International Journal of Advances in Computer Networks and Its Security– IJCNS
Volume 5: Issue 2 [ISSN : 2250-3757]
Publication Date : 30 October, 2015

Web Application Vulnerabilities Detection Model:

An Approach for Decreasing the Number of Web Application Attack

[Ain Zubaidah Binti Mohd Saleh, Nur Amizah Binti Rozali, Alya Geogiana Buja, Dr. Kamarularifin Bin Abd. Jalil, Dr. Fakariah Binti Haji Mohd Ali and Teh Faradilla Abdul Rahman]

***Abstract*—Apparently most of the web application exists up to today has some vulnerability that can be exploited by unauthorized person. Some of well-known web application vulnerabilities are Structured Query Language (SQL) Injection, Cross-Site Scripting (XSS) and Buffer Overflow. By compromising with these web application vulnerabilities, the system cracker can gain information about the user and lead to the reputation of the respective organization. Usually development of a web application does not realize that their web application has vulnerabilities. They only realize it when there is an attack or manipulation of their code by someone. Even though SQL Injection is very easy to protect against, there are still large numbers of the system on the internet are vulnerable to this type of attack because there will be a few subtle conditions that can go undetected. Therefore, this paper proposes a detection model for detecting and recognizing the web vulnerabilities.**

Keywords—Web application vulnerabilities; SQL Injection; Cross Site Scripting; Buffer Overflow; Pattern Recognition

I. Introduction

Today, the internet has become part of our daily life. The internet has used almost for every task, including learning, work and business. As the internet becomes more and more important, there are some people known as hackers that can disrupt the peace of using internet. Therefore, network security is needed to protect user's browsing activities from any kind of web vulnerabilities. Network security is a term used to define a field that focuses on securing networks from any attack or exploit especially by hackers. To secure the network, it is usually handled by network administrators in each organization that applies security policy. The main reason why network security is implemented is to protect the network and all its resources from being accessed through the network by unauthorized as well as to make sure the three main goals of security; integrity, availability and confidentiality is safe.

The web or the term World Wide Web (WWW) is one of the ways of accessing information over the internet. The Web uses the Hypertext Transfer Protocol (HTTP) protocol to transmit data through the internet as the HTTP is a language is known over the internet. HTTP protocol is used by web services in order to let the applications communicate with each other for sharing of business logic. By using a web browser, web documents called web pages can be accessed. The web pages may contain text, videos, sound, images and some multimedia components. Web pages are linked to each other via the hyperlinks. One of the important components of the web is the URL (Uniform Resource Locator). Every website on the web has a unique URL making the process of searching for a specific site a lot easier. URL standardized naming convention for addressing documents available over the Internet or Intranet.

Since the web has many pages and links, it is possible for it to have some loop holes that can cause information leakage. These loop holes are called vulnerabilities which are the weakness of a web that give an attacker advantage to exploit it and try to gain unauthorized access to his target. There are lots of vulnerabilities that can be exploited, but three of the most common web application vulnerabilities that exist in a web application are Structured Query Language (SQL) Injection, Cross-Site Scripting (XSS), and Buffer Overflow. SQL Injection is an attack in which the attacker inserts SQL commands into a form or parameter values [1]. It exploits the use of SQL query in the application. Cross-site scripting is an attack that exploits the use of JavaScript in a web. JavaScript is code that is downloaded into the user's browser. Using JavaScript, the confidential information can be gained as JavaScript has the ability to access confidential information [2]. On the other hand, the buffer overflow is an exploit that can make the memory allocated to a certain application become massive [3]. For example, an application expecting a five-digit postcode, therefore the programmer only allocates enough memory for the perimeter. If an attacker enters more than five digits for example, hundreds of digits, the application will end up using more memory than what it should.

Currently, there are some web application vulnerabilities scanners exist on the internet. Some are free source and some are costly in order to use it. However, there are some problems encountered in these scanners. Most of them encounter false negatives and false positive. False negatives is the scanner does not find any vulnerabilities in a website and telling user that the web is safe while actually the web application have some vulnerabilities, whereas false positive is when the scanner telling the user that vulnerabilities do exist when actually there is none.

Therefore, this paper discusses on the proposed model for recognizing the web vulnerabilities by searching for the suspicious and defined web vulnerability criteria. This work can help the web application administrator to take a look and always standby in secure mode to avoid and secure mode for avoiding any attacks from the attacker.

II. Literature Review

A. *Computer Security and Web Application*

Computer security is a way to prevent or protect computer against unauthorized person that intent to access, destruct or alter information. Computer security is divided into several types; some of them are hardware security, software security, information security and other [3]. Information security is the protection of information in order to achieve three goals of security which are availability, confidentiality and integrity [4]. Usually, the information security today will involve the use of the internet. Hence, information security of the web is highly recommended to be observed and controlled accordingly.

The web application can be described as an output of the developed script language such as Active Server Page (ASP) and Java Server Page (JSP) [1]. According to [5], the most widely used languages are Java, .Net and PHP, however many organization uses ASP in their web system. Depends on its purpose, this application can process and store the data into the back end database [6]. It is made of several components that can be compromised like URL and the user input from which considered as the weakness or the vulnerabilities of the web application. The flaws or weakness of a web application can be used by the attacker or a hacker to exploit the web application. The proliferation of web application has increased the number of information being stored online; as a result the attack on web application also keeps on growing [6].

B. Web Application Vulnerabilities

There are many types of web vulnerabilities. Some of them are SQL Injection, Cross-Site Scripting (XSS), Cross Site and Buffer Overflow.

SQL is a programming language that was created for the purpose of managing the data stored in a relational database. Managing of data, including adding, deleting, editing and retrieving all the information stored in a database. For web application, SQL is the only way for users to communicate with the database. Some examples of a relational database are Oracle, Microsoft Access and MySQL [7]. One of serious attack based on SQL is SQL Injection. It allows hackers to gain unrestricted access to the databases underlying in the web application. As these databases may contain sensitive user information, the causing security desecration can include many problems such as fraud and identity theft [8]. In some cases, attackers can even use an SQL Injection vulnerability to take control of and corrupt the system that hosts the Web application.

Cross Site Scripting (XSS) is a well-known and one of the common web application vulnerability being used by hackers or attacker today [9]. Other than that, some of another method that are listed like Information Disclosures, Content Spoofing and Stolen Credentials most likely the after effects of XSS attacks. XSS can be categorized into three types which are: Stored, Reflected and DOM Based Attack [10]. In Stored XSS, the attacker usually stored the malicious script in different kind of ways, such as message forum, comment field and other trapping ways they have been decorated by the attackers. While in Reflected XSS, the attack happens when a user click an unknown link such in an email message or in web server. After user click the link, the injected code will reflected off the web server with some error message and search result. Differ from other two types of XSS attack, DOM (Document Object Model) Based Attack is kind of XSS attack that performed by altering the DOM environment on the client side rather than sending the malicious code to the server.

The buffer overflow is an exploit that can make the memory allocated to a certain application become massive [3]. For example, an application expecting a five-digit postcode, therefore the programmer only allocates enough memory for the perimeter. If an attacker enters more than five digits for example, hundreds of digits, the application will end up using more memory than what it should.

Meanwhile, cookies are used to tell the system or application who is the user that currently using the system. For example, when a user login to the web application a cookie will be assigned. After the cookie has been assigned, the application will recognize the user by using the cookie [13]. Cookie poisoning is an exploit that involve the cookie to get confidential information. Usually this type of attack used for identity theft. It is said that this type of attack is easy to avoid by installing the firewall.

Based on the report by IBM in 2012, it stated that XSS and SQL Injection are growing rapidly throughout the year. On the other hand, [5] reported that XSS had become the number one vulnerability to most of programming language on the web; the highest number of XSS found was in Pearl language, followed by Java and PHP while SQL injection vulnerability was found mostly in ColdFusion, followed by ASP.NET. Among the web vulnerabilities discussed, Buffer Overflow has the lowest percentage found on the web applications. However, it was stated that many vulnerabilities found were not affected by language choice.

C. Exact String Matching

In order to gain the process of detecting the web application vulnerabilities, a few exact string matching algorithm has been studied. Brute Force string matching algorithm also known as a Naïve string matching algorithm is a very straightforward approach that based on the definitions and the problem statements [14]. According to [15], Knuth-Morris-Pratt is another example of string matching algorithms. The idea of this algorithm basically is the same as the brute force algorithm, but it is an enhanced version as it can shorten the time. This is because using a preprocessing phase, the window shifted will not necessarily buy one. Same as Brute force string matching algorithm compares from left to right. Boyer-Moore string matching is one of the fastest string search algorithm as it search the string for the pattern but not each of character in the string. As the pattern length increase, the algorithm will run faster. This algorithm usually used to search for virus patterns, search database and do other tasks that need for searching large amount of data in a short period of time [14].

D. Related Works

There are some related works that have been done by some other researchers out there regarding the web application vulnerability scanner. Some of them just focus on a certain type of web application vulnerability, but some able to detect many types of vulnerability in one project. Simply refer to [16] a tool for detecting SQL Injection called the SQL Rand has been developed. This approach basically based on instruction-set randomization. It provides a framework that instead of using normal SQL keywords; the developer can produce queries using randomized instruction. However, as it applies a secret key to alter instructions, security of this approach is dependent on attackers not being able to find out the key.

Other research is Secubat. It is a web vulnerability scanner that can detect SQL Injection and XSS. This scanner that uses a black-box approach to crawl and scan web sites for the attendance of exploitable SQL Injection and XSS vulnerabilities contains three main components which are: Crawling Component which gathers a set of target websites,

Attack Component launches the configured attacks against these targets and Analysis Component examines the results returned by the web applications to determine whether an attack was successful [17].

Johns et al. [18] proposed a Server-Side Detection of Cross-site Scripting Attacks (XSSDS) that use passive detection system to identify successful XSS attacks. It depends on the HTTP traffic incoming parameter and reflected XSS issue. The proposed detector relies upon a direct comparison of incoming HTTP-parameters and outgoing HTML. Stored XSS is therefore not always detectable with this approach: the required direct relationship between HTTP request and response does not necessarily exist. It might be possible to detect the initial exploiting request/response pair, if the given stored XSS takes effect immediately. However, in certain cases, the HTTP request that injects the malicious payload permanently in the application and the poisoned HTML response are not created successively.

III. Methodology

There are six phases involved, which are information gathering, data collection, selection of algorithm, identification of the process, construction of an algorithm and evaluation.

The first phase is information gathering in order to come out with the research requirements. During the first phase, issues and areas in the current trend of information security will be studied thoroughly. The studies will include some explanation about web, web vulnerabilities and algorithms that are currently being utilized for the string searching. Once these requirements are identified, the next phase is data collection. It is conducted to identify and define the criteria of web vulnerability. Experts' views and opinions are extremely required during this phase.

The third phase is to determine the algorithm requirement for the method and to select the suitable detection technique. Based on the requirements, a selection of algorithm that most suitable to be used will be conducted. As a result, Boyer-Moore is selected as the most suitable algorithm to be used since it can search for a pattern in a short period of time. The fourth phase specifically focuses on how to perform the process of detecting the web application vulnerabilities. During this phase, the algorithm for detecting the web application vulnerabilities will be constructed. Then the algorithm was designed and developed.

The next phase is constructing the algorithm for detecting the web application vulnerabilities. Fig. 1 shows the development of algorithms. Initially the algorithm process was determined based on the vulnerabilities, and while final phase is evaluating the method. Some experiments should be conducted in this phase. The performance of the proposed detection model will be evaluated by:

1) Measuring the efficiency of the proposed detection model by measuring the time taken for it to complete the scan
2) Measuring the accuracy of the proposed detection model by comparing the total vulnerabilities detected in the target to the total vulnerabilities exist.

IV. The Proposed Model

This section discusses on the proposed model for detecting the web application vulnerabilities. The proposed model composes of three main detection modules. Each module will execute the process by employing the Boyer Moore string matching algorithm.

A. *The Overview of The Proposed Model*

Upon starting the proposed detection model, several modules for each of the vulnerabilities will be loaded. The module will scan for the attribute or criteria of the web application vulnerabilities. Fig. 2 shows the flow chart of how the proposed model will work.

The first process is initiated by the user by entering the input string or files that will be used to execute the detection module. The string can be the URL of the site or web application while the file can be the stream of source code for each web application. The server of the target will be scanned in order to get the information needed. Next step is the launching of the modules and perform each of their functions. The first detection module launch will be the SQL Injection Detection Module, followed by the XSS Detection Module and finally with the Buffer Overflow Detection Module. Lastly, a report will be generated based on the finding of the scan. The respective web application can be detected as not vulnerable at all, or vulnerable to SQL injection attack or XSS attack or Buffer Overflow or vulnerable to all types web application vulnerabilities.

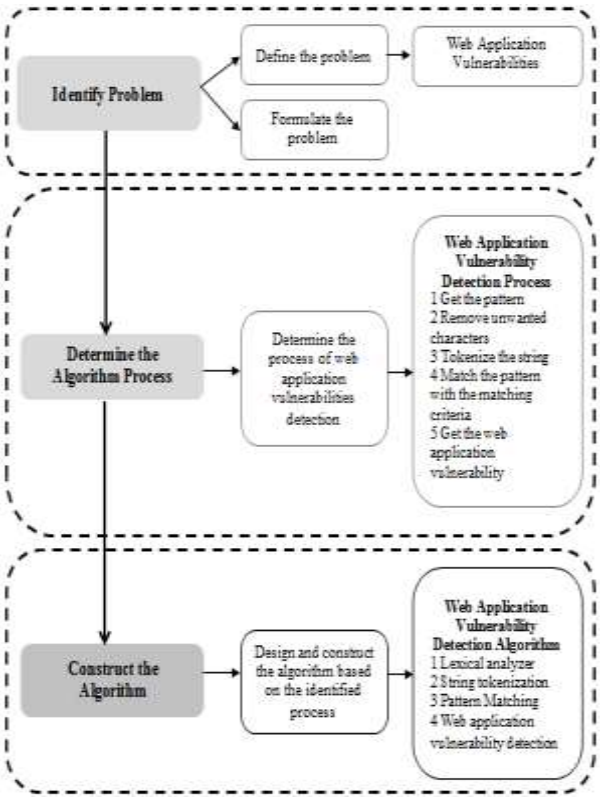

Figure 1: Web Vulnerabilities Detection Model Algorithm Development

Based on the chosen string matching algorithm, each of the string or files will be scanned for the defined attributes of each web application vulnerability. The number of matching attribute will be calculated for the reporting purposes. The larger number of percentage, the more vulnerable the web application.

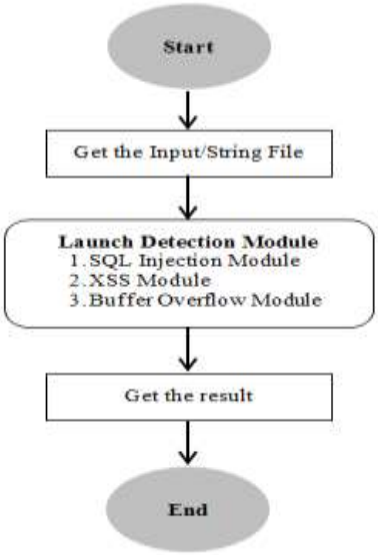

Figure 2: Proposed Model

B. *Input, Scan, Match and Report*

Fig. 3 shows the process of inserting the input string, scanning, matching and reporting. In the first place, the input string of the source file will be scanned character by character. The employment of the Boyer Moore string matching algorithm will allow the model to scan character by character starting from left to the right. The complete scanned word will be matched with the defined attributes of each web vulnerabilities.

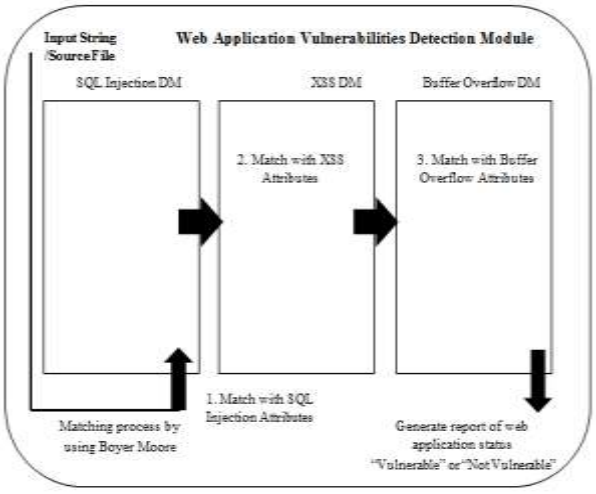

Figure 3: The Process of Web Application Vulnerabilities Detection by Employing Boyer Moore String Matching Algorithm

This is called pattern recognition. The matching will be started with SQL Injection attributes followed by XSS and Buffer Overflow attributes [18]. If there is a match, the web application is considered vulnerable and the report will be generated from the use of web administrator to take further action. The process of the detection will be optimized.

V. Conclusion

The increasing use of the web has led to the increasing of cybercrime in the internet world. User's confidential details are easily exposed to anonymous who always use them in the wrong way. Many methods were used to overcome the problem, but produce a false result. Therefore, a few models were studied and finally one model was chosen. The proposed model uses Boyer Moore string matching algorithm with the hope it will be able to detect web application vulnerabilities such as SQL injection, XSS and Buffer Overflow.

A few experiments will be conducted in the future to test the proposed model in detecting web application vulnerabilities. This proposed model is expected to be able to detect the vulnerabilities and generate a report for reference and further action by the web administrator.

Acknowledgment

The authors would like to thank Universiti Teknologi MARA for supporting this work. This work was supported by Universiti Teknologi MARA, Malaysia and Ministry of Higher Education of Malaysia under Research Acculturation Grant (RAGS) 2013.

References

[1] Dr. E. Benoist, Advanced Web Technology 9) OWASP Top 10 Vulnerabilities & Cross Site Scripting, 2012

[2] Visa Bulletin, Top Three E-commerce Vulnerabilities and Acquirer Actions to Ensure Their Merchants Protect Online Data, 2012

[3] B. Pawar Pankaj, M. Nagle and K. Kawadkar Pankaj. 2012. Prevention of Buffer overflow Attack Blocker Using IDS, International Journal of Computer Science and Network (IJCSN) Volume 1, Issue 5, October 2012 www.ijcsn.org ISSN 2277-5420.

[4] Stallings, W. Network Security Essentials Applications And Standards, 4th Edition, Pearson, ISBN 13: 978-0-13-706792-3, 2011.

[5] Whitehat Security Report. 2014 Website Security Statistics Report. http://info.whitehatsec.com/rs/whitehatsecurity/images/statsreport2014-20140410.pdf (access on 17 November 2014)

[6] TrendLabs, Web Application Vulnerabilities HOW'S YOUR BUSINESS ON THE WEB?, 2012.

[7] R.P. Mahapatra and S. Khan, A Survey Of Sql Injection Countermeasures, International Journal of Computer Science & Engineering Survey (IJCSES) Vol.3, No.3, June 2012.

[8] G.J. Halfond William, J. Viegas and A. Orso. 2006. A classification of SQL-injection attacks and countermeasure, In Proceedings of the IEEE International Symposium on Secure Software Engineering, Arlington, VA, USA, pp: 13-15.

[9] J. Manico, Future of XSS Defense, 2012.

[10] A. Kumar Baranwal. 2012. Approaches to detect SQL Injection and XSS in web applications. *EECE 571B, Term Survey Paper, Canada*, pp: 1-12.

[11] J.Burns,Cross Site Request Forgery An introduction to a com mon web application weakness, Information Security Partners LLC, 2007.

[12] F5 Networks, Inc, Web Application Vulnerabilities and Avoiding Application Exposure, 2007.

[13] C. Charras and T. Lecroq, Handbook of Exact String-Matching Algoritms.

[14] N. Singla and D. Garg. 2012. String Matching Algorithms and their Applicability in various Applications, International Journal of Soft Computing and Engineering (IJSCE)ISSN: 2231-2307, Volume-I, Issue-6

[15] S.W. Boyd and A.D.Keromytis, SQLrand: Preventing SQL Injection Attacks, 2004.

[16] S. Kals, E. Kirda, C. Kruegel and N.Jovanovic, SecuBat: A Web Vulnerability Scanner, 2006.

[17] M. Johns, B. Engelmann and J. Posegga. 2008. Xssds: Server-side detection of cross-site scripting attacks. In Computer Security Applications Conference, 2008. ACSAC 2008. Annual , Dec. 8-12, IEEE Computer Society, pp: 335-344. DOI: 10.1109/ACSAC.2008.36

[18] Buja, G.; Bin Abd Jalil, K.; Bt Hj Mohd Ali, F.; Rahman, T.F.A., "Detection model for SQL injection attack: An approach for preventing a web application from the SQL injection attack," *Computer Applications and Industrial Electronics (ISCAIE), 2014 IEEE Symposium on* , vol., no., pp.60,64, 7-8 April 2014