Malicious Attack Detector

N.Jayakanthan[1],
[1](Department of Computer Applications, Kumaraguru College of Technology, Coimbatore E-mail:haijai2@gmail.com)
Dr.M.Manikantan
[2](Department of Computer Applications, Kumaraguru College of Technology,Coimbatore)

ABSTRACT

The taint URLs guide the client to suspicious websites which collects the users invaluable information and exploit their system. Here we propose an authentication based approach to detect such websites. We developed a tool called Analyzer an enhanced browser prevent the user from malicious attack. It analyze the website using EWLSVM (Ehnanced Weighted Least Square Twin Support Vector Machine) machine learning algorithm to find the website is malicious or not. The proposed approach is compared with existing approaches which reports low false positive and false negative. The experimental approach shows the proposed approach correctly detects all phishing and genuine website without any false positive and negatives. It overcomes many drawback of the existing signature based approaches.

Keywords: *Malware, Classifier, SVM*

---

1 Introduction

The goal of phishing attacks is to steal user identities and credentials. Phishing is the act of attempting to acquire information such as usernames, passwords, and credit card details (and sometimes, indirectly, money) by masquerading as a trustworthy entity in an electronic communication. Communications purporting to be from popular social web sites, auction sites, online payment processors or IT administrators are commonly used to lure the unsuspecting public. Phishing emails may contain links to websites that are infected with malware[1]. Phishing is typically carried out by e-mail spoofing or instant messaging and it often directs users to enter details at a fake website whose look and feel are almost identical to the legitimate one. Phishing is an example of social engineering techniques used to deceive users and exploits the poor usability of current web security technologies. Attempts to deal with the growing number of reported phishing incidents include legislation, user training, public awareness, and The goal of phishing attack

Unfortunately, no existing technical mechanism fully solves this problem. For example, SSL only authen- ticates a web server's IP address or hostname to a browser and protects the communication channel as well. Nonethe- less, it provides no guarantee the HTML files sent by the web server are not misleading. [2]Some schemes are proposed to enable a user to authenticate a server with a priori secu- rity association. However, those schemes are not applicable to websites which a user visits for the first time. Moreover, it requires user awareness of existence of the authentication Nonetheless, if the user is already alerted, a simple URL checking can prevent the phishing attack. (HTTP Transaction) A request sent from the browser to the server and the corresponding response from the server to the browser, both. When a phishing site maliciously claims a false identity, it always demonstrates abnormal behaviors compared to a honest site, which are indicated by some web DOM objects in the page and HTTP transactions.

The advantage of this approach includes that it does not rely on any prior knowledge of the server or users' security expertise, the adversary has much less adaptability since the detection is independent of any specific phishing strategy and it causes no changes on users' existing
navigation behavior.

Given the rising threat posed by malicious web pages, it is not surprising that researchers have started to investigate techniques to protect web users. Currently, the most widespread protection[3] is based on URL blacklists. These blacklists (such as Google Safe Browsing) store URLs that were found to be malicious. The lists are queried by a

browser before visiting a web page. When the URL is found on the blacklist, the connection is terminated or a warning is displayed. Of course, to be able to build and maintain such a blacklist, automated detection mechanisms are required that can find on the Internet web pages containing malicious content.

Unfortunately, for obvious reasons, very few details have been revealed about Google's filter. In particular, the authors only provide examples of three page features and report that they use a proprietary machine-learning framework. The very existence of Google's blacklist provides evidence that the overall system i.e. combining the filter with the back-end analysis tools works.

2.Related Work

In the last few years, the detection of web pages that launch drive-by-download attacks has become an active area of research and several new approaches have been proposed. Dynamic approaches. Dynamic approaches use honeyclient systems to visit web pages and determine if they are malicious or not. In high-interaction honeyclients, the analysis is performed by using traditional browsers running in a monitored environment and detecting signs of a successful drive-by-download attack (e.g., changes in the file system, the registry, or the set of running processes) [4,7,9,10]. In low-interaction honeyclients, the analysis relies on emulated browsers whose execution during the visit of a web page is monitored to detect the manifestation of an attack (e.g., the invocation of a vulnerable method in a plugin) [5,6,8, 11]. Both high- and low-interaction systems require to fully execute the contents of a web page. This includes fetching the page itself, all the resources that are linked from it, and, most importantly, interpreting the associated dynamic content, such as JavaScript code[12]. These approaches usually yield good detection rates with low false positives, since, by performing dynamic analysis, they have complete "visibility" into the actions performed by an attack. The down-side is that this analysis can be relatively slow, because of the time required by the browser (either simulated or real) to retrieve and execute all the contents comprising a web page, taking from a few seconds to several minutes, depending on the complexity of the analyzed page. Scalability issues with today's honeyclient systems (relatively slow processing speed combined with relatively high hardware requirements) motivated our work on a filtering system[13]. Our filter achieves higher performance by forgoing dynamic analysis (e.g., the interpretation of JavaScript code), and relying instead on static analysis only. Static approaches [14] to the detection of drive by download attacks rely on the analysis of the static aspects of a web page, such as its textual content, features of its HTML and JavaScript code, and characteristics of the associated URL.

3.Methodology

The proposed approach using various features like domain , sub domain name ip address and special character to analyze the existing website. The machine learning algorithm the Support Vector Machine (SVM) is used for this purpose.

Algorithm

Algorithm 1 Support Vector Machine
Set SV = { closest pair from opposite classes }
while there are violating points do
Find a violator
Set SV = Set SV S
violator
if any $\alpha p < 0$ due to addition of c to S then
set SV = set SV \ p
repeat till all such points are pruned
end if
end while

Our simulation program is written in Java and run on a Sun T-2000 server, which contains an 8-core 1 GHz Ultra SPARC CPU, 16 GB of RAM, and the Solaris 10 operating system. The *dataset* is stored in a *mysql* relational database.

The Website analyze the features of the website like domain, sub domain, and malicious special characters . If the website features match the features in the profile then the website detected is reported as malicious other the website is declared as genuine. According to performance analysis in normal case and attack case our tool reported low false positive and negative rate.

4. Implementation and Environment

We implemented Phishfilter, and used to analyze various websites.. This tools are also used the current Twitter, Google, and Wikipedia. Our tools also applied for various search engines. ur tool is analyzed 2,000 URLs per day. The Phishfilter fetches pages which analyzes each page and extracts and stores all the features. Once all features have been extracted from a page, uses the models learned in the previous training phase to evaluate its maliciousness. If a page has been identified as likely malicious, It report it as malicious.

5. Conclusion

The proposed approach eliminates need for a moderator to analyze the features of the website manually. It detect where the website of the given URL is genuine or malicious automatically. Initially our tool is trained using 300 genuine and malicious URLs. In testing phase our tool is capable enough to detect malicious and genuine websites with low false positive and negative rate the experimental results shows our approach is efficient in detecting malicious attacks.

Reference

[1] C. Estan, S. Savage, and G. Varghese, "Automatically inferring patterns of resource consumption in network traffic," in *Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications (SIGCOMM)*, 2003, pp. 137–148.

[2] "CERT Incident Note IN-99-07," 1999. [Online]. Available: http://www.cert.org/incident notes/IN-99-07.html.

[3] "Planetlab testbed," http://www.planet-lab.org.

[4] M. Locasto, J. Parekh, A. Keromytis, and S. Stolfo, "Towards Collaborative Security and P2P Intrusion Detection," in *Proceedings of the 2005 IEEE Workshop on Information Assurance and Security*, 2005.

[5] "CERT Advisory CA-2003-04 MS-SQL Server Worm," 2003. [Online]. Available: http://www.cert.org/advisories/CA-2003-04.html.

[6] Y. Hu, D. M. Chiu, and J. C. Lui, "Adaptive Flow Aggregation - A New Solution for Robust Flow Monitoring under Security Attacks," in *Tenth IEEE/IFIP Network Operations and Management Symposium (NOMS)*, 2006, pp. 424–435.

[7] J. Jung, B. Krishnamurthy, and M. Rabinovich, "Flash crowds and denial of service attacks: characterization and implications for CDNs and web sites," in *Proceedings of the 11th International Conference on World Wide Web (WWW)*, 2002, pp. 293–304.

[8] "CERT Advisory CA-1996-21 TCP SYN Flooding and IP Spoofing Attacks," 1996. [Online]. Available: http://www.cert.org/advisories/CA-1996-21.html.

[9] "CERT Advisory CA-2003-20 W32/Blaster worm," 2003. [Online]. Available: http://www.cert.org/advisories/CA-2003-20.html.

[10] S. Gibson, "Distributed Reflection Denial of Service - Description and analysis of a potent, increasingly prevalent, and worrisome Internet attack," 2002. [Online]. Available: http://www.grc.com/dos/drdos.html.

[11] S. Katti, B. Krishnamurthy, and D. Katabi, "Collaborating against common enemies," in *Internet Measurement Conference*, 2005.

[12] S. Axelsson, "The Base-Rate Fallacy and Its Implications for the Difficulty of Intrusion Detection," in *ACM Conference on Computer and Communications Security*, 1999, pp. 1–7.

[13] G. K. Bhattacharyya and R. A. Johnson, *Statistical concepts and methods*. New York: Wiley, 1977.

[14] D. Karger, E. Lehman, T. Leighton, R. Panigrahy, M. Levine, and D. Lewin, "Consistent hashing and random trees: distributed caching protocols for relieving hot spots on the World Wide Web," in *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, 1997, pp. 654–663.

[15] "Dshield org," http://www.dshield.org.

[16] C. V. Zhou, S. Karunasekera, and C. Leckie, "Evaluation of a Decentralized Architecture for Large Scale Collaborative Intrusion Detection," in *the Tenth IFIP/IEEE International Symposium on Integrated Network Management (IM), Germany*, 2007, pp. 80–89.