Malicious Websites Detection and Search Engine Protection

Hao Zhou, Jianhua Sun, and Hao Chen

***Abstract*—With the development of the Internet, the amount of information is expanding rapidly. Naturally, search engine becomes the backbone of information management. Nevertheless, the flooding of large number of malicious websites on search engine has posed tremendous threat to our users. Most of exiting systems to detect malicious websites focus on specific attack. At the same time, available browser extensions based on blacklist are powerless to countless websites. In this paper, we present a lightweight approach using static analysis techniques to quickly discriminate malicious sites comprising malware, drive-by-download and phishing sites. We extract comprehensive features to classify labeled dataset using various machine learning algorithms. Large scale evaluation of our dataset shows that the classification accuracy reaches 97.5% with low overhead. Furthermore, we achieved a chrome plugin to detect malicious search result websites based on our classification model.**

***Index Terms*—Malicious websites, feature extracting, machine learning.**

I. Introduction

The World Wide Web has become an essential way for us to obtain information. Among billions of websites, we are eager to find the appropriate one. At the same time, search engines provide an interface to index related information and have the engine digging out the most relevant. Actually, search engine drives 70% [1] of traffic to most web sites. Unfortunately, large number of malicious websites indexed by search engine.

If a user accesses the malicious website through search engine with no perception, malicious scripts usually launch attacks to install rogue program, steal personal identities and credentials, or even control the victim's machine as a part of botnet for further attack. In view of the malicious websites, prevention in advance is more valuable and preferable than handling after infected. It's very important to detect search result pages before visiting it.

Hence, some security agencies have developed various browser plugins to protect users from accessing unsafe websites. These site ratings are based on tests conducted by using an army of computers that seek out all kinds of threats. They can only offer modest accuracy since any database is limited and websites are updating rapidly.

In this paper, we focus on detecting malicious (malware, drive-by download and phishing) websites, and we designed a browser plugin running on Google Chrome since it's the most widely used web browser. We extract comprehensive features for each webpage containing URL, HTTP response header, content, domain information, search engine, pagerank, WOT score and Alexa information. To accurately and quickly determine which page is harmful or useful, we classify all web pages into two categories. Using various classification algorithms in Weka that is a popular suite of machine learning software, our classification accuracy reached 97.5%.

Manuscript received November 25, 2012; revised March 17, 2013.

The authors were with Hunan University, Changsha, CHINA. (e-mail: zhouhao6278@yahoo.com.cn; jhsun@aimlab.org; and haochen@aimlab.org)

II. Related Work

In this section, we review some detect methods which are adopted to discriminate malicious web pages. Furthermore, several related browser plugins will also be introduced.

A. Dynamic Method

Dynamic detecting system consists of high- and low-interaction honeyclient. High-interaction honeyclient monitors the state changes of real system. When a browser is running, it analyzes the potential malicious behavior, such as file download, abnormal process initiation, redirection, and illicit registry read and write [2]. In general, high-interaction honeyclients such as Wepawet [3] need longer time and more system resources, correspondingly higher accuracy. Comparing with high-interaction system, in low-interaction honeyclients, emulated browser is usually used to send HTTP requests to the web server, and signature-based or heuristic approach are employed to determine the malicious sites [4], [5].

Both high- and low-interaction systems load the web page through a browser (actual or emulated). After opening a web page, all linked resources will be retrieved by HTTP protocol, and all dynamic scripts will be executed. Redirection (if exists) will increase the cost of dynamic analysis. Since our goal aims at protecting search engine, long time is unacceptable, we are apt to employ static approach.

B. Static Method

Static approaches are based on features which behave differently in regular and malicious web pages. John P. John et al. [6] proposed a method to detect malware by poisoning search results, the features are history-based keywords change in URL, lexical features of a URL and webpage structural similarity. J. Ma *et al.* [7] detect phishing or spam websites through URL, host-related features. Jun Ho Huh *et al.* [8] detect phishing using popular search engines. Christian Seifert *et al.* [9] analyze the underlying DNS and web server relationships to detect drive-by download attack. Davide Canali *et al.* [10] combined previous work extracting HTML

DOI: 10.7763/JACN.2013.V1.52

and JavaScript features to detect drive-by download exploit. Birhanu Eshete *et al.* [11] first detect overall malicious websites, including malware, phishing, drive-by download and injection pages. URL, Page-Source and social reputation features are extracted. After extracting the features, machine learning algorithms are used for classification. These works are impractical with single classification. Inevitably, they work as a pre-filtering before dynamic analysis. On the basis of the previous studies, we extract a lot of new features for a webpage. Taking into account accuracy and response time on browser plugin, we finally select HTTP head, URL, content, domain information, pagerank, search engine results, WOT score and Alexa information as our classification features.

C. Available Search Engine Protection Plugins

Web Security Guard and SiteAdvisor, according to their respective database, warn users before access a malicious website.

WOT [12], relying on given score by invited users, shows five different icons next to the search results.

Millions of users benefit from these plugins every day. A wide range of users do not means without blemish. A common defect in these plugins is the limited database and generally not the same result. There is still a large number of websites not included in their databases, they have to show uncertain icon.

III. System Overview

To effectively prevent the attack of search engine poisoning, we designed a browser plugin running in Google chrome. The overview of our system is shown in Fig. 1.

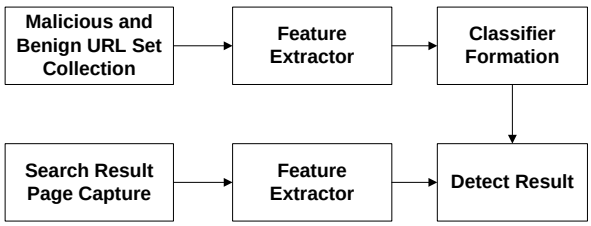


Fig. 1. System architecture

Our work focuses on how to avoid users accessing malicious websites on dominant search engine. Exiting plugins are not enough to cope with countless websites. Through manual analysis of a variety of pages, we find that the features which we have extracted are different between benign and malicious websites.

A. Data Collection

Since our approach depends on machine learning, dataset must be sufficient. Many malicious websites from public blacklist can't be accesses currently, so we gathered malicious websites from several security agencies.

1) Benign URLs

We collected 22014 benign URLs from three following resources. Web Security Guard, containing both benign and various malicious URLs, randomly obtained URLs from Yahoo's directory, and Alexa top 500 websites.

2) Malicious URLs:

We collected 14696 malicious websites from some public announced malware and phishing websites, like malwaredomainlist.com, malwaredomains.com, cleanmx.de, websecurityguard.com, www.mwsl.org.cn, hosts-file.net, malwareblacklist.com, webscan.360.cn, zeustracker.abuse.ch, www.mwsl.org.cn and phishtank.com.

B. Feature Selection

The features we selected are static, any dynamic code within a web page is not executed. All resources linked to the page are retrieved by Jsoup [13]. Jsoup is a Java library that parses HTML to the same DOM as modern browsers do. We use it to retrieve the response of a website and get desired information through parsing the HTML. In this work, we selected 42 special features to discriminate malicious web pages.

1) URL features

A URL represents uniform resource locator, also known as web address. Some information embodied in malicious URL is obviously inequable. For instance, an IP address or port exists in hostname. Hostname has an unusual large number of dots and its length is longer than ordinary one. Three or more consecutive identical letters appear in malicious hostname. All sort of symbols (such as '&', '%', '_') and numeric character in URL account for a large proportion.

2) HTTP response header field and content features

HTTP response header and content via the URL request is an essential part to identify malicious URL, so we can effectively prevent cloak and redirection from evading inspection.

In benign URLs, the header field contains Connection and Expires frequently. The Connection header field prohibit communicated by proxies over further connections. It efficiently cut off the long link. Expires header field is used to control the browser cache time. Conversely, if a malicious URL tries to employ HTTP redirection to migrate users to another exploit server, Content-Location header field is used to supply the resource location. In order to keep connection alive after the request is issued, Keep-Alive header field would appear in malicious URL. We do not use other HTTP header field.

Due to the intension of malicious web developers, they adopted more skills in designing webpage. We found the following features in HTML content.

a) The number of website in title element and number of Meta element

For the sake of indexing by search engine, malicious pages utilize many targeted websites in title element and keywords in Meta element. The refresh tag in Meta element and cross-domain redirection is also extracted.

b) The number of invisible elements and script

We found most of malicious websites are inclined to employ invisible elements. It is common to see the hidden input element, as well as invisible elements such as div, iframe, span, and audio. Malicious scripts are generally embedded in these invisible elements to conceal their genuine intention.

c) The number of suspected elements and with small dimension

Some of elements such as iframe, frame, embed, object are used to include external resources. Applet element is also extracted since these small applications can be executed locally.

Apart from these features, the body element and Meta element in body element are also counted. Malicious website developers attempt to redirect users to their deliberately designed webpage with just a few lines of dynamic script in head element. We calculate the ratio of head element length in the entire page. Generally, the ratio is relatively high in malicious URLs.

3) JavaScript features

JavaScript as a client-side scripting language was developed for interacting with the user. In spite of browsers providing some security mechanisms such as sand-boxing, it is inevitably compromised by malicious JavaScript code. In order to escape the static analysis, obfuscated code even encryption strategy are customarily emerged in malicious scripts.

In the following, we introduce the JavaScript features which are habitually exploited. Considering that the time spent on content matching is substantial, we extract the following 13 features: the JavaScript element number, the entropy of JavaScript in the page, the entropy of external JavaScript scripts in all scripts, the longest string length of src attribute of external JavaScript, the maximum length of all scripts, URL number in inline scripts, the rest of them are described in detail below.

a) JavaScript redirection

JavaScript is a quite versatile scripting language, and JavaScript redirection is the hardest to be detected among all redirection techniques through static analysis. We extract the frequently-used method including document.location, document.href, window.location, and location in JavaScript code. JavaScript redirection is rarely encountered in legitimate site.

b) The number of dynamic function

Dynamic functions interpret the embedded JavaScript code. These function calls are eval, document.write, and document.writeln.

c) The number of dynamic function

In malicious scripts, abundant objects are created using var, new, function or document.createElement.

d) The Number of obfuscation functions

Confusing functions in malicious scripts can reduce the risk of being detected, we pick up these functions such as concat, split, replace, encode, fromCharCode, unescape and substring.

e) The number of timer functions

The occurrences of the setInteval () and setTimeOut () functions are separately used for setting the execution interval and delay time. Using these functions in malicious scripts can easily evade the detecting technology based on execution if detecting time is not enough.

f) The Number of Event Functions

This feature records the common event triggering functions and event monitoring functions. We count the total number of calls of the following functions: onerror, onload, onunload, onbeforeload, onbeforeunload, onmouseover, addEventListener, attachEvent, dispatchEvent and fireEvent. These function calls can be utilized to realize drive-by download.

g) The number of suspicious functions or strings

We observed some other suspicious functions or strings are also encountered in malicious script, so we count the number of the following function: Math.random, insertBefore, ActiveXObject, innerHTML, Ajax and navigator.

4) Search engine and pagerank features

Google search engine is the most influential and popular search engine. According to pagerank technique, it checks the entire network link structure to decide the importance of the website. Comprehensively considering the importance and relevance associated with the specific query, most correlative and reliable result will be listed on the top. We use Google's advanced search function to obtain the results. For a given URL, such as www.malware.com.br/lists.shtml, we extracted domain name as malware.com.br, before the domain name, character string site: and link: was separately added to fetch the search result number. Site query mode get site number indexed by Google, if no search result or the website does not appeared in the first result, we count the number as zero. Link query mode inquire site number consisting of all having link to this website. At the same time, Google pagerank value as a great criterion to evaluate a website. These features are extracted through Google query.

5) Domain features

Since the domain registering process will be uniformly recorded by registration service provider, we can obtain some important information via accessing www.domaintools.com. We record the following features: the existence of the title, the ratio of the title relevancy, the description relevancy and the keyword relevancy, the ratio of search engine optimization, the number indexed by DMOZ, the ratio of outbound chain, the number included by Wikipedia, the ratio of missing images, the survival time of the domain. These features are highly persuasive arguments, it offers a detailed and authoritative judgment about the domain.

6) Alexa features

Alexa is a professional corporation which releases world ranking and provides traffic information aiming at the target website. We extract the following features for every domain. Relevant link count, access speed, average page views, back link number and presence of ranking. For a malicious website, some of these features are inexistent. Because the malicious websites are unpopular, these features can quickly distinguish between popular and unpopular one.

7) WOT features

WOT (web of trust) is a free add-on that maintains a database of known malicious and safe sites reported by the WOT community. It shows a colored traffic light next to the search engine results, email and other sites. When you attempt to visit a known malicious website in search results, WOT will warns you using red icon in advance. Similarly, green icon and yellow icon respectively stand for good and risky site.

These icons are based on the score which is marked by all experienced visitors. We are inspired from this security extension and find a mass of sites which are not evaluated by users. Exceptionally, malicious site manager may manipulate the score. In order to make the detection more accurate, we use it as a part of our features.

IV. Classifier Formations

In this section, we perform the evaluation of our feature selection. Malicious URL detection can be viewed as a classification problem, and the general process contains extracting features from both labeled benign and malicious URLs. For a labeled URL the feature vector like (V_1, V_2, V_i, …, V_n, C) where V_i represents the feature value, n is the total number of features, and C is the category containing malicious and benign. After getting the training dataset, training model is constructed using different classification algorithms, and then model applied to classify testing dataset and unknown URL if the evaluating accuracy aim to testing dataset is acceptable.

A. Evaluation

Our experimental system runs on a Pentium dual-core 3.20GHz CPU with 32-bit windows XP operating system and 2GB of memory installed.

TABLE I: Dataset Used for Training and Testing

Application	Benign	Malicious	Total
Training	11007	7,348	18,355
Testing	11007	7,348	18,355

TABLE I: Detailed Performance Indicator for Different Classification

Classifier	Accuracy	False Positive Rate	False Negative Rate
Naïve Bayes	88.3	0.138	0.097
Logister	95.0	0.069	0.032
J48 Decision Tree	95.7	0.034	0.018
Random Tree	96.6	0.032	0.016
Random Forest	97.5	0.027	0.025
LibSVM	92.4	0.103	0.048
Bayes Net	90.0	0.103	0.098

We obtain the entire feature vector through Jsoup which we introduced before. The average time spent on extracting features is about 3-4 seconds. After the dataset is generated, we use the Weka machine learning platform to train our dataset with 10-fold cross validation. In order to balance the dataset between training and testing set, we divide it into two equal parts for a given website as shown in Table I. A number of supervised learning algorithms such as J48, Random Tree were evaluated and three crucial evaluation parameters (Accuracy, False Positive Rate and False Negative Rate) were recorded as shown in Table II. Accuracy refers to the right predicting ratio for specified type of testing sample. False positive rate refers to the ratio between malicious websites misjudged as benign websites and all testing sites. Similarly, false negative rate indicates that the percentage of benign websites misjudge as malicious among all testing set. From the table, it can be seen that random forest algorithm performs best. Through the observation of false classified samples, we found that most of them are phishing websites since they own less script relative to other malicious websites. At the same time, relevant link count in Alexa information as the root of Random Tree is the most apparent feature to separate malicious and benign websites. We are committed to guiding users accessing search engine webpage, labeling benign websites as malicious is tolerated, malicious websites misjudge as benign is not acceptable. Our detecting result is extremely reducing the risk to be threatened by malicious website since the false positive rate is less than 0.05%. The generated model of random forest algorithm is slightly better than random tree but excessive decision trees is not good for rapidly detect search engine results. We select the random decision tree as the final detection method in our plugin since it provides a detailed and concise evaluation measure.

B. Discuss Feature Robustness

Some of malicious websites may use cloaking or redirection [14], [15] techniques to pollute search results. Cloaking is a cheating method that delivers different content for search engine crawler and browser, we obtain the page content through HTTP request rather than crawler to simulate the actual click behavior, which makes cloaking incur no effect to our analysis. All types of redirection features are catched, we just can't precisely ensure the existence of JavaScript redirection due to obfuscated code. We are confident to detect it because obfuscated JavaScript features are extracted and our dataset containing a lot of JavaScript redirection instances. Malicious webpage especially for phishing might be designed to imitate popular website, however, we can also detect it according to its Alexa and domain information.

V. Plugin Implementation

In this section, we describe the implementation of our plugin developed for avoiding users to visit malicious search engine websites on Google Chrome. Chrome provides integrated API for developers to implement extension which makes surfing Internet more convenient and desirable.

Next, we describe the implementation of our search engine plugin. In the plugin, content script used to interact with web page, background script is used to send the HTTP request and get desired features. First, content script monitors the document event DomContentLoaded. Once the event occurs, from the visitorial URL, we can see whether it contains search engine query, and if so, all returned search links are fetched. Next, a connection is created between content script and background script using chrome.extension.connect function, content script sends the links to background script then waits background script to return detecting results. For each

received searching link, background script extracts previous introduced features through accessing several websites including the link itself. All response through asynchronous XMLHttpRequest are string object, HTML object is created by createElement function to parse it into a DOM tree. Corresponding to the classifier formation, identical features are grabbed after XMLHttpRequest response status equals 200. When background script completes the feature extraction for a search link, previous generated random tree algorithm is used to determine that the search link is malicious or not. Good or bad detecting result sends to content script using postMessage function. Based on the received message, content script appends green or red icon next to the link using innerHTML function. Especially for an unable visited website, a gray icon will be shown on the right side of the link. Some links are invalid in search engine results, it can notify the users not to click these links again. Generally, these differently colored icons are not presented in order due to the discrepant response speed for each link.

As above, the implementation procedure of our plugin are described, a plenty of search tests show that the performance overhead of our plugin is in the range of 3 to 6 seconds to analyze all the search result pages. Compared to existing plugins based on black and white list, our plugin is not fast enough since most of the time is spent on obtaining third-part resources. For the purpose of improving the detection accuracy, a few seconds is acceptable since users are more willing to get safe and credible search results. Our plugin as a good safeguard for search engine is implemented on mainstream search engines like Google, Bing and Yahoo.

VI. Conclusion

Malicious websites pose significant threat to Internet security, and effectively preventing user from accessing these sites becomes vitally necessary. Exiting solutions to detect malicious websites mainly focus on specific attack types. Meanwhile, exiting plugins is defective with limited database. We described a static approach based on classification algorithm and extracted plentiful features for each webpage consisting of URL, HTTP response header, content, domain information, search engine, pagerank, WOT score and Alexa information. Much of features are firstly introduced to detect all kinds of malicious websites. Our experiment demonstrates that the selected features greatly improve the classification performance and are robust to combat potential malicious websites. The browser plugin we developed is a good safeguard for search engine poisoning and is no longer relying on a limited database like exiting plugins.

References

[1] Reality SEO. (2006). [Online]. Available: http://realityseo.com/Google_refers_70_percent.html

[2] M. T. Qassrawi and H. Zhang, "Detecting malicious web servers with honeyclients," in *P*roc. *Journal of Networks Conf.*, Wuhan, 2011, pp. 145-152.

[3] Wepawet. (2012). [Online]. Available: http://wepawet.cs.ucsb.edu

[4] M. Cova, C. Kruege, and G. Vigna, "Detection and analysis of drive-by-download attacks and malicious javascript code," in *P*roc. *19th International Conf. World Wide Web*, Raleigh, 2010, pp. 281-290.

[5] L. Lu, R. Perdisci, and W. Lee, "SURF: detecting and measuring search poisoning," in *Proc. 18th ACM Conf. Computer and Communications Security*, Chicago, 2011, pp. 467-476.

[6] J. P. John, F. Yu, Y. L. Xie *et al*., "DeSEO: Combating search-result poisoning," in *Proc. 20th USENIX Conf. Security*, San Francisco, 2011, pp. 20-35.

[7] J. Ma, L. K. Saul, S. Savage *et al*., "Learning to detect malicious URLs," in *Proc. ACM Trans. Intelligent Systems and Technology*, vol. 2, pp. 328-471, April 2011.

[8] J. H. Huh and H. Kim, "Phishing Detection with popular search engines: simple and effective," in *Proc. 4th Canada-France MITACS Conf. Foundations and Practice of Security*, Paris, 2011, pp. 194-207.

[9] C. Seifert, I. Welch, P. Komisarczuk *et al*., "Identification of malicious web pages through analysis of underlying DNS and web server relationships," in *Proc. 33th IEEE Conf. Local Computer Networks, Montreal*, 2008, pp. 935-941.

[10] D. Canali, M. Cova, and G. Vigna, "Prophiler: a fast filter for the large-scale detection of malicious web pages," in *Proc. 20th International Conf. World Wide Web*, India, 2011, pp. 197-206.

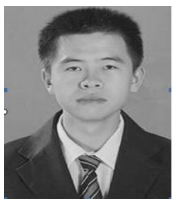
[11] B. Eshete, A. Villa_orita, and K. Weldemariam, "BINSPECT: holistic analysis and detection of malicious web pages," in *Proc. 8th ICST Conf. Security and Privacy in Communication Networks*, Padua, 2012, pp.149-166.

[12] Safe Browsing Tool, WOT. (2012). [Online]. Available: http://www.mywot.com

[13] Jsoup. (2012). [Online]. Available: http://jsoup.org

[14] B. Wu and B. D. Davison, "Cloaking and redirection: a preliminary study," in *Proc. AIRWeb Conf.*, Chiba, 2005, pp. 7-16.

[15] K. Chellapilla and A. Maykov, "Taxonomy of javascript redirection spam," in *Proc. 3rd International Workshop Conf. Adversarial Information Retrieval on the Web*, Banff, 2007, pp. 81-88.



Hao Zhou was born in Changsha, China, year 1988. He received the Bachelor degree in communication engineering from XiangTan University in 2010. Now he is toward the Master degree in Information and communication engineering, Hunan University, China. His research interest is Web Security.

Jianhua Sun received the PhD degree in computer science from Huazhong University of Science and Technology, China, in 2005. She is an associate professor at College of Information Science and Technology, Hunan University, China. Her research interests are security and operating system.

Hao Chen received the PhD degree in computer science from Huazhong University of Science and technology, China, in 2005. He is an associate professor at College of Information Science and Technology, Hunan University, China. His research interests include security, virtual machines, operating systems, distributed and parallel computing. He is a member of the IEEE.