http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 13 (5): 934-940, 2014
ISSN 1812-5638 / DOI: 10.3923/itj.2014.934.940

An Efficient Web Vulnerability Scanning Method Based on Template Matching

[1]Xiaohang Liu, [1]Qin Chen, [1]Longlong Li and [2]Shuiming Chi
[1]College of Computer Science, Hangzhou Dianzi University, Hangzhou, 310018, China
[2]Jinhua Biqi Network Technology, Co. Ltd., Jin Hua, 321017, China

Abstract: Web vulnerabilities develop in a dramatic speed ac-cording to the booming of IT technology and have become a serious threat to web security. Meanwhile, most dominant scanning approaches in current research community and commercial systems are low efficiency. In this study we propose a novel approach to improve the efficiency of web vulnerabilities scanner by designing the similarity comparison and template extraction. Therefore the novel approach can obtain the web structure dynamically and avoid capturing the same pages repeatedly. Experiments in this study show that our method achieves efficient detection of Web page vulnerability, meanwhile, it also has a high adaptability and flexibility.

Key words: Web vulnerability, vulnerability scanning, web directory tree structure, template matching, web crawler

INTRODUCTION

Network technology has developed in a unbelievable speed and web applications have been qualified with openness, ease of use and development. A growing number of corporations, government agencies and individuals have established their own sites on the Internet, which, however, prompts the security issues of web application. There are many types of vulnerabilities in web application, each of which can be the target of web attack. SQL injection attack is a main-stream approach of web attacks. SQL injection attackers make use of the absence of data legitimacy judgment of the user input in a web application and may obtain administrator privileges with a carefully constructed SQL statements to insert special characters and commands and attack the back-end database through the input areas of web pages (such as URL, forms, etc.). Therefore, in order to ensure the security of web applications, web vulnerability scanner which is used for detecting and digging out the SQL injection vulnerability has become a essential part of network security.

Using the prominent and commercial scanners, while scanning web vulnerabilities, all of the pages in one site need to be crawled and then to be analyzed and tested. Although, this method can achieve high scanning accuracy but for large sites, that will lead to excessive overhead. Therefore, in order to improve the efficiency of SQL injection vulnerability scanning, reducing the workload of web scanner on the premise of ensuring a certain accuracy of vulnerability scanning has become the current urgent problem.

Through analyzing a lot of web pages from many business sites, we find that web pages has a high similarity of structure in the same directory. Therefore, on premise of ensuring a certain accuracy of vulnerability scanning, just crawling some of the pages in the same directory for testing can reach the goals of improving detection efficiency in web vulnerability scanning. Based on the above research we propose a novel efficient web vulnerability scanning method which randomly crawls a certain number of pages in the same directory of one site. We calculates the similarity of those web pages, if the similarity reaches a certain threshold, we will acknowledge the fact that the directory pages are generated by the same template and stop crawling other pages. The only thing to do is to conduct SQL injection with the crawled pages and to detect web vulnerability by Scanner. Experimental results show that the proposed method ensures the number of injection points detected roughly unchanged but achieves a substantial reduction of the number of the pages which need to be detected and verify the effectiveness of the new method.

RELATED WORK

Researches of Detection and prevention against SQL injection have achieved a lot of research results but most of recently researches of SQL injection vulnerability

Corresponding Author: Xiaohang Liu, College of Computer Science, Hangzhou Dianzi University, Hangzhou, 310018, China

detection are focus on the development of SQL injection tool without studying its essence (Peng and Fan, 2010). Currently, there are many excellent domestic and overseas injection detection tools. For example, NBSI is a veteran ASP (active server page) injection detection tool designed for MSSQL (Microsoft structured query language) Server, which mainly used in MSSQL database injection detection.AppScan is a vulnerability scanning applied to detect security vulnerabilities caused by web design and site building. Moreover, as SQLIA appeared in an earlier time, researches for the detection of such attacks is relatively more extensive and deeper mainly in two aspects: (1) Vulnerability Detection and (2) Attack Defense. In the vulnerability detection, Shin (2006) using the input flow analysis and input validation analysis establishes a white box to test input data so as to find SQL injection vulnerabilities. WAVES (Huang *et al*., 2003) uses the web search to find vulnerabilities in web applications and makes use of pattern table and attack techniques to generate attack code to find SQL injection vulnerability. However, the above methods pay more attention on the effectiveness and integrity of SQL injection vulnerability detecting than the detection efficiency of the scanning software. Inefficient detection software will bring enormous time and hardware cost and this defect will greatly reduce the market competitiveness of web vulnerability scanner. This study presents a vulnerability scanning method based on template matching, which just compensates for the lack of scanning efficiency in the current study.

FRAMEWORK OF RESOLUTION

The purpose of SQL injection vulnerability detection is to use a simulated attack mode which constructs a special SQL statement on the URL address of the targeted web site to proceed injection test and then determines whether there are injection vulnerabilities according to the response content. For example, if we add the the SQL statement and 1 = 1 to URL address and the sever returns a normal page, however or and 1 = 2 make the sever return a page including database error messages or others different from normal pages, it will be judged that there are SQL injection vulnerabilities in the web page. The scanning process of SQL injection vulnerability can be described as: (1) Using web crawler to crawl pages, (2) The structure analysis of web page and look for any possible injection points, (3) Send simulated attack data to the injection point, (4) Determine whether there are SQL injection vulnerabilities in the test pages by analyzing the return data. During the detection of SQL injection vulnerabilities, those approaches which crawl all the pages in the site and look for possible injection point for vulnerability testing will lead to low efficiency of vulnerability scanning. For the low efficiency of current web vulnerability scanner, we proposes an efficient vulnerability scanning approach based on template matching. The approach maintains a directory tree of the current site during the process of crawling pages. Each node of the directory tree is a directory, which can contain subdirectories or pages. when a web page is crawled by the vulnerability scanner, it will be stored into the appropriate directory node according to its URL address. If the number of pages under the directory node reaches a specified amount η, we proceed template matching through calculating the web similarity of the pages under the directory. If the similarity reaches a certain threshold λ, we can determine that the pages under the current directory are generated by one template and we do not have to crawl the other pages under the directory. Finally, we conduct SQL injection testing to the exacted pages. The framework of the program is shown in Fig. 1.

ALGORITHM DESIGN

This solution will be described as Algorithm 1:

Algorithm 1: Scanner(Root) function

Require:
the current root directory, Root.
Ensure:
The result of SQL injection test of the current directory, SQLITest(root)
1: Let catalog root be a tree with a single leaf
2: **for** url in the root **do**
3: crawl the links in current url by crawler(url) and save them into the set web
4: calculate the sampling parameters from the size of web as [λ, η]
5: select η links randomly and saves them into sampling set web'
6: calculate the similarity by simParse(web') and save the result as simValue
7: **if** simValue>λ**then**
8: select a link randomly from web'as url'
9: Scanner(url')
10: **else**
11: Scanner(url)
12: **end if**
13: **end for**
14: **return** SQLITest(root);

Line 1 maintains a directory tree of the current site. Root is the current root directory. Line 2 and 3 begin to traverse each url of the current root directory. Each url, as a node of directory tree, is stored in current directory tree. Line 4 uses the interface crawler() of web crawler to crawl

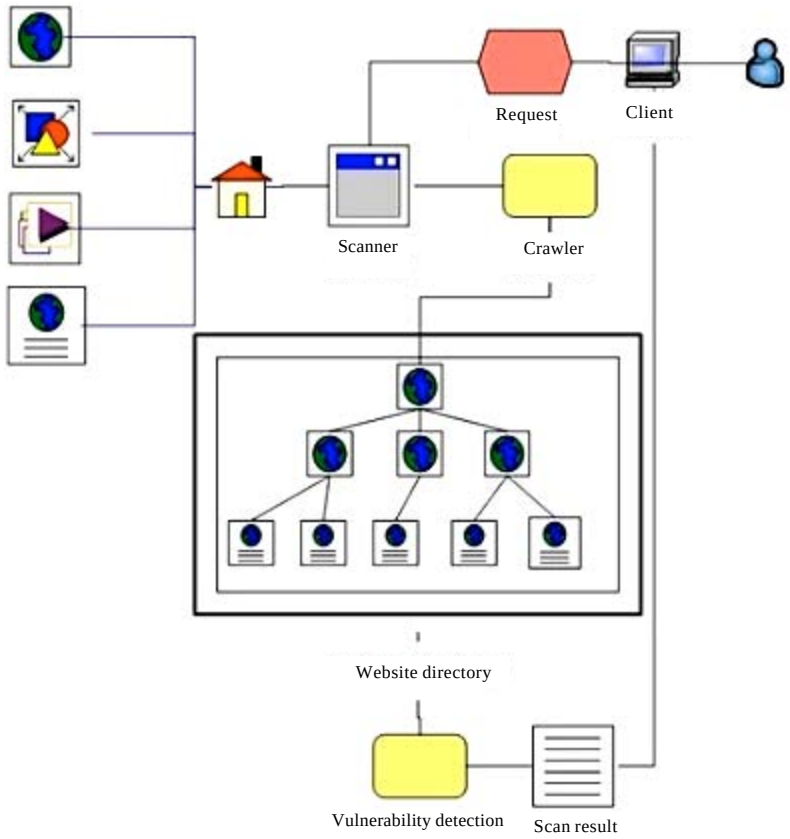

Fig. 1: Framework of the program

the web link information of current url, which is stored in set web. Line 5 samples the web pages from the current url and get the returned value [λ, η]. λ is the similarity threshold and η is the specified amount of pages under the directory node.Line 6 takes η links randomly and saves them into sampling set web'. Line 7 using the function simParse() to calculate and analyze the web similarity to get the similarity of current url. Line 8-13 lines analyze the similarity, if the similarity is larger than the threshold λ, we can conclude that the pages linked with current url come from the same template and only one of them need to be detected; otherwise, we need recusing the algorithm to detect the pages from the current directory as root directory. After all the child directories have finished traversing,we conduct SQL injection testing on the pages of root directory. The algorithmic process is shown in Fig. 2.

Web crawler design: This web crawler traverses the current page by breadth-first strategy and crawls all links which meet the conditions have not been visited links. The design of web crawler module is shown in Algorithm 2:

Algorithm 2: Crawler(current)

Require:
The current root web page, current.
Ensure:
The queue of the pages which have been visited, visitedURL
1 : download the links in current and save into the set currentWeb
2 : Parse the links in currentWeb into the queue urlQueue
3 : Initialize the set visitedURL
4 : **for all** url in urlQueue **do**
5 : if url is injectable and not been visited by crawler **then**
6 : add the url into the set visitedURL
7 : **else**
8 : calculate the similarity between url and the links in visitedURL as simValue
9 : **if** simValue>λ **then**
10 : **continue;**
11 : **end if**
12 : **end if**
13 : **end for**
14 : **return** vistiedURL

Alogrithum 2 shows the pseudo-code of web crawler mod-ule. The first 1-2 lines download and parse the current page. currentWeb and urlQueue preserve the current pages and the queue of current pages, respectively. Line 3 initializes the queue visitedURL to store the visited urls which meet the conditions. Line 4-10 design the process of crawling url. And the crawled web

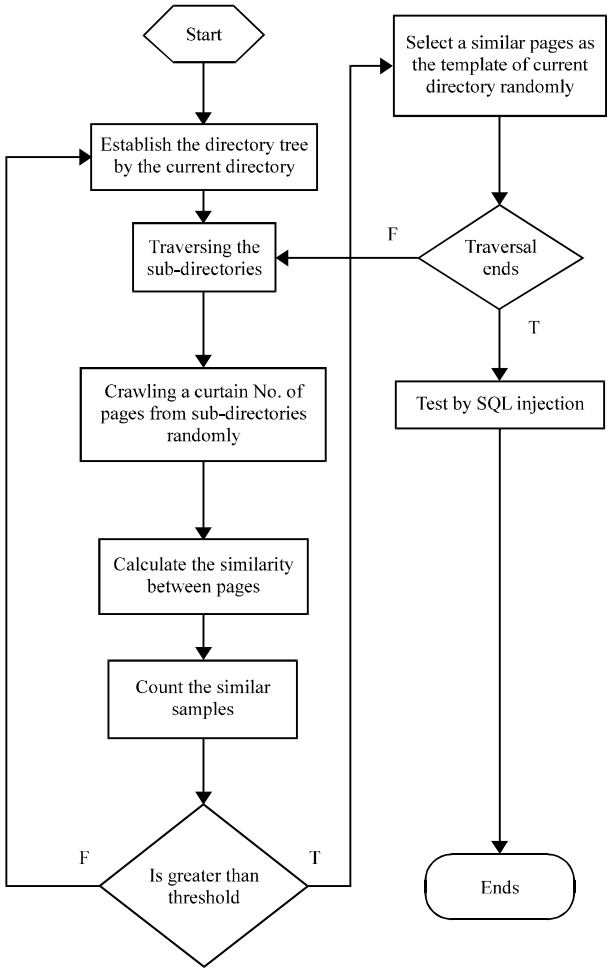

Fig. 2: Algorithm of template matching

pages are used for SQL injection testing, so it is not necessary to crawl static url addresses without injection point. This function eventually returns the url queue which may have the possibility of SQL injection vulnerability in the current page. Figure 3 shows the process of web crawler.

Pages sampling: This study samples pages based on the hyper-geometric distribution. Hyper-geometric distribution is statistically a kind of d iscrete probability distribution in statistics.

In the unreturned sampling of web similarity calculation, if in N urls there are M urls are dissimilar web links and the dissimilar number X = k is obtained from sampling n, then:

$$P(X = k) = \frac{C(M,k) * C(NM, nk)}{C(N,n) * C(a,b)}$$

is a combination of classical probability, in which a is the minimum and b is the limit. We call the random variable X follows the hyper-geometric distribution.

In this study, we randomly grab limited amount $\eta \in [Y, Z]$ of pages. Yand Zare set in advance and start to calculate the number when reaching the threshold λ of similarity probability.

Similarity calculation and analysis: To calculate the structural similarity, we get the tag sequences of the sampled pages by parsing the label arrangement of page structure and filtering form tags and static text labels. Then we calculate and compare the similarity between tag sequence of pages to judge the similarity. The similarity threshold of page structure which obtained based on the experiment results which will be shown in evalution section. Tag sequence similarity calculation makes use of the classical Longest Common Subsequence

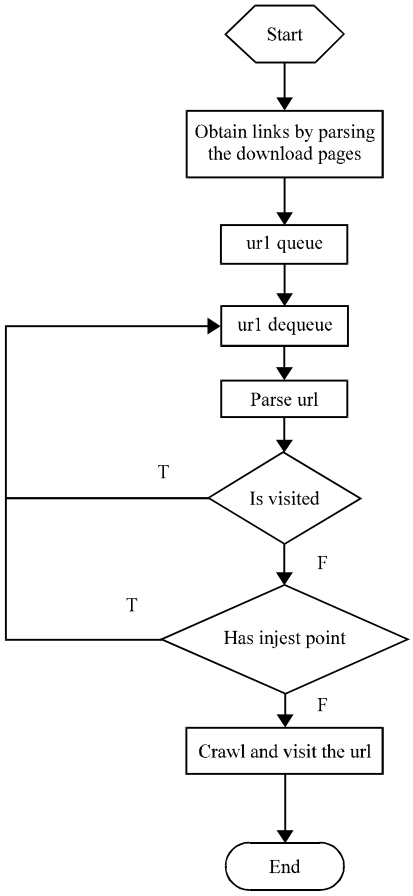

Fig. 3: Work flow of the web crawler

(Wagner and Fisher, 1974) (LCS Algorithm) in the field of Dynamic Programming. The pseudo code for this algorithm is shown in Algorithm 3:

Algorithm 3: SimParse(web)

Require:
The set of web pages, web.
Ensure:
The percent of the similar pages among the input web pages,simValue
1 : initialize tagQueue, tempQueue
2 : Parse tag sequence of web into tagQueue
3 : save the size oftagQueue into len, set the 0 into slen
4 : **while** tagQueue is not empty **do**
5 : dequeue from tagQueue as temp
6 : **if** tempQueue is empty **then**
7 : enqueue the temp into tempQueue
8 : **else**
9 : **for all** sq in tempQueue **do**
10 : compare the similarity between sq and tempQueue by simCompare(sq, temp) as sim;
11 : **if** sim>=δ **then**
12 : slen++
13 : **break**
14 : **end if**
15 : **end for**
16 : **end if**
17 : **end while**
18 : calculate the simValue by slen/len
19 : **return** simValue

Function simParse() calculates the successful matching probability of sampling pages, i.e., the percentage of number of pages in the template queue to pages. If successful matching probability is certainly greater than the threshold δ, then we do not need to continue crawling other pages; otherwise, crawling will be continued. Line 1 initializes the tag queue tagQueue and the template queue tempQueue. We analyze the page in line 2. The specific process of parsing is: (1) Obtain a site link, (2) Set the variable to filter static tag and form and (3) Get all labels and save them into the queue tagQueue. Line 3 sets the number of current page and the number of similar pages. Line 4-16 traverse the queue and analyze the similarity through matching with the pages from template queue and calculate the number of similar pages. In the process of pages matching, function simCompare() is responsible for calculating the structural similarity of web pages. The function parses pages into label sequences and calculates the longest sequence to obtain the similarity between tag sequences, then saves the result to the variable sim. Line 17-18 calculate the percentage of similar pages and return the similarity simValue.

SQL injection testing: The application of blind SQL injection firstly need to lookup and validate the injection points. These three ways can be helpful (Liu and Ma, 2011): (1) Cause general error, (2) Confirm blind spot and (3) Splitting injection.

We mainly uses blind SQL injection testing in this study, which sends a specially constructed SQL statement to the server in order to access the database. We analyze the feedback returned by the sever to determine whether SQL injection vulnerabilities exists and if there are vulnerabilities, returns type of vulnerability.

EVALUATION

Similarity threshold validation: In order to validate the optimal value of similarity threshold δ, we select 300 groups of pages from isomorphic sites, such as buying sites, news sites, technology sites, a university library and a forum to proceed the similarity test, each of them from the same group can be identified as the similar pages. After analyzing the Statistics of the test result and the suitable similarity threshold, we select 300 groups of pages from isomorphic sites, the distribution of web structural similarity is shown in Fig. 4.

Based on the results of this experiment, we selects δ = 0.85 as the optimal threshold of page structure similarity.

Crawler efficiency: In order to test the advantage of improving efficiency of the web crawler, we select several

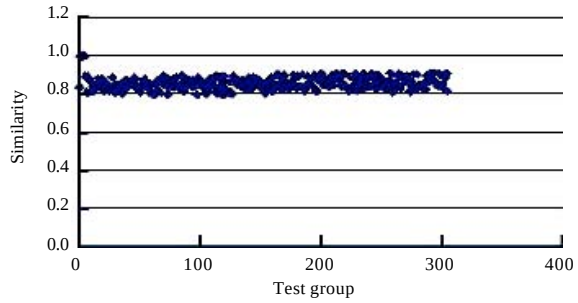

Fig. 4: Distribution of page similarity

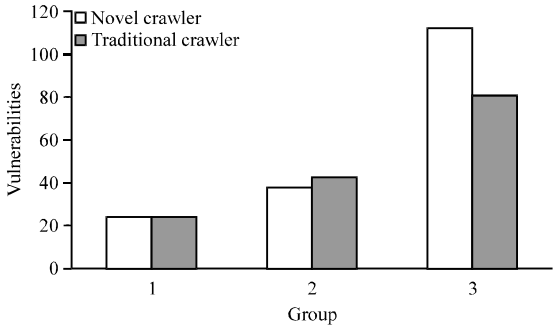

Fig. 5: No. of detected SQL injection points under different crawling strategies

Table 1: Crawling results of web crawler

Web style	Novel crawler	Traditional crawler
News	129	>10000
Buy	108	>10000
Technology	88	>10000

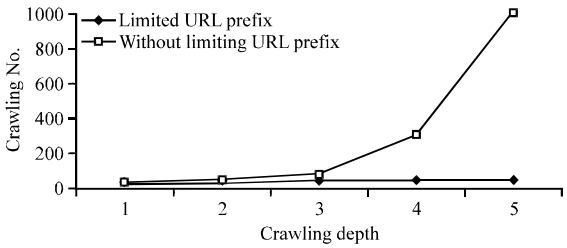

Fig. 6: Relationship of the No. of crawling pages and crawling depth before and after improving

sites like news sites, technology sites and buy sites to crawl the pages as three groups in our experiment with the depth of 3. The result shown in Table 1 is the average number of crawling pages in each group. The results is shown in Table 1.

Table 1 shows that, the novel web crawler based on template matching has un-competed advantages on SQL injection vulnerability scanning over traditional crawler, it can greatly reduce the number of pages to crawl and improve the detection efficiency significantly.

Detection efficiency: In order to test the efficient detection, we conduct SQL injection testing on the crawled pages. Group 1, 2, 3 are the results of crawling a buy site, a forum site, a technical site results in two ways, where the buy site and technology site are well-known large sites, while the forum site is a small site. The experiment sets the crawling depth as 3 and the crawling upper limit as 10000. The result is shown in Fig. 5.

Figure 5 shows that, due to the specified crawling upper limit, the novel crawler can quickly select different structures of pages in the detection process of large sites, while the traditional crawler need to detect a lot of pages with repeated structures. In the detection of small site, detection rate of our method in injection point accords with the traditional method basically. Although, the traditional web crawler may get a more complete detection of injection point by getting the greater crawling upper limit in some cases, the huge overhead caused by inefficient crawler is intolerable.

Crawler depth testing: In addition, we carry out the depth experiment of web crawler and reflects the dependence of crawling depth in our algorithm by comparing the relationship between the number of crawled pages and the depth of crawling before and after increasing url prefix. The functions of each web forum varies, the sensitivity of crawling depth is also shown in different conditions, so we take the sports template under a news site for example to evaluate the the sensitivity of crawling depth by different methods. The our evaluate result is shown in Fig. 6.

Figure 6 shows that, although this method can be largely avoided crawling repeated structure pages, unsuitable depth setting often leads to desired result because of the sensitivity of crawling depth. In Fig. 6, the optimal crawling depth is 3. Once larger than this value, it is inevitable to crawl unrelated pages through the external links. This will not only affect the efficiency of crawling but also affect the final result of the scanning. We limit the crawling scope of all the forums need to be scanned by limiting the url prefix of crawled pages to avoid the interference of external links. As our method is depth-insensitivity, our web vulnerability scanner will achieve a strong adaptability.

CONCLUSION

We focuses on the features of SQL injection vulnerability and proposes a method of web similarity calculation based on template matching in this study,

which will conduct similarity and search the page template. Through, avoiding crawling web pages with a similar structure, we obtain a higher efficiency of crawling pages in the process of web vulnerability scanning. Meanwhile, we introduce a method to avoids interference from external web links to the experimental result by set the limit of URL prefix. Experimental result shows that the proposed solution obtains not only the higher efficiency of crawling pages but also the stronger adaptability to insensitivity on the depth setting during the web vulnerability scanning.

ACKNOWLEDGMENT

This study is supported in part by the Zhejiang Provincial Technical Plan Project (No. 2012C11026-4).

REFERENCES

Huang, Y.W., S.K. Huang, T.P. Lin and C.H. Tsai, 2003. Web application security assessment by fault injection and behavior monitoring. Proceedings of the 12th International conference on World Wide Web, May 20-24, 2003, Budapest, Hungary, pp: 148-159.

Liu, H. and Y.X. Ma, 2011. Web data extraction research based on wrapper and Xpath technology. Adv. Mater. Res., 271-273: 706-721.

Peng, G. and M.Y. Fan, 2010. Crawler technology based on improved SQL injection vulnerability detection. Appl. Res. Comput., 27: 2605-2607.

Shin, Y., 2006. Improving the identification of actual input manipulation vulnerabilities. Proceedings of the 14th ACM SIGSOFT Symposium on Foundations of Software Engineering, November 5-11, 2006, Portland, USA.

Wagner, R.A. and M.J. Fisher, 1974. The string-to-string correction problem. J. ACM (JACM), 21: 168-173.