IJETIE International Journal of Emerging Technology and Innovative Engineering
Volume I, Issue 4, April 2015
ISSN: 2394 - 6598
www.ijetie.org

A Collaborative Work For Browser Security Through Plug-In Features

Sagar P.Kolhe
Sir Visvesvaraya Institute of technology, Nashik
Rocking09sagark@gmail.com

Rohan Vadnere
Sir Visvesvaraya Institute of technology, Nashik
rohanvadnere@gmail.com

PRITAM VADHE
Sir Visvesvaraya Institute of technology, Nashik
Pritam.wadhe1992@gmail.com

MANGESH NARKHEDE
Sir Visvesvaraya Institute of technology, Nashik
mangeshn@live.in

PROF.K.N.SHEDGE
Assistant Proffessor
Sir Visvesvaraya Institute of technology, Nashik
kishorshedge2007@gmail.com

ABSTRACT

Many contemporary web sites incorporate third-party content in the form of advertisement, social networking widgets and maps. While such interweaving of codes from multiple sources often enhanced the user experience, the web site may not always trust the source of third-party code. Moreover due to proliferation of ad-networks and content distribution network, the true source of content may be hidden behind multiple levels of indirection. This leads to illegal indexing of web pages too. We indent to provide few plug-in features as in for spamdexing and sandboxing. Most of the exiting study of web spam detection explicitly or implicitly performs offline on search engine side. The online web spam detection is also useful for some different scenario. We implement the plug-in for online scenario. in that we get three types of labeling data. We collected and adopted that data for learning a reliable web spam classifier. Moreover, a sandboxing approach is also providing for web pages having personal and confidential information exchange. To get rid of the ads, a feature will also provide in plug-in. The main objective of plug-in is to increase the browser efficiency and performance.

Keywords

Spamdexing, Sandboxing, Plug-In

1. INTRODUCTION

1.1 Spamdexing

Spamming the search engines (or spamdexing) is the practice of using unethical or unprofessional techniques to try to improve search engine rankings. You should be aware of what constitutes spamming so as to avoid trouble with the search engines. Also known as search engine spam or web spam is the deliberate manipulation of search engine indexes. Its intended to mislead search engines into ranking some pages higher than they deserve. Practice of including information in a web page such that indexing satisfies the spamdexer but dissatisfy the search engine providers and users.

The problem arises when site operators load their Web pages with hundreds of extraneous terms so search engines will list them among legitimate addresses. The process is called "spamdexing," a combination of spamming the Internet term for sending users unsolicited information and indexing."Spamdexing is the practice of search engine spamming. It is a form of Search Engine Optimization (SEO) spamming, which is the art of making a website attractive to the major search engines for optimal indexing. The goal of a web designer is to create a web page that will find favorable rankings in the search engines, and they create their pages according the standards that they believe will help. Some of them resort to spamdexing, often unbeknownst to their clients. While spamdexing has interfered with the finding of information on the internet, measures have been taken to curb it with some success. Spamdexing was a big problem in the 1990s, and search engines were fairly useless because they were compromised by spamdexing. Once Google came on the scene that all changed Google developed a page ranking system that fought against spamdexing quite well, discounting spam sites and awarding true, relevant websites with high page rankings.

1.2 Sandboxing

A sandbox is a testing environment that isolates untested code changes and outright experimentation from the production environment or repository,in the context of software development including Web

development and revision control [2]. Sandboxing protects ”live” servers and their data, vetted source code distributions, and other collections of code, data and/or content, proprietary or public, from changes that could be damaging (regardless of the intent of the author of those changes) to a mission critical system or which could simply be difficult to revert. Sandboxesreplicate at least the minimal functionality needed to accurately test the programs or other code under development.

While JavaScript was originally designed for adding small scripting capabilities to Web pages, its uses have become dramatically more sophisticated during the past 15 years. JavaScript content on Web pages today provides rich functionalities such as auto filling of form fields, re-acting to user events (like key presses and mouse clicks), asynchronously interacting with the server, providing social-networking support, etc., which together lead to an enhanced user experience. Furthermore, a large amount of JavaScript content these days comes from third-party sources such as advertisement By further analogy, the term “sandbox” can also be applied in computing and networking to other temporary or indefinite isolation areas, such as security sandboxes and search engine sandboxes (both of which have highly specific meanings), that prevent incoming data from affecting a “live” system (or aspects thereof) unless/until defined requirements or criteria have been met.

2. RELATED WORK

2.1 Spamdexing

Spamdexing, generally refers to any deliberate actions that are “intended to mislead search engines into ranking some pages higher than they deserve” compared to other properly indexed wed. Driven by the huge benefits of increasing/changing rankings of websites in the search results, website owners prefer adopting various web spam tricks to quickly achieve their goal rather than devoting their money, time and efforts to consolidate the content and quality of their websites [1]. The large amounts of spam pages on the web not only pollute the search results in search engines, but also dissipate many resources for search engines and many commercial competitor companies. Detecting spam pages on the web is emerging in the past several years. Many detection methods have been proposed to combat spamdexing. As search engines are the main victim of web spam, most of existing web spam detection methods explicitly or implicitly assume that the detection process is conducted on the search engine side. If a web page is labeled as “spam” by search engines, it is permanently removed from the indexes of pages maintained by those search engines, and thus will never be shown in any search results. Generally, the web spam detection on the search engine side is considered to be an offline process which is done on server side. However, our recent studies indicate that an online detection of web spam is also useful in specific scenarios due to several reasons. First, whether a page is spam or non-spam is in fact subjective, and differs on user’s choice which is been not consider in any past web spam detection algorithms.

2.2 Sandboxing

In concern of sandboxing, The Web browsers which are one of the main sources of information retrieval from the web, have much vulnerability. Though these browsers have private browsing feature like in Chrome, Firefox and Internet Explorer but they are still prone to attacks as stated in [4]. They describe the flaws that are extant in browsers even after having private mode, which the user imagines is secure. They describe attacks prone to a local attacker and a web attacker who can access personal data even if the user uses private browsing mode. These flaw points the use of browser extensions leaving trace of websites visited, on the disk which can be accessed by an attacker. Hence even if any user makes use of private browsing features by browsers, they are still not secure. But this gives an insight that web URL’s can be accessed in both normal and private mode of the browsers [4]. Each web browser differs in its extensible architecture and working. Extensions depend on the architecture, whether they can be developed or not. For example, from the following list of browsers, Internet Explorer, Firefox, Chrome and Safari, only Safari doesn’t support extensions. [4]. An extension called BROWSERSPY was developed, which did not require any special privileges but still it managed to take complete control over the browser and observe all activity performed through the browser staying undetectable. Extensions can be harmful but at the same time helpful. Therefore, there can be both security oriented and security hindering extensions existing in the extensions market [3].

3. PROPOSED SYSTEM

3.1 Spamdexing

Core idea is to build multiple web spam classifiers using different training data then adopt ensemble learning techniques to combine multiple classification results three web spam classifiers: The first classifier is constructed based on collected us spam judgments. The collected data come from us explicit and implicit spam labels. The data are easy to collect using the implemented web browser plug-in. The second classifier is constructed based on collected data from other users who share similar spam judgments of u. This classifier is extremely useful when collected us

data set is too small to build a reliable web spam classifier. The last classifier is constructed using collected data of remaining users.

We believe this classifier provides crowd intelligence, and it is important to address the well-known cold start problem. Once three different classifiers are built, an ensemble learning framework is adopted to provide ultimate spam labeling results. We also want to emphasize that conducting online web spam detection can complement the existing studies about offline web spam detection. The two web spam detection processes are conducted at different stages. After search engines detect and filter out many spam pages offline, an online web spam detector can still function so as to further refine personalized web spam detection results.

3.1.1 Collecting Spam Labeling Data

The web browser plug-in collects Self Data [1] from two different sources;the one is explicit judgments and the other implicit judgments.

1) To collect explicit judgments from users, the web browser plug-in provides an integral interface for users to explicitly submit spam labels.

a) If a user is browsing a page and he/she does not think the page is spam, the user can click the interface to upload his non-spam judgments.

b) If the user thinks the page is spam, the user can also upload the labeled result to the central cloud server.

2) Users implicit judgments of spam can also be collected when they are surfing the web.

a) For example, if a page is labeled as spam/non-spam by the web browser plug-in and the user does not update the spam detection result, it implicitly means the user agrees with such spam judgments.

b) In addition, all the pages in user's bookmarks are implicitly treated as non-spam, as users bookmark a page if they think the page is useful and they will view the page in the future.

3.1.2 Ensemble Learning of Online Web Spam Detection

In our framework, the collected Self Data, Peer Data, and Public Data capture different spam judgments. It is necessary to integrate them into a unified model. We adopt ensemble learning techniques as existing studies have shown that using multiple models can obtain better predictive performance than using any single ones. Main methods used will be

1. Majority Rule
2. Average Rule
3. Weighted Rule

3.2 Sandboxing

We describe the design, implementation and evaluation for a secure web browser browsing n securing data on the web form pages. Web Forms is an ASP.NET feature that you can use to create the user interface for your Web applications. Web Forms pages offer you a powerful and straightforward programming model that uses familiar rapid application development (RAD) techniques to build sophisticated Web-enabled user interfaces. When data that has been entered into HTML forms is submitted, the names and values in the form elements are encoded and sent to the server in an HTTP request message using GET or POST. Forms are usually combined with programs written in various programming language to allow developers to create dynamic web sites. The most popular languages include both client-side and/or server-side languages.

Although any programming language can be used on the server to process a form's data, the most commonly used languages are scripting languages, which tend to have stronger string handling functionality than programming languages such as C, and also have automatic memory management which helps to prevent buffer overrun attacks[8].

User generally used this web form pages for various proposes, where the data entered by the user is crucial. As this data entered by the user is very crucial and must not get leaked to the third party. This question's the web browser security, thus we provide a architecture as in where any web form page will be opened in a different session thus by providing security to the entered information on web form pages. In our proposed system no history for this session will be created. Thus there will b no flaw pointing towards browser extensions that leaving trace of websites visited, on the disk which can be accessed by an attacker. Similarly we are taking care of the cookies which are a small piece of data sent from a website and stored in a user's web browser while the user is browsing that website. Every time the user loads the website, the browser sends the cookie back to the server to notify the website of the user's previous activity. This my record the user's browsing activity (including clicking particular buttons, logging in, or recording which pages were visited by the user as far back as months or years ago). This is not a ideal case for a secured web browser which we will be taking into consideration.

In our proposed system the user will be provided will the save or flush option where the in case of save option all the cookie data will be saved in a encrypted format which user can access it later whenever in future he access the same page with a simple predefined (by user itself) 3 digit pin whenever he tried to access the same. For example, if the user has typed his credit card details and decide to save option at the end of the session, then the details will be saved in encrypted format, now when user again access any web form pages with the credit card details, the access to that page will after authenticating through the 3 digit

IJETIE
International Journal of Emerging Technology and Innovative Engineering
Volume I, Issue 4, April 2015
ISSN: 2394 - 6598
www.ijetie.org

pin, otherwise it will restart to completely new session all together.

Similarly if user chooses the flush option then all the history along with cookies will b deleted from the user's computer disk.

4. DATA ANALYSIS

The classification of known and unknown sites worked well for cases 1, 2 and 5. The websites for case 3 and 4 were known by hardly any participant. In contrast, the classification of unknown sites matched very well. Thus, the following results will mostly be taken from the unknown websites [1]. For the well-known websites, we found a tendency of them being rated nearly equally to the unknown sites whilst the plug-in seemed to have more influence on people using unknown sites.

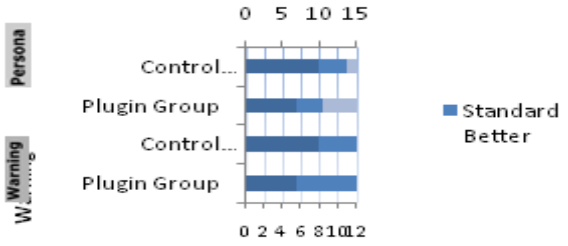

Fig .1: Classification of known and unknown sites worked

Comparing the five cases, the SSL Persona-enhanced browser outperformed the standard one. The medians and means of the participants' answers ranging from -2 to +2 can be found in table 1. In case of correctly SSL-secured sites, people in the plug-in group rated trustworthiness, the willingness to log in and the ability to determine security higher. These values support H1.

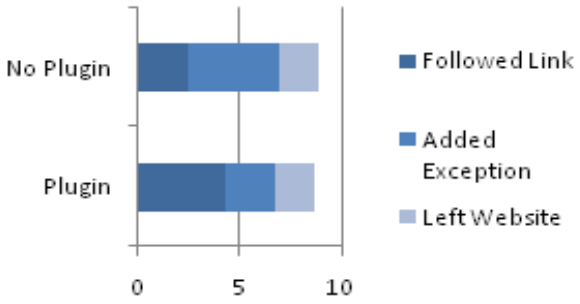

Fig. 2: Comparison between plug-in and non plug-in browsers

For the partially encrypted page using the warning Persona, we expected to get lower ratings for trust and log in willingness. The ability to determine security should still be higher. Again all those assumptions hold when looking at the median values. This supports
H2.

In case of the non-SSL sites and the phishing sites, both groups saw a standard any Persona. For H3, we expected the results of the plug-in group to drop due to a missing positive feedback. This did not happen. The study duration was much too short for people to get used to the plug-in and expecting the green Persona to show up. Furthermore, the order of the visited websites was randomized for each participant.

Looking at the warning page for a non-matching URL, participants had three options to choose from: Leaving the site, visiting the link the certificate was intended for and adding an exception and continuing to the site. The best option in such a case would be to visit the URL the certificate was originally issued for. This was done by zero (known) and two (unknown) participants for the control group in contrast to six (known) and five (unknown) using the plug-in.

When confronted with an untrusted issuer certificate, the only option was to leave the site or set up an exception. Since our websites both were genuine but used self-signed certificates, setting up an exception was okay in this case. Again plug-in users used the exception more often. Comparing the correct answers with a repeated-measures ANOVA – within-subject factors: "known" and "warning type" – the results for the between subjects variable "group" are highly significant which confirms H4.

After rating the websites, the participants were debriefed and shown some side-beside images of a browser with and without the plug-in. Overall, 75% of the participants preferred the plug-in version [1]. The same holds for the comparison of the warnings. 71% preferred the plug-in warnings.

5. CONCLUSION

Web browsers provide user-friendly interfaces for users to interact with different Internet contents with flexible and rich plug-in environments. Due to their ubiquitous and flexible nature, we propose to evaluate the performance of cloud services in a browser-based network measurement platform. Delay is one of the fundamental performance metrics of Cloud Services. We developed a browser-based delay measurement tested over the Internet so that different delay measurement tools could be evaluated in the same real network environment. Our measurement results revealed the properties of the application layer delay over real Internet paths, and how these properties varied

from the underlying network layer delay. The objective of this dissertation is to develop techniques for solving the JavaScript sandboxing problem.

6. REFERENCES

[1] Cailing Dong, Bin Zhou, "An Ensemble Learning Framework for Online Web Spam Detection", 2013 12th International Conference on Machine Learning and Applications.

[2] Adam Barth, Collin Jackson, Charles Reis, Google Chrome Team "The Security Architecture of the Chromium Browser".

[3] ACCUVANT LABS security assessment and research, "Browser Security Comparison – A Quantitative Approach", Pages 140, Version 0.0, Revision Date: 12/6/2011.

[4] Aggarwal, G., Bursztein, E., Jackson, C., AND Boneh, D. An analysis of private browsing modes in modern browsers. In Proceedings of the 19th USENIX conference on Security (2010), USENIX Security'10.

[5] AlexandrosNtoulas, Marc Najork, Mark Manasse, Dennis Fetterly, "Detecting Spam Web Pages through Content Analysis", WWW 2006, May 23-26, 2006, Edinburgh, Scotland, ACM 1-59593-323-9/06/0005.

[6] Bennet Yee, David Sehr, Gregory Dardyk, J. Bradley Chen, Robert Muth, Tavis Ormandy, Shiki Okasaka, NehaNarula, and Nicholas Fullagar, "Native Client: A Sandbox for Portable, Untrusted x86 Native Code", 2009 IEEE Symposium on Security and Privacy.

[7] J. Abernethy, O. Chapelle, and C. Castillo. Web spam identification through content and hyperlinks. In AIRWeb'08,pages 41–44. ACM, 2008.

[8] Adam Barth, Collin Jackson, and John C. Mitchell.Robust defenses for cross-site request forgery. In 15th ACM Conference on Computer and CommunicationsSecurity (CCS), October 2008.

[9] C. Cortes and V. Vapnik. Support-vector networks. Machinelearning, 20(3):273–297, 1995.

[10] G. Jeh and J. Widom. Scaling personalized web search. InWWW'03, pages 271–279 ACM, 2003.

[11] P. Francis, S. Jamin, C. Jin, Y. Jin, D. Raz, Y. Shavitt, and L.Zhang, ID Maps: A global Internet host distance estimation service, IEEE/ACM Trans. Net., vol. 9, no. 5, pp. 525540, Oct. 2001.