International Journal of Security and Its Applications
Vol.9, No.9 (2015), pp.33-44
http://dx.doi.org/10.14257/ijsia.2015.9.9.04

Malicious URL Detection Algorithm based on BM Pattern Matching

Fuqiang Yu

College of Information Technology, Hebei Normal University, Shijiazhuang 050024, China

Abstract

In the virus world of Internet, it is a challenging and urgent problem that how can we ensure the safety of search engines. A security subsystem of the search engine based on the research of content-based image search engine system V2.0 is developed. A malicious URL (Uniform Resource Locator) detection method based on BM (Boyer-Moore) pattern matching is proposed. The main research contents and results are as follows: Many malicious URLs could be downloaded by web image search, which may cause unnecessary loses to the users. So the malicious URL detection algorithm based on BM pattern matching is proposed. This method is to let the URL source code match the virus characteristics in the database to confirm whether the URL is safe or not. Web image search detects 203 malicious URLs based on this method. By kaspersky scanning, we confirmed 189 URLs to be malicious URLs, and the error rate is 6.9%, and the accurate rate is 93.1%. The experimental results show that the malicious URL detection algorithm provides secure URLs for web image search engine..

Keywords: *search engine system; malicious URL detection; BM pattern matching; security URLs; web image search engine*

1. Introduction

With the development of the computer technique, Internet has become a more and more popular tool in daily life. However, along with all the utilities it brings to people, it also causes many threats to Web security. Currently, Web security is becoming a key factor in the development of web constraints. In order to improve the security of the Web site, use the web vulnerability scanning tools on the site for safety testing, and try to eliminate safety hazards that may be exploited by attackers [1-4]. Specify the address of the web page, and the first and foremost task of security testing is to identify all external interfaces exposed by the page that attacked the entry point [5-6].

Now, some of the top foreign companies and research institutions are conducting the related research and developing some systems that can run the test, and its application areas include the vulnerability found, the worm detection and the identification of the dangerous websites. So far, the international famous research institutions and commercial companies had developed their own application systems [7-9], such as Microsoft HoneyMonkey system; IBM's Billy Goat system; Symantec company's Script Blocking technology. Google's engineers think that the remote attack and firewall are already past and take advantage of the browser, and Google utilizes the strengths and resources of their own search engine technology. First, the URL extraction technology obtains URL in the webpage resource library [10]. Then the extracted URL is transmitted to the implementation, monitoring and analyzing sections. The part uses the virtual machine and the IE browser to analyze, and the final results are returned. Malicious URL will be stored in the malicious Web resource library. Google also take advantage of some of the third-party agency's

ISSN: 1738-9976 IJSIA

data, which mainly comes from the www.stopbadware.org, and the agency's data has been integrated into the Google search results, and the labeling of dangerous results is returned to the users [10-12]. In addition, Google also concerns about the security of the content of the website containing harmful code hidden in the code of the page itself, such as web redirection information. Google particularly concerns about the use of the advertising information page. Google has acquired probably 300,000 suspicious URLs. After the final confirmation of the detection system, about 10,000 to 30,000 URLs are harmful. In addition, the University of Washington, there are some scholars in the study of spyware on the Web; the main idea is based on reptiles [13-15, 18-21].

The paper proposes the malicious URL detection method based on BM pattern matching on the basis of the analysis of the many sources characteristics of URLs. This method let the URL source code match the virus characteristics in the database to confirm whether the URL is safety or not. A large number of malicious URL detection experiments are done, and the malicious URLs are found by the experiments, and the labels are given. When the users use a Web image to search engine pictures, if check out the picture where the URL is malicious URL, the search engines give out a message, and the users decide whether they open this URL or not. So, to some extent, it protects the safety of the users of the search engine and improves the credibility of Web image search engine.

2. Theoretical Basis

2.1 Related Concepts of Malicious URL

URL is a string used to describe the information resources on the Internet. URL is also known as a web address, which is the standard resource address on the Internet. There is the information of protocol, host, port, path access the webpage in the URL. It was originally used to as the address of World Wide Web (www) invented by Tim Berners-Lee. Now it has been the standard RFC1738 for the Internet by the World Wide Web Consortium. So we can access the webpage's of our own needs through the URL. Malicious URL refers to a Web page that contains malicious code corresponding URLs. When users visit these malicious URLs corresponding website, There may result in the malicious behavior of the passwords leaks and damaging file on the host computer. The malicious code is a program by embedding malicious code secretly to another section of the program, so as to achieve the destruction of computer data, run with destructive or invasive procedures to achieve the purposes of the destruction of the infected computer data security [16]. Malicious code generally has the following characteristics, such as malicious purpose, it is a program having a role by performing the program. There are four categories of the transmission: Trojan horses, viruses, worms, and mobile codes. The malicious code can not identified through some special encryption means on the general situation.

2.2 BM Pattern Matching Algorithm

BM algorithm is an exact string matching algorithm (as distinguished from fuzzy matching). BM algorithm uses the right-to-left comparison method and applies to two heuristic rules that the bad character rule and the good suffix rule to decide the right jump distance. In the process of matching in the BM algorithm, take the larger in Skip (x) and Shift (j) as the jump distance.

3. Issues are Raised

Source code analysis is one of the static analysis techniques. Source code analysis analyzes the source code of each URL, usually with predefined suspicious signature pattern matching to detect malicious URL. Malicious code carries out the malicious behavior generally making use of the following three techniques: code obfuscation, URL redirection and exploits [17].

3.1 Code Obfuscation

In order to evade the detection analysis of the malicious behavior, some malicious websites use the code obfuscation techniques to evade anti-virus software signature-based scanning. Such as:

1) eval () functions and document write () function are injected to the script to achieve dynamic code injection.

2) In order to form an unreadable long string, use escape () function encoding the character, and finally use unescape () function in the script or a browser to decode.

3) In order to achieve the sub-string replacement, often use a function or variable.

4) Custom decoding procedures are written in the script.

3.2 URL Redirection

Many malicious sites automatically redirect another URL of the browser. When the browser accesses a URL, the response time of this URL will automatically instructs the browser to access one or more other URLs without affecting the contents of the display on the user. Redirection uses he following technologies:

1) Use the response code 301, 302 of http protocol to redirect.

2) Use html tags including the src attribute of the iframe, frame in the frameset and script tags to link to external url.

3) Use the function of the script including window. location. Replace (), window. location. href () and window. open ().

3.3. Usage of the Loopholes

The malicious Web system or browser vulnerabilities take advantage of the program, when the users access these pages, these procedures can download Trojans or other malicious software on the host resulting in the unsafe state of the host. To use the Internet browser, the Internet Explorer vulnerability risk level higher, using methods generally divided into the following two:

1) Due to loopholes error lead to the implementation of shellcodein in the source code of the page that contains shellcode.

2) Use the component or other vulnerabilities to download and run the program. The common functions are createobject () function, ActiveXObject () function or the URL address file contained in object tags.

Need two checks for page code for the above characteristics. The first level is the extracted signature pattern matching for the script. Focus on the encryption function, the high risk level script function into check. The second level, check the length of the URL and file name suffix in the high-risk level iframe, frame, script and link tags and check the object tag if they contain the URL address of the file.

Some characteristics of the malicious URL source code are analyzed above containing the URLs owning these features downloaded to the database server through a Web image search, search engines available to users of these URLs, and it may cause user PC poisoning and destruct the users host data, steal information stored on the user host, user name and password, and may even cause hardware

damage. The purpose of the URL static detection method is these malicious URLs detected.

4. Method of this Article

4.1 Malicious URL Detection Implementation Process

The malicious URL static detection is the analysis of data in the web page source to see if it contains the malicious code to identify these malicious codes, which can use pattern matching algorithm [13]. The source codes downloaded by the data finder match the viruses of the database. If the match is successful, this URL is suspicious, whether it is secure, that attribute is set to false, if the match is not successful, it indicates that this URL is yet to be investigated; his security property is set to true. URLs source codes are as the main string, and the characteristics of a virus are as the substring, and find the signature of the virus in the source code. If we find this URL is unsafe. If you did not find the description of this URL and it is safe. The specific steps are as follows:

Step1: BM pattern matching algorithm matches for URL source and virus signatures.

Step2: The malicious code and malicious URL description are stored in the database as a BM mode matching substring.

Step3: Detect malicious Web data structure for malicious web page source codes and database signature matching to prepare;

Step4: Detect malicious Web algorithm descriptions to search collected URL of the source code and malicious signature match.

The following subsections describe the four steps of this method process.

4.2 BM Pattern Matching Algorithm

Set the text string T and the pattern string P. First, T and P are done left-aligned, then from right to left for comparison as shown in Figure 1.

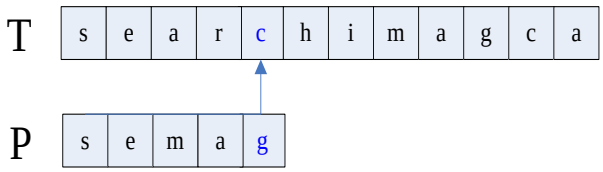

Figure 1. Initialization Location Map of Characters Matching

If a comparison does not match, BM algorithm uses two heuristic rules, that is the bad character rule and the good suffix rule to calculate the mode string moving to the right distance until the end of the matching process.

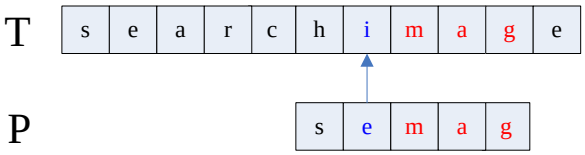

Figure 2. Good Suffix Maps of Bad Character

In Fig.2, the first character that does not match (i in T and e in P) is the bad characters and the matched portion (mag in T and P) is good suffix.

4.2.1 Bad Character Rule: In the right-to-left scanning process of the BM algorithm, if the character x does not match, the following discussion of two cases are done.

1) If the character x does not appear in the mode P, m texts from character x are obviously impossible to match P successfully, directly skipped entirely to the region.

2) if x is in mode P, the characters are aligned.

Represented by mathematical formulas set Skip (x) is the distance of the P shifted rightward, the length m for the pattern string P, max (x) is the right-most position of the character x in P.

$$Skip(x) = \begin{cases} m; x \neq P[j](1 \leq j \leq m) \\ m - \max(x); \{k \mid P(k) = x, 1 \leq k \leq m\} \end{cases} \quad (1)$$

For example, Fig.3 the blue part (g and c), the occurrence does not match for once.

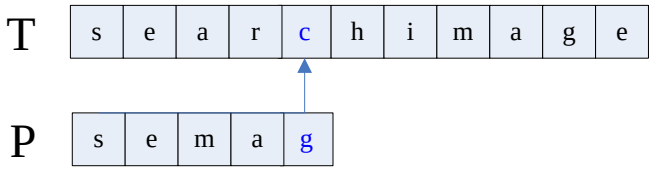

Figure 3. Bad Character Map

Calculating the moving distance skip (c) = m = 5, then P moves to the right 5. Move is as shown in Figure 4.

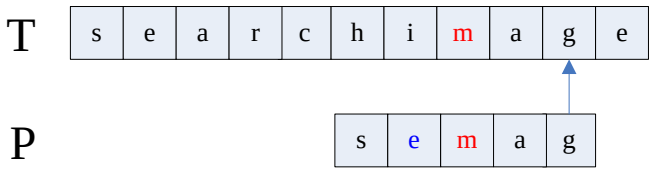

Figure 4. Hops and Jumps of Bad Character

4.2.2 Good Suffix Rule: If a character does not match at the same time and the part of the character match is successful, the following two cases will be discussed.

If the matched portion P' is a certain position in the P' at the position t' in P has been also appeared, and the position t' of the previous character and the position t of the previous character are not the same, the P shifts right so that t' is the corresponding t just the location.

If any location in P has been matched part of the P' that does not occur again, then find the longest prefix x of P as the same P'' and the suffix of P', and P is moved rightward so that x corresponds to just P'' suffix location.

Let Shift (j) as a model for the distance P is shifted rightward, m is the length of the string of P, j is the current matching of character positions, s is the distance of t 'and t (in this case i) or the distance x and P'' represented by mathematical formulas (in this case ii).

$$shift(j) = \min\{s \mid P[j+1 \ldots m] = P[j-s+1 \ldots m-s]\} \,\&\&\, (P[j] \neq P[j-s])(j > s), P[s+1 \ldots m] = P[1 \ldots m](j \leq s) \quad (2)$$

For example, in Fig.5, match part of the mag (blue) did not appear in P.

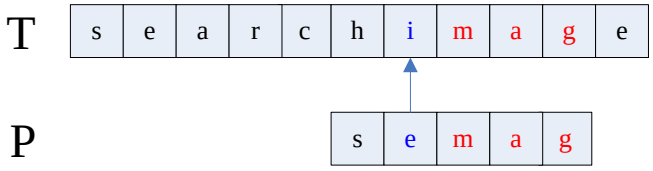

Figure 5. Suffix Map

Mag in does not appear in se, so dealt with in accordance with the second case of the good suffix rule. But after hopping distance moves 5 according to the good suffix rule, P beyonds the length of the P, so the match is not successful, and the match ends.

4.3 Malicious Codes and Malicious URL Description

Malicious codes in this article contain some source code obfuscation, URL redirection, correlation function of the vulnerability use, and the specific functions are shown in Table 1.

Table 1. Malicious Codes

Match type	Specific matches description
Code obfuscation	document.write (), eval () escape (), unescape (), etc.
URL redirection	replace, href, open, etc. of window.location
Exploitation of the vulnerability	shellcode createobject (), activexobject (), object, etc.

The major malicious URL is the recognized virus URL and site, and the specific depictions are shown in Table 2.

Table 2. Malicious URL

Malicious URL	Corresponding website malicious URL
QQ44455	hxxp://www.QQ4455.com
888999	hxxp://www.888999.com
winzhengwo	hxxp://winzhengwo.126.com
kimo	home.kimo.com.tw/avnvyou520/
…	…
pixpox	hxxp://www.pixpox.com

4.4 Detect Malicious Web Data Structures

(1) malicious code characteristics data structure
{
Private string ID;
Private string character;
Private string type;
}
Virus ID number (14 bytes), the virus signature (200 bytes), the virus types (4 bytes).
(2) URL of the malicious code data structure
Public class vituesurlInformation
{

Private string id;
Private string website;
Private string vitus_url;
Private sting vitus_description;
}

Virus URL ID (14 bytes), the virus URL site where (200 bytes), the virus URL (4 bytes), the virus URL description (200 bytes).

4.5. Detect Malicious Web Algorithm Description

Algorithm: BMSearch (* ptrn * buf)
Input: the source code buf of the Web page, virus signatures ptrn in the database
Output: true or false
Steps: B1 [Find matching signature] use ADO connection database technology of VC to look up virus database feature items.

B2 [Judge whether the string match to the end] Signature for comparison determines the match live source code and viruses. If there is no match to the end, swerve to B4. If the match to the end, swerve to B5.

B3 [Calculate jump length] calculate jump number of bytes in accordance with the rules of the bad character and calculate the jump length according to the good suffix rule.

B4 [Match] matching success returns true, otherwise it returns false, swerve to B3.

B5 [End] return results and exit the module.

The flowchart of signature matching is shown in Figure 6.

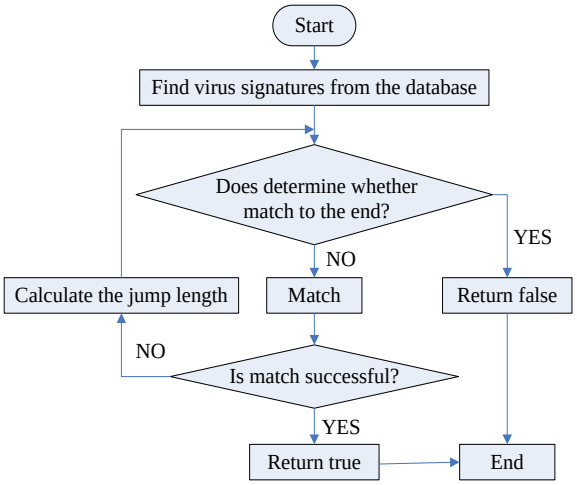

Figure 6. Flowchart of Signature Matching

5. Performance Analysis

The malicious URL detection based on BM pattern matching algorithm has been implemented in the previous section, and the method does the pattern matching of the URL source codes downloaded by the searcher finder and virus signatures and virus URL in the database to identify suspicious URL. There are many pattern matching algorithms, and this paper uses the BM pattern matching algorithm, this is because of the higher efficiency of BM pattern matching algorithm. Let's analyze the time complexity of the three methods.

For BF algorithm, let the length of the main string T is n and the length of the substring P is m. In the matched successful case, consider two extreme cases.

In the best case, per trip unsuccessful match occurs in the comparison of the first pair of characters, set match success occurred in si at the number of character comparisons in the previous match i-1 trip compared i-1 times, i trip successful match is total m times, so compare the total i-1+m times, and all matching possible successes is total n-m+1 times. The succeeded matching probability is pi with T from si, in the case of probability pi = 1/(n-m+1), so the average number of comparisons in the best case is:

$$\sum_{i=1}^{n-m+1} pi^{*}(i-1+m) = \sum_{i=1}^{n-m+1} \frac{1}{n-m+1} * (i-1+m) = \frac{n+m}{2} \quad (3)$$

That is, the time complexity is O(n+m) in the best case.

In the worst case, each unsuccessful matching is the last character of T: set successful matching occurred in si positon and compare total (i-1)*m times in the previous i-1 matches, and compare m times at the i-th successful match, so a total comparing is i*m times. Thus the average number of comparisons in the worst is:

$$\sum_{i=1}^{n-m+1} pi * (i*m) = \sum_{i=1}^{n-m+1} \frac{1}{n-m+1} (i*m) = \frac{m*(n-m+2)}{2} \quad (4)$$

That is, the worst-case time complexity is O(n*m).

KMP algorithm is unlike BF algorithm as each match fails only pattern string P right one character comparisons from P [1] began, so the time complexity of KMP algorithm is lower than O(m*n) of BF algorithm. The time complexity is O(m) in the steps of initializing Next [j] data of KMP algorithm, and the time complexity in the string matching process is O(n), so the time complexity of KMP algorithm is O(m+n).

The time complexity of BM pattern matching is analyzed by the same method, and the time complexity is O(n/m) in the best case and the time complexity is O(n*m) in the worst case.

The above three algorithms are pattern-matching algorithm, and which one is more suitable for further analysis to be applied to the detection of malicious URLs. The time complexity of the above three algorithms is done the analysis, the following are the contrast as shown in Table 3.

Table 3. Time Complexity Analysis of Three Algorithms

Pattern matching algorithm	BF	KMP	BM
Best time complexity	O(m+n）	O(m+n）	O(n/m)
Worst time complexity	O(n*m）	O(n*m）	O(n*m）
Average time complexity	O(n*m）	O(m+n）	O(n/m)

We know that BM algorithm has higher efficiency from the time complexity analysis of the mode matching algorithm, so the malicious URL detection algorithm based on BM mode matching has lower computational complexity.

6. Experimental Results and Analysis

In this paper, the search engine SooTu based on the contents of the independent development is as the experimental platform. The experiment is the comparison of the collected virus URLs and the entire collected URLs, and then the virus URLs are validated to analyze in other software. The system uses Java and VC language to implement. The evaluation of the system performance adopts the detection rate of

the virus URL as the measure. The detection rate of the virus URLs is defined as follows:

$$V = \frac{D}{S} \tag{5}$$

Where, D is the number of the suspicious URL web image search system detected within a certain time, the unit is number. S is the total amount of the URL data downloaded from the Web, Web search systems within a certain time.

6.1 Experimental Conditions

This article is developed on the basis of the development of V2.0 of our group, the experimental search on these performance advantages of V2.0. The details are as follows:

(1) The search strategy is based on the depth-first breadth.
(2) Use I/O buffers of two types of images and URL disk simultaneously. The double buffer queue number of elements in the URL to be mined queue is set to 100, and the image disk I/O cycle buffer pool and URL disk I/O.
(3) Use multiple threads to download URL and the images, and the number of threads is set to 12.

6.2 Experimental Results

The experiment is done specific statistic data based on the above experimental conditions for consecutive eight days for the downloaded URL number and the number of virus URL shown in Table 4.

Table 4. URL Experimental Data

Time(d)	Download URL	Virus URL
1	2016	21
2	3026	37
3	2514	20
4	3122	57
5	2817	13
6	3011	26
7	2888	11
8	3090	18

Calculate the detection rate of $v_1, v_2, \ldots, v_8$ of the daily malicious URLs according to the formula (5) by URL experimental data in Table 4, respectively. Calculate the average detection rate $\bar{v}$ the malicious URL of the daily malicious URLs. The results are shown in Table 5.

Table 5. Detection Rate of Malicious URL

	Detection rate
v_1	1.05%
v_2	1.23%
v_3	0.80%
v_4	1.84%
v_5	0.46%
v_6	0.46%
v_7	0.38%

v_8	0.58%
$\bar{v}$	0.90%

From Table 5, the average detection rate of the virus URL is 0.90%.

6.3 Analysis of Experimental Results

The content-based Web image search visits total 22500 URLs belonging to 1027 domains, and 203 suspicious malicious pages are found accounting for 0.725% of all pages. The malicious codes are 189 confirmed through Kaspersky scan, and the accuracy rate is 93.1%, and the false rate is 6.9%. Fig.7 describes the number of the malicious URLs.

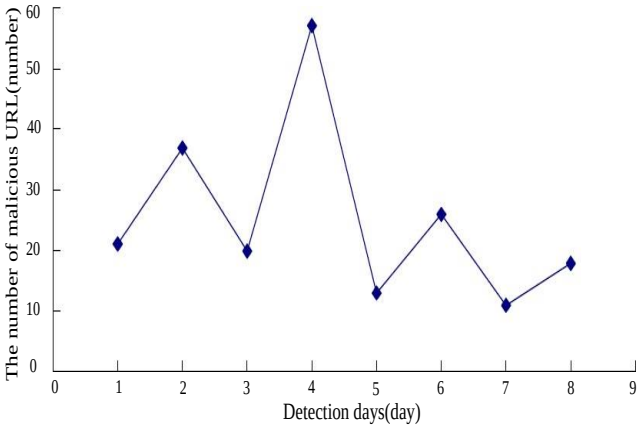

Figure 7. Detection Number Diagram of Malicious URL

Due to uneven distribution of daily visits to the page type the number of daily detected malicious URLs from Fig.7 is inequality. If the visited pages focus on the category of music, pornography, likely due to the these pages entrainment malicious codes, so the number of malicious codes captured will more than others. Captured virus URL is verified by the Kaspersky virus engine. It can been seen that the code obfuscation accounts for 27%, and the exploits account for 33%, and the URL redirection accounts for 40% by the analysis of the detected virus URL as shown in Fig.8.

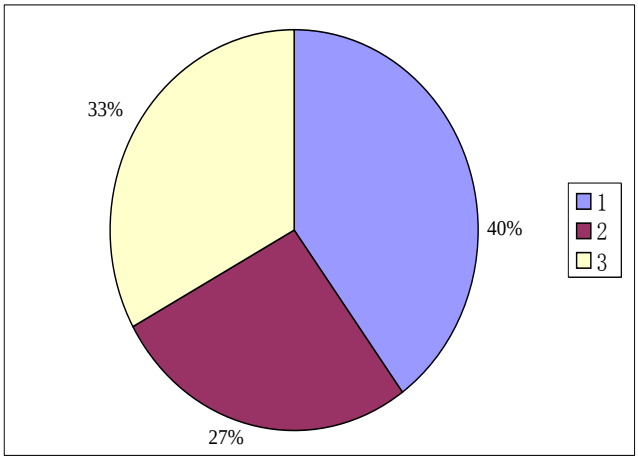

Figure 8. Category Maps of Malicious URL

In Figure 8, blue (1) is URL redirection, and red (2) is the code obfuscation, and yellow (3) is the exploit.

7. Conclusion

In this paper, the malicious URL detection based on BM mode matching is implemented. The source codes of URL and the malicious functions in the database are done the pattern matching through the analysis of the finder downloading the source codes so that the malicious URLs can been detected. Web image search engine queries the pictures. When the origin of the picture is malicious URL, give the user a prompt and the users themselves decide whether to open URL. It is proven that the detection for the malicious URL is obvious effect by doing a lot of experiments. And URLs are validated in the professional anti-virus software and the accuracy rate is more than 90%.

References

[1] F. Noriyuki and H. Pkenichi, "A personal system for web image retrieval", Proceedings of the 4th international symposium Information and communication technologies, **(2005),** pp. 209-216**.**
[2] S. Brin and L. Page, "The Anatomy of a Large-scale Hypertextual Web Search Engine", Computer Networks, vol. 30, **(1998),** pp. 107-117**.**
[3] E. Kirda, C. Kruegel and G. Vigna, "A Client-side Solution for Mitigating Cross-site Scripting Attacks", Proc. of 2006 ACM Symposium on Applied Computing, **(2006),** pp. 330-337**.**
[4] A. Mori, T. Lzumida and T. Sawada, "A Tool for Analyzing and Detecting Malicious Mobile Code", Proc. of the 28thInternational Conference on Software Engineering, (**2006),** pp. 831-834**.**
[5] L. Thierry, "Fast exact string matching algorithms", Information Processing Letters, vol. 102, no. 1, **(2007),** pp. 229-235**.**
[6] P. Euripides, V. Epimendes and M. Evangelos, "Searching for logo and trademark images on the web", Proceedings of the 6th ACM international conference Image and video retrieval, **(2007),** pp. 541-548**.**
[7] X F. He, D. Cai and J R. Wen, "Clustering and searching WWW images using link and page layout analysis", ACM Transactions on Multimedia Computing, Communications, and Applications, vol. 3, no. 2, **(2007),** pp. 10-35**.**
[8] K. Gunhee and T. Antonio, "Unsupervised Detection of Regions of Interest Using Iterative Link Analysis", Advances in Neural Information Process in Systems, **(2010),** pp. 467-474.
[9] T. Dziubak and J. Matulewski, "An object-oriented implementation of a solver of the time-dependent Schrodinger equation using the CUDA technology", COMPUTER PHYSICS COMMUNICATIONS, vol. 183, no. 3. **(2012),** pp. 800-812**.**
[10] K. Wang and A R. Smith, "Efficient kinematical simulation of reflection high-energy electron diffraction streak patterns for crystal surfaces", COMPUTER PHYSICS COMMUNICATIONS, vol. 182, no. 10, **(2011),** pp. 2208-2212.
[11] A N. Du, B X. Fang and X C. Yun, "Comparison of stringmatching algorithms: An aid to information content security", International Conference on Machine Learning and Cybernetics, **(2003),** pp. 2996-3001**.**
[12] M. Karim, A. Walenstein and A. Lakhotia, "Malware phylogeny generation using permutations of code", Journal in Computer Virology, vol. 1 , no. 1–2, **(2005),** pp. 13–23.
[13] J. Kolter and M. Maloof, "Learning to detect and classify malicious executables in the wild", Journal of Machine Learning Research, vol. 7, (**2006),** pp. 2721–2744.
[14] M. Christodorescu, S. Jha and S A. Seshia, "Semantics-aware malware detection", In: IEEE Symposium on Security and Privacy, **(2005),** pp. 32–46**.**
[15] C. Grier, S. Tang and S T. King, "Secure Web Browsing with the OP Web Browser", In Proceedings of the 2008 IEEE Symposium on Security and Privacy (sp 2008), **(2008),** pp. 402-416**.**
[16] Y M. Wang, D. Beck and X. Jiang, "Automated web patrol with strider honey monkeys", In: Proceedings of Network and Distributed Systems Security Symposium, pages 35C49, **(2006).**
[17] [17] Moshchuk A, Bragin T, Deville D, et al, SpyProxy: Execution-based Detection of Malicious Web Content, In Proceeding of the 16th USENIX Security Symposium, **(2007),** pp. 118-121.
[18] J. C. K. Yau, L. C. K. Hui and S. N. Cheung, "Trustworthy Browsing - a secure Web accessing model", The 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service, **(2005),** pp. 542 - 547**.**
[19] C.-M. Chen and Y.-H. Ou, "Secure Mechanism for Mobile Web Browsing", 2011 IEEE 17th International Conference on Parallel and Distributed Systems (ICPADS), **(2011),** pp. 924 - 928**.**

[20] S.-W. Hsiao, Y. S. Sun, F.-C. Ao and M. C. Chen, "A Secure Proxy-Based Cross-Domain Communication for Web Mashups", 2011 Ninth IEEE European Conference on Web Services (ECOWS), **(2011),** pp. 57 – 64.
[21] R. Bhandari and U. Suman, "Broker based secure web service composition using star topology", 2012 CSI Sixth International Conference on Software Engineering (CONSEG), **(2012),** pp. 1-7.

Author

Fuqiang Yu (1970-), Male, he is an associate professor of Hebei Normal University, received M.S degree from Hebei University. His research area mainly includes technology and security of network, security of computer's information, large scale data processing, database, etc.

