. Correctness Analysis**

The correctness analysis is based on the following two points:

1) The matching path starts at the bottom right corner. If the user's input is correct, then through the comparison of p and q, it is only possible to add a number of vacant in the p sequence to match q exactly, and the number of consecutive vacant is exactly the number of user's input. Therefore, the path can only move from the lower right corner to the left or along the diagonal to the upper left corner during the comparison.

2) The data type of database can be divided into character type and number type. In the database condition statement, if the field is character type, the separator must be added in the condition judgment (It's up to the application to use single or double quotation marks for separators); if the field is numeric, only a number or a "." can be added in the condition judgment.

When there is a malicious SQL injection attack, the input structure of the SQL statement or the type of user input will change, which makes the final result of the comparison against the above two principles.

This paper defines the ParseTree (p, q, num) function. Where p is the pattern string, q is the target string, and num is the number inputted by the user. This function is called before performing the database operation. If there is no SQL injection attack, function returns true and executes statements, otherwise, it returns false. The experimental statistical results are shown in table 1.

| SQL statement | Number of SQL injection attacks | Number of detected attacks | Average time consuming | Load rate of detection | Accuracy rate |
|---------------|---------------------------------|----------------------------|-----------------------|------------------------|---------------|
| 193           | 107                             | 107                        | 32ms                  | 2%                     | 100%          |
4.3. Performance Testing
In order to further verify the performance advantages of the proposed algorithm, we test and compare it with other types of SQL injection detection methods. During the experiment, the server does not do other work. The response time includes: statement analysis time, returning results after database operation, closing connection, etc. The results are shown in Figure 6. The more the number of SQL injection attacks, the shorter the response time of the algorithm, and the more obvious the advantages. When the number of SQL injections reaches 350, ParseTree has only a response time of 0.04ms, it is more than twice that the response time of the other two methods. Compared with the detection technology based on learning and dynamic detection technology, the advantage of this method is that it does not rely on the accuracy of any learning algorithm, and does not need the support of any external library, reducing the complexity of time and space.

![Figure 6. Performance Comparison of Three Algorithms](image)

5. Conclusion
Sequence alignment algorithm was born in the research of biological sequence, and then applied to the field of computer. Sequence alignment algorithm contains multiple detection directions, which has great plasticity and can be continuously mined and improved. In the future, it can become the main detection method in the field of network security defence. The Needleman-Wunsch pair-wise sequence alignment algorithm proposed in this paper can perform sequence comparison and analysis in the global-scope, and can quickly find out the best alignment results, which has a small response delay for SQL injection attacks. The algorithm is simple and does not need to rely on the support of external database, which reduces the complexity of time and space.

Acknowledgments
This work was financially supported by the project supported by the key natural science research projects of universities and colleges in Anhui province (KJ2016A311, KJ2018A0589).

References
[1] Satter, A., & Hossain, B. M. (2016) Vulnerabilities assessment of emerging web-based services in developing countries. International Journal of Information Engineering and Electronic Business, 8(5), 1.
[2] McWhirter, P.R., et al. (2018) SQL Injection Attack classification through the feature extraction of SQL query strings using a Gap-Weighted String Subsequence Kernel. Journal of information security and applications, 40: 199-216.
[3] OWASP, T. (2013) 10: Ten Most Critical Web Application Security Risks.
[4] OWASP, T. (2017) Top 10-2017 The Ten Most Critical Web Application Security Risks. URL: owasp.org/images/7/72/OWASP_Top_10-2017_%2028en, 29.
[5] Needleman, S. B., & Wunsch, C. D. (1970) A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of molecular biology, 48(3): 443-453.
[6] Zhang, Y., An, J. (2017) SQL injection prevention strategy based on HttpModule. Network Security Technology & Application, 11: 60-61.
[7] Roy, S., Singh, A. K., & Sairam, A. S. (2012) A novel approach to prevent sql injection attack using url filter. International Journal of Innovation, Management and Technology, 3(5):499.
[8] Qin, G., Guo, F., Xu, F., et al. (2013) A static analysis method of anti-SQL injection attack. Computer Engineering & Science, 35(2): 68-73.
[9] Li, X., Zhang, W., Zheng, L. (2018) Vulnerability Detection Using Second-Order SQL Injection Combining Dynamic and Static Analysis. Journal of Huaqiao University(Natural Science) 39(04): 600-605.
[10] Li, H., Zou, J. (2017) Research of SQL Injection Detection Based on SVM and Text Feature Extraction. Netinfo Security, 17(12): 40-46.
[11] Kar D, Agarwal K, Sahoo A K, et al. (2016) Detection of SQL injection attacks using Hidden Markov Model. 2016 IEEE International Conference on Engineering and Technology (ICETECH). IEEE. pp: 1-6.
[12] Zhang, Z. (2016) Research and Implementation Based on Neural Network for SQL Vulnerability Detecting Problem. BeiJing University of Technology.
[13] Sankoff D. (1972) Matching sequences under deletion-insertion constraints. Proceedings of the National Academy Sciences of the United States of America, 69(1):4-6.
[14] Smith T F, Waterman M S. (1981) Identification of common molecular subsequences. Journal of Molecular Biology, 147(1):195-7.