Malicious Web Pages Detection Based on Abnormal Visibility Recognition

Bin Liang\(^1\), Jianjun Huang\(^1\), Fang Liu\(^1\), Dawei Wang\(^1\), Daxiang Dong\(^1\), Zhaohui Liang\(^1\)\(^2\)

1. School of Information, Renmin University of China, Beijing 100872, China
2. Key Laboratory of Data Engineering and Knowledge Engineering (Renmin University of China), MOE, Beijing 100872, China

\{liangb, hjj, liuzhfang, wdw, ddx, lzh\}@ruc.edu.cn

Abstract—In recent years, Web sites have already become the attackers’ main target. When attackers embed malicious code in the Web pages, they generally change the display mode of the corresponding HTML tags to make the display effect of malicious code invisible or almost invisible to the browser users. In this paper, the concept of abnormal visibility is proposed to describe the display feature setting of malicious code embedded. According to the concept, a malicious code detection method based on abnormal visibility recognition is designed and a prototype system is implemented. Compared to traditional methods and systems, the method has higher efficiency and less maintenance cost. Besides, a special-purpose JavaScript interpreter is implemented to get the execution output of browser-end scripts that are often used to generate malicious code dynamically by attackers. Experiments show that this system can detect most of the malicious Web pages efficiently and at the same time locate the malicious code in the source code accurately.

Keywords—malicious Web pages; detection; abnormal visibility; JavaScript interpreting

I. INTRODUCTION

In recent years, with the rapid extension of Web application domain and the continuous exposure of the attack method, Web sites have become the attackers’ main target. In July 2008, we perform an investigation to 90 Web sites selected randomly in CERNET; results show that 29 of these Web sites contained malicious code, accounting for 32.2%. Namely, nearly one third of the Web sites contain malicious code. The investigation also shows that the Web sites containing the malicious code generated by JavaScript account for 39%. The most common attack mode is that the attackers intrude into the target Web server by exploiting the system vulnerability and then embed the Trojan horses in the Web pages. Once the users access the Web pages containing malicious code with Web browser, their computers might do some bad operations in background according to the will of attackers.

The existing methods of malicious code detection are almost based on the virus signature matching. That will result in a huge signatures database and high maintenance cost. Accordingly, the efficiency of detection will be affected. Besides, most of the Web page analysis systems cannot interpret and execute JavaScript scripts. It is difficult to detect the malicious code dynamically with code signature matching.

Based on investigation, we can find that when the attackers embed Trojan horses in the Web pages, they generally change the display mode of the corresponding tags in the Web pages so as not to change the display effect of browsers. On Sept. 2008, our investigation to 116 Web sites containing malicious code shows that the malicious Web pages with this feature account for 98.28%. Therefore, we propose the concept of abnormal visibility to describe this display feature setting of malicious codes. According to the concept, a malicious code detection method based on abnormal visibility recognition is designed and a prototype system is implemented.

The prototype system use Web spider and HTML parser to get and parse the Web pages from target site. The system will parse the codes of the Web pages and convert them into the data structures that can be recognized by the detection engine, then match these structures with the abnormal visibility fingerprints and locate the possible malicious codes in the source codes of pages. For the reason that the attackers often use JavaScript to generate malicious codes dynamically, this system can also interpret and execute JavaScript scripts. Experiments show that this system can scan a Web site real-time and detect most of the malicious codes in the Web pages efficiently, and accurately locate malicious codes in source codes of pages.

The rest of this paper is organized as follows. Section II describes related work. Section III gives the concept model and detection method. Section IV describes the design and implementation of prototype system. Section V presents the experiments, and Section VI concludes this paper.

II. RELATED WORKS

M. Benedikt et al. in Bell Lab give a solution in VeriWeb [1] to detect the executable paths of Web sites which may be exploited by attackers. They use browser to get links, predefined structure to fill forms. M. Almgren et al. perform intrusion-detection by attacking behavior pattern matching, which mines the server logging in real time [2]. However, it has to build new signatures for different sites. HoneyMonkeys [3] and HoneyC [4] are both based on Honey Pot technique. The former is costly and slow; the latter is also based on signatures of malicious codes. The system in [5] and Monkey-Spider [6] download possible malicious pages to local machines; scan them with specific tools such as anti-virus software. Depending upon the tools’ signature library, both of

This work has been supported by the National Natural Science Foundation of China under Grant No.60703102, No. 60873213 and No.60703103; the Beijing Science Foundation under Grant No.4082018; the National 863 High-tech Program of China under Grant No. 2007AA01Z414; the Open Project of MOE Key Laboratory of Data Engineering and Knowledge Engineering.
them can’t resolve special conditions and update in time. StopBadWare.org [7] is an organization especially established to detect malicious Web sites. It accepts partners’ informing, evaluates and detects pages following its Guidelines [8]. Recently, Google throws an alarm about possible malicious sites when searching with support of StopBadWare.org. Besides, DSWLab releases a product called Sucop to detect malicious Web pages [9]. Various anti-virus softwares can also detect the malicious codes passively at browser-end.

The weaknesses of the above works are as follows:

- All the above works are based on the virus signature, resulting in the large quantity of maintenance and the disadvantages of update. Method in [2] is not applicable for all cases.
- The methods based on the virus signature can’t be with high performance because of the huge cost of string matching.
- VeriWeb can analyze JavaScript scripts using string matching. However, it can hardly match all sorts of scripts used to embed malicious codes, especially encoded scripts.

III. DETECTION METHOD

A. Abnormal Visibility

In order to be not detected by browser-end users, the attackers generally change the display mode of the tags related to malicious codes embedded in Web pages to make it invisible or almost invisible. This characteristic of display effect of malicious codes is called abnormal visibility. Need to be noticed is that this kind of change can hardly occur in the normal Web page programming.

There are three main forms of abnormal visibility shown as follows:

1) To change the width or height attributes of iframe tag so that the display effect of the malicious codes embedded in the iframe tag is invisible or almost invisible. For example, as depicted in Fig. 1 (a), the values of width and height of malicious iframe tag are set to zero so that the display area will be zero. Consequently, the content of embedded malicious link would not be shown in browser and detected by end user.

2) Set the display style of the iframe tags to “display: none”. As shown in Fig. 2, although the values of width and height are normal, the tag is invisible.

3) Use browser-end scripts to generate malicious iframe tags dynamically. The example code is shown in Fig. 3 (a). After downloading, the script will be executed by browser and generate an iframe tag. As shown in Fig. 3 (b), in order to avoid the detection, many attackers also encode the JavaScript before embedding it in the Web page. After interpretation, the result of Fig. 3 (b) is the same as that of Fig. 3 (a). From the examples, we can see that it is impossible to identify the malicious tag codes from these scripts directly with string matching.

As shown in Table I, our investigation to 116 Web sites containing malicious codes shows that only 1.72% of the malicious codes are not with the abnormal visibility characteristic. In other words, we can detect most of malicious Web pages based on the effective recognition of abnormal visibility.
<table>
<thead>
<tr>
<th>Type</th>
<th>Number</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using <em>iframe</em> tags to embed malicious codes directly in the Web pages</td>
<td>25</td>
<td>21.55%</td>
</tr>
<tr>
<td>Using the JavaScript language to embed malicious codes</td>
<td>66</td>
<td>56.90%</td>
</tr>
<tr>
<td>Containing both of the above two types</td>
<td>23</td>
<td>19.83%</td>
</tr>
<tr>
<td>Others</td>
<td>2</td>
<td>1.72%</td>
</tr>
<tr>
<td>Total</td>
<td>116</td>
<td>100%</td>
</tr>
</tbody>
</table>

**B. Detection Method**

Based on above discussion, we design a method to detect malicious Web pages with abnormal visibility recognition. First, get the HTML codes of Web pages and parse it to resolvable structures, e.g. linked lists of tags. Second, for the pages containing browser-end scripts, the scripts will be extracted and executed with an interpreter; the output of execution in HTML format will be integrated with the other parts of original Web page and be passed to HTML parser as a whole. Final, check the tags attributes related to display effect according to abnormal visibility fingerprints. For example, detector will calculate the width and height values of related tags (*iframe*) of target Web pages based on parsing results and compare it with a threshold value. If the width or height value is less than the threshold, we consider that there is an abnormal visibility in the Web page and it would be regarded as a possible malicious page.

Compared to traditional signature matching, our method has some remarkable advantages as follows:

- The number of abnormal visibility modes is relatively fewer and their fingerprints are more stable than the malicious codes signatures. Therefore, the maintenances of the abnormal visibility fingerprint library are much easier. Besides, this method is able to detect the unknown malicious codes whose signatures have not been constructed.
- Due to the method only have to examine a few attributes values of the HTML tags instead of matching with a huge number of virus signatures, our method is efficient compared to the methods based on the virus signatures greatly.
- Our method can record the location of the tags when parsing the Web pages so as to locate the malicious codes in the source codes accurately. It makes the clearance to the malicious codes more convenient.

**IV. PROTOTYPE SYSTEM**

We design and implement a prototype of malicious Web pages detection system based on abnormal visibility recognition. The architecture of the prototype system is depicted in Fig. 4. The system consists of four main components: Web spider, HTML parser, JavaScript engine, and abnormal visibility detector.

**A. Spider**

In order to automatically detect the malicious Web pages in Web sites, a spider is implemented with Java language to fetch page content automatically. The spider will start from an initial URL; catch the HTML source code of the page and extract link URLs. These link URLs are stored in a queue; the spider will get a successive link URL as the new target resource link from the queue. The contents gained by spider will be feed to HTML parser.

Besides, the information in the Deep Web space is 400 to 550 times larger than the publicly indexable Web [10]. Malicious Web pages may also exist in the Deep Web space. For a higher coverage, the spider can send necessary authorization information to access authorization required Web pages, e.g. authorization cookie, user name and password of basic authorization protocol, etc.

**B. HTML Parser**

Either when the spider extracts the URLs or when the detector inspects abnormal visibility, a well-formed structure representing the tags in HTML pages is needed. The system implements a HTML parser based on a Java HTML parsing library [11]. For detecting malicious tags and crawling in Deep Web, we extend the library and implement some new classes for identifying key tags. For example, a new class IFrameTag is implemented for parsing *iframe* tags. With the parser, the HTML codes will be converted to some resolvable linked list structure of tags, which makes spider and detector can identify and analyze the HTML codes.

**C. JavaScript Engine**

As shown in Fig. 3, it is necessary for malicious pages detection to interpret the JavaScript code in HTML pages. Rhino is a JavaScript interpreter class library integrated in Java.
6.0 [12], but it can’t support the operations to browser built-in objects. Browser-end scripts will use these built-in objects to output execution result. For this reason, in our JavaScript engine, some simulation built-in objects are constructed to collect the output results of scripts, e.g. document, window, location, etc. These simulation objects provide the basic execution environment for the operations of scripts used to generate malicious codes dynamically. Our JavaScript engine consists of a Rhino-based JavaScript interpreter and some necessary simulation built-in objects. As depicted in Fig. 5, the scripts extracted from original Web page will be fed to JavaScript engine; the interpreter will interpret and execute them. When the scripts need to output some data to browser, e.g. HTML codes generated, the built-in objects will receive the data like a real browser. After interpreting, the output data collected by built-in objects will be integrated with HTML codes of original Web page and passed to parser.

![Figure 5. JavaScript Engine](image)

**D. Detector**

A detector is used to detect abnormal visibility. The system determines whether a tag is abnormal by matching its attributes values and content with the abnormal visibility fingerprints. For plain HTML codes, the detector will check the width and height attributes of related HTML tags directly. For JavaScript scripts, detection will be performed to the tags in the results of interpretation and execution. The width and height values will be compared with a threshold. If they are less than the threshold, we consider that there is an abnormal visibility in the Web page and it would be regarded as a possible malicious page.

To calculate the values of width and height attributes set in percentage format, we define a generalized display screen area as a calculation baseline whose area is 1027*768 pixels. For example, if the width setting of an iframe tag is 0.1%, the corresponding width value would be 1.024 pixels. If the threshold is 5 pixels, the tags will be considered abnormal. Besides, if the display style of the iframe tags is set to "display: none", the width and height values will be regarded as zero.

The detection will be performed according to some abnormal visibility fingerprints. So far, the prototype system has three kinds of fingerprints as follows:

1) Abnormal width or height.
2) Abnormal display: none display style.
3) Abnormal iframe generated by scripts.

In practice, some Web sites may introduce invisible iframe tags for normal application functions. To avoid false positive, the detector only reports the invisible iframe tags whose srcs link to external sites.

**V. EXPERIMENTS**

Using the prototype system, we detect 60 Web sites that were reported to have malicious codes by StopBadWare.org. From the result of the detection shown in Table II, we can see the detection system with comparatively low false positive and false negative.

**TABLE II. EXPERIMENT RESULTS FOR 60 WEB SITES**

<table>
<thead>
<tr>
<th>Item</th>
<th>Prototype System</th>
<th>Sucop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scanned Pages</td>
<td>66882</td>
<td>30561</td>
</tr>
<tr>
<td>Malicious Pages</td>
<td>30561</td>
<td>804</td>
</tr>
<tr>
<td>False Negative</td>
<td>804 (2.63%)</td>
<td>605 (1.99%)</td>
</tr>
<tr>
<td>False Positive</td>
<td>605 (1.99%)</td>
<td>0</td>
</tr>
</tbody>
</table>

Besides, for accurate effectiveness and efficiency testing, we construct an experimental Web site and set 14 suspected pages in it, include 9 malicious iframe tags, 3 malicious scripts (2 encoded) and 2 non-malicious pages. The results of detection using the prototype system and Sucop are shown in Table III. From the experiment, we can see that our method is superior to Sucop on performance and with lower false negative. Besides, all three malicious pages generated by scripts can not be detected by Sucop. Although two non-malicious pages are regarded as malicious by the prototype system, the result is acceptable. In most cases, false negative is more serious than false positive.

**TABLE III. EXPERIMENT RESULTS FOR EXPERIMENTAL WEB SITES**

<table>
<thead>
<tr>
<th>Item</th>
<th>Prototype System</th>
<th>Sucop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>0.95s</td>
<td>31.5s</td>
</tr>
<tr>
<td>Report Pages</td>
<td>14</td>
<td>8</td>
</tr>
<tr>
<td>False Positive</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>False Negative</td>
<td>0</td>
<td>6</td>
</tr>
</tbody>
</table>

In addition, several local Web sites are constructed for performance testing. According to results, we can see that the average time consuming of every link is only 0.03s.
Based on above three experiments, the prototype system is proved to have the following advantages:

1) The system offers high performance.
2) Based on abnormal visibility recognition, the system can detect almost all the Web pages containing malicious codes.
3) This system can detect the malicious codes generated by scripts, especially the encoded scripts.

Besides, the system can provide accurate row and column location information of detected malicious codes.

VI. CONCLUSION

Based on the analysis and statistics of Web malicious codes, we introduce the concept of abnormal visibility and propose a corresponding detection method to detect malicious Web pages effectively and efficiently. A prototype detection system is implemented based on abnormal visibility recognition. The experiments show that the system can detect a Web site with high performance, and can detect almost all the Web pages containing malicious codes. The system can be used to monitor the security state of target Web sites and provide security alarm for end users before visiting malicious Web pages.

In the future, we will improve and refine the abnormal visibility fingerprints to avoid false negative as much as possible.

REFERENCES