International Journal of Engineering & Technology, 7 (3.6) (2018) 106-109

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

Web Application Vulnerability Detection Using Hybrid String Matching Algorithm

B.J. Santhosh Kumar[1*], Kankanala Pujitha[2]

[1]*Department Of Computer Science, Amrita School Of Arts And Sciences, Amrita Vishwa Vidyapeetham, Mysuru, Karnataka, India.*
[2]*Department Of Computer Science, Amrita School Of Arts And Sciences, Amrita Vishwa Vidyapeetham, Mysuru, Karnataka, India*
E-Mail:Pujithakankanala@Gmail.Com
**Corresponding Author E-Mail:Santhoshbj50@Gmail.Com*

Abstract

Application uses URL as contribution for Web Application Vulnerabilities recognition. if the length of URL is too long then it will consume more time to scan the URL (Ain Zubaidah et.al 2014).Existing system can notice the web pages but not overall web application. This application will test for URL of any length using String matching algorithm. To avoid XSS and CSRF and detect attacks that try to sidestep program upheld arrangements by white list and DOM sandboxing techniques (Elias Athanasopoulos et.al.2012). The web application incorporates a rundown of cryptographic hashes of legitimate (trusted) client side contents. In the event that there is a cryptographic hash for the content in the white list. On the off chance that the hash is discovered the content is viewed as trusted or not trusted. This application makes utilization of SHA-1 for making a message process. The web server stores reliable scripts inside div or span HTML components that are attribute as reliable. DOM sandboxing helps in identifying the script or code. Partitioning Program Symbols into Code and Non-code. This helps to identify any hidden code in trusted tag, which bypass web server. Scanning the website for detecting the injection locations and injecting the mischievous XSS assault vectors in such infusion focuses and check for these assaults in the helpless web application(Shashank Gupta et.al 2015).The proposed application improve the false negative rate.

Keywords: *SHA-1, DOM sandboxing, URL, SQL.*

1. Introduction

SQL Injection is an assault strategy that adventures a security vulnerabilities taking place in the database layer of an application. An attackers use injections to get unauthorized access from data structure, and DBMS. Which is the most commonly used web application vulnerabilities A Database is the most important for many, if not all, web applications and is used to store useful information of the application. SQL Injections occurs when a program writer takes end user's input that is gives openly as a SQL Statement and does not properly validate and filter out dangerous characters. Which allows an aggressor to change SQL reports sent to the database as parameters, allow to not only robber data from
atabase, as well as changing, and erase it. The growing dependency on web applications has made them a regular target for aggressors. Among these client-side code injection in the browser location, is also called as SQL injection**.** SQL injection assaults can yields in a huge collection of areas, from local code to catalogs and web applications. In this paper enlisting new forms of SQL injection and detecting using combination of KMP Matching algorithm and BMP matching algorithm.

2. Related Works

Ain Zubaidah Mohd Saleh et.al,(2015)[1] has described the method for web application vulnerabilities detection by using Boyer-Moore String Matching Algorithm which is used to detect SQL injection attack and Buffer overflow attack. The result of this paper is in terms of low false negative and no false positive with low processing time it has some limitations if the input URL is too long, then it will take more time to scan the URL. Donald Ray, Jay Ligatti (2012)[2] have explained the existing definition of Code injection attacks The flaws also make it possible for benign inputs to be treated as attacks. Web application attacks performed. Different Code-injection attacks on output (CIAOs) methods. Partitioning Program Symbols into Code and Non-code. Elias Athanasopoulos et.al (2012) [3] has proposed detection of XSS attacks that seek to bypass browser-enforced policies. The web application includes a list of cryptographic hashes of valid (trusted) client side scripts. If this method is vulnerable to the node splitting attack, in need to test on client-side part of the proposal in Firefox, safari policies. There is a cryptographic hash for the script in the white list. If the hash is found the script is considered trusted. DOM sandboxing works as follows. The web server places trusted scripts inside div or span HTML elements that are attribute as trusted. Priti Singh et.al [4] has proposed Detection of SQL Injection and XSS Vulnerability in Web Application it consists of some limitations: Detection of SQL Injection and XSS
Vulnerability in Web Application. Designing a tool (filter) for the HTTP request sent by the client. Limited to SQL injection attack but cannot detect which bypass. Nadiya UP and Maya Mathew [5] has proposed Vulnerability detection in Web application which is used to detect the vulnerabilities in web application the author combined taint analysis to detect the vulnerabilities with data mining to predict whether it is false positive or real vulnerability. Here open source static analysis tool has been used to detect the vulnerability. The combination of source code static analysis and data mining did not provide entirely correct

results. Only provides probabilistic result. Prabakar M. A, et.al [6]. An efficient technique to detect and prevent SQL injection using pattern-matching algorithm. Here the initial stage did not produce the false positive and false negative. The pattern matching process took O (n) time to detect the vulnerabilities. Buja.G et.al [7] (2014) has proposed SQL injection attacks scanner using Boyer Moore Pratt matching algorithm. The proposed method able to detect with defined criteria of SQL injection on web application and the proposed work is helpful to the web application developer to give more security to their application from unethical person outside the network. It is helpful to system admin to give secure to the system. Abdalla Wasef Marashdih and Zarul Fitri Zaaba [8] (2016) Has proposed XSS attacks detection approaches in web application, the author used genetic algorithm to detect the XSS attacks, the author succeeded to detect XSS vulnerability using java web application without false positive results. When he implemented in PHP he got many false positive results, because they did not eliminate infeasible pathway from flow graph. Raghuvanshi, K. K., & Dixit, D. B. (2014) [9] proposed prevention and detection techniques of SQL injection. The proposed work has done survey on SQL injection, also described different types of SQL injection detection and prevention techniques.

Anjugum S & Murugun A(2014)[10], has described The attributes and data in the input query are encrypted using AES (Advanced Encryption Standard) algorithm which is fast, and requires little memory.

Once the query is arrived at server side, which is decrypted by using the same key and in turn converts into various token which are stored in to another dynamic table. The performance comparison of cipher text over normal text shows that, cipher text is very difficult and time consuming to crack.

Query tokenization technique converts the input query into various tokens. These tokens are generated by detecting single quote, double dashes and space in an input query. All string before a single quote, before double dashes and before a space constitutes a token.

Tokenization process executes in following four essential steps and then forwarded to the server side. This approach does not require major changes to application code and has negligible effect on performance even at higher load conditions due to its low processing overhead.

3. Existing System

Ain Zubaidah Mohd Saleh a, et.al [1]. Has proposed, a method for web application exposures detection by using Boyer-Moore string matching algorithm. This paper describes the method for web application vulnerabilities recognition by using BMP Algorithm to notice SQL injection attack and Buffer overflow attack.

Existing method used URL as contribution for Web Application exposures recognition. The existing system used two sets of experiments to evaluate the performance of the accurateness (false positive and false negative) and effectiveness (processing time).

The results is low false negative and false positive for all acknowledgment. For SQL Injection, many number of injections is used for the different webpages, web pages tested will effect on the processing time, where the higher number of dissimilar facts of injection, the handling time will become high. For Buffer Overflow, the length of URL will be the main factor of increasing of handling time.

For XSS and CSRF, the dimensions and difficulty of the URL also be the main factor the handling time will be high. [1] Result is 80% accurateness and 20% false negatives effectiveness is reduced to 10%.

4. Proposed Methodology

In Proposed application uses URL as contribution for the Web Application.

Two Algorithms we are using to detect the SQL injection. Compared two algorithms result.

The time complexity of the algorithms will vary but output will be same, finally combined both the algorithms to get efficiency.

Knuth morris pratt matching algorithm

The main part of the KMP algorithm calculates the array F, which is also called as the prefix function. If calculation of F or the prefix function can be done efficiently, the KMP algorithm finds prefix function in O (length of the string) time.

Boyer Moore Pattern Matching Algorithm: The BMP algorithm uses information, which is collected in the pre- process advance to skip pieces of the text, bringing about a lower steady factor than numerous other pattern search algorithms.

Generally, the algorithm works speedier when the example length increments. The key highlights of the algorithm is to coordinate on the end of the example as opposed to the beginning piece of the example, and it will skip various characters instead of looking through each character in the content.

KMP algorithm works from left to write as well as BMP algorithm works from right to left. Both the patterns will work at a time, where the pattern will be matched with the given string then the pattern will be found.

In the proposed application will give a username and password, it will generates a SQL query.

Then the pattern matching algorithm applies on that query, if the query is matched with that pattern it will shows the alarm, the alarm sends a message to administrator that there is a SQL injection on this Web application. If the query is not matched with the pattern matching algorithm then it accepts to login web application.

Proposed Algorithm Step1: Create a table dKMP. Step2: Create a table dBMP

Step3: Defining primary value of index E, which is parallel to the position of pattern with respect to the string.

Step4: Describing primary values of indices jKMP and jBMP, which shows the starting and ending of the pattern respectively.

Step5: Equating the characters of pattern with index jKMP and equivalent string character, equating the characters of the pattern with index jBMP and the equivalent string character. If atleast one observation ends with a mismatch then it will go to step 11.

Step6: If jBMP is greater than jKMP then go to step9.

Step7: As output will get a message to inform that, the pattern is matches with a part of string.

Step8: Go to step15.

Step9: Increasing index jKMP by one, decreasing index jBMP by one

Step10: Go to Step5.

Step11: Choosing the larger shift from,

JKMP-dKMP [jKMP] and dBMP [E+patternlength-jBMP].

Step12: Shifting the pattern to the right relative to the sting increasing the value of index E by the shift defined in step11.

Step13: If the length of the string is greater than the sum of index E and the length of the pattern, then go to step4.

Step14: As output will get a message to inform that the pattern is not found.

Step15: The process will end here.

Proposed queries

- SELECT * FROM Login WHERE
  Username='1' OR '1'='1' password='1' OR '1'='1'

- SELECT * FROM student WHRER username=username
- UNION SELECT password FROM login WHERE username=’username’
- SELECT * FROM login WHERE column LIKE %M%F%.

Proposed flow chart

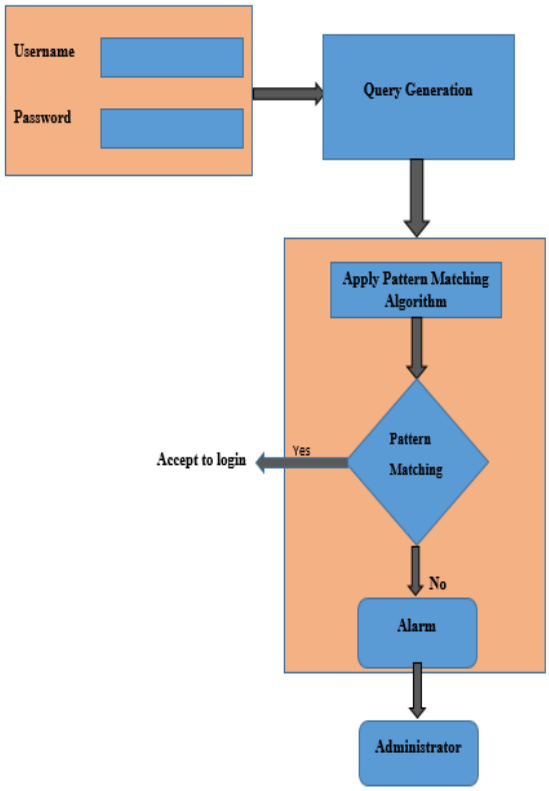


Fig. 1: Proposed flow chart

5. Result and Discussion

Web application for user login has implemented using java NetBeans and MySQL database to store user information. The proposed work detects SQL injection using hybrid string matching algorithm. For hybrid string algorithm. We have used KMP and BMP string matching algorithms. Above the author has been included the algorithm. The proposed work reduced the false negative rate.

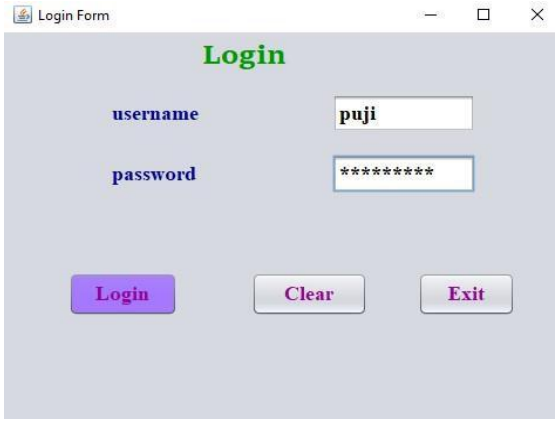


Fig. 2: Without SQL injection

The above Fig-2 shows login without SQL injection, that means if we give correct username and password, which is stored in user database, then only the user, can get access of the application.

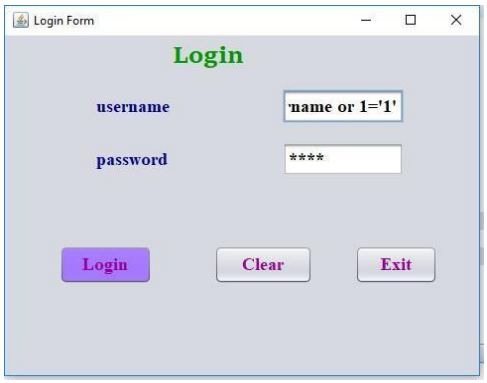


Fig. 3: With SQL injection

SELECT * FROM Users WHERE Username='1' OR '1' = '1' AND Password='1' OR '1' = '1';

The above Fig-3 shows login an application with SQL injection, that means, by using the above query the hacker can get access to the web application without giving correct user name and password.

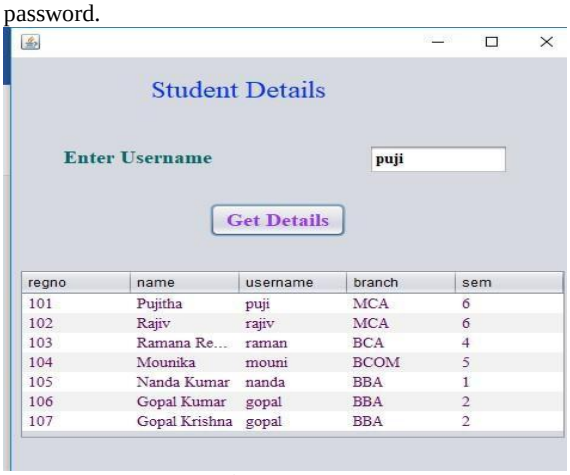


Fig. 4: Fetching the data from the database using SQL injection

SELECT * FROM student WHERE username=username;

The above Fig-4 shows fetching the data from the database using SQL injection, the author used above given SQL query to fetch the data from the database using unauthorized access.

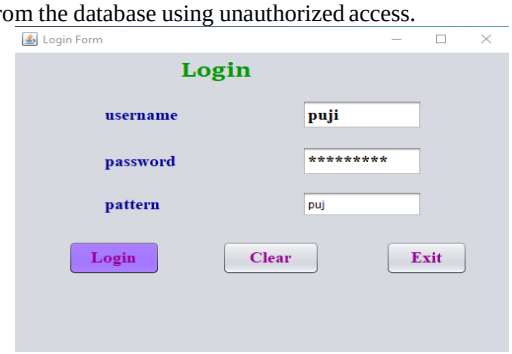
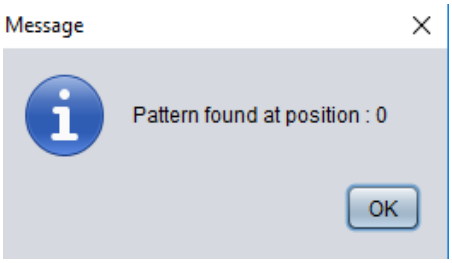


Fig. 5: Using hybrid string matching algorithm



The above Fig-5 shows if the pattern is matched with the user name then it will show the message where the pattern has found, and it will allow to access the login page. If the pattern is not matched with user name it will show a message to provide correct

login details. This application used large amount of dataset to match with pattern.

6. Conclusion

In the proposed work, the author proposed SQL injection detection on web applications. The author successfully detected SQL injection on web applications using hybrid string matching algorithm. This application used KMP pattern matching algorithm and BMP algorithm. The application compared both the algorithms after that the proposed work combined both the algorithms to get efficient result. However, it takes more time to scan the input. In addition, proposed work improved false negative rate.

References

[1] Saleh AZM, Rozali NA, Buja AG, Jalil KA, Ali FHM & Rahman TFA, "A method for web application vulnerabilities detection by using boyer-moore string matching algorithm", *Procedia Computer Science*, Vol.72, (2015), pp.112-121.

[2] Ray D & Ligatti J, "Defining code-injection attacks", *ACM SIGPLAN Notices*, Vol.47, No.1, (2012), pp.179-190.

[3] Athanasopoulos E, Pappas V & Markatos EP, "Code-injection attacks in browsers supporting policies", *Proceedings of the 2nd Workshop on Web 2.0 Security & Privacy (W2SP)*, (2009).

[4] Singh P, Thevar K, Shetty P & Shaikh B, "Detection of SQL Injection and XSS Vulnerability in Web Application", *Prevent*, Vol.1, No.4, (2013).

[5] Manojkumar R, "Vulnerability Detection Behind Web Applications", *Software Engineering and Technology*, Vol.7, No.7, (2015), pp.191-193.

[6] Prabakar MA, Karthikeyan M & Marimuthu K, "An efficient technique for preventing SQL injection attack using pattern matching algorithm", *International Conference on Emerging Trends in Computing, Communication and Nanotechnology (ICE-CCN)*, (2013), pp.503-506.

[7] Rahman TFA, Buja AG, Abd K & Ali FM, "SQL Injection Attack Scanner Using Boyer- Moore String Matching Algorithm", *JCP*, Vol.12, No.2, (2017), pp.183-189.

[8] Marashdih AW & Zaaba ZF, "Cross Site Scripting: Detection Approaches in Web Application", *International Journal of Advanced Computer Science and Applications*, Vol.7, No.10, (2016).

[9] Raghuvanshi KK & Dixit, DB, "Prevention and Detection Techniques for SQL Injection Attacks. *International Journal of Computer Trends and Technology (IJCTT)*, Vol.12, (2014).

[10] Anjugam S & Murugan A, "Efficient method for preventing SQL injection attacks on web applications using encryption and tokenization", *International Journal*, Vol.4, No.4, (2014).

[11] Pushpa BR, "Enhancing Data Security by Adapting Network Security and Cryptographic Paradigms", *International Journal of Computer Science and Information Technologies*, Vol.5, (2014), pp.1319–1321.

[12] Joseph S & Jevitha, KP, "Evaluating the Effectiveness of Conventional Fixes for SQL Injection Vulnerability", *Proceedings of 3rd International Conference on Advanced Computing, Networking and Informatics*, Vol.2,

[13] Shiva Kumar KM, Shruthi K & Shruthi V, "Secured data aggregation in wireless sensor network", *International Journal of Applied Engineering Research*, Vol.10, (2015), pp.26761-26768.