ISSN:0975-9646

Blacklisting Sites and Detecting Web Spams

Ranbir Singh , Bhupender Bhadana

Department of Computer Science
B.S.A.I.T.M,
Faridabad, India

***Abstract*-The paper aims at blacklisting websites and detecting web spasm using IP Cloaking. Malicious Spyware are causing a significant threat to desktop security and are playing with the integrity of the system. The misuse of websites to serve exploit code to compromise hosts on the Internet has increased drastically in the recent years. Many approaches to tackle the problem of spam have been proposed. Spamming is any deliberate action solely in order to boost a web page's position in search engine results, incommensurate with page's real value. Web Spam is the Web pages that are the result of spamming. Web spam is the deliberate manipulation of search engine indexes. It is one of the search engine optimization methods. The paper provides an efficient way that prevents users from browsing malicious Web sites by providing a service to check a Web site for malignity before the user opens it. Hence if a Web site has been reported to be malicious, the browser can warn the user and suggest not visiting it.**

Keywords: DHT protocol, IP cloaking, spam detection.

I. INTRODUCTION

Internet has become a major source of Information Retrieval in recent times as the amount of information is growing on the internet. This increase in information has raised a major threat as more and more criminal minds try to exploit it for their needs. Internet crime has become a dangerous threat to both home users and companies. According to the Internet Crime Complaint Center, the amount of complaints linked to Internet fraud hit a new record in 2008 by causing a total loss of $265 million. The fact that this number almost quadrupled in only four years demonstrates that cyber crime rates are rising and the need for protection against it is higher than ever [1].

As security in server based applications is increasing, attackers have started to target client side applications, such as the web browsers or document readers. As these applications are installed on almost every host they make a valuable target for an attacker. In order to get people to visit specially prepared websites that exploit current web browser vulnerabilities, links are advertised using email SPAM. Other methods include blog comments, guestbook entries, twitter, or messages distributed across social networks as done by the Koobface worm [2].

This problem can be rectified by aggressive filtering of email SPAM. But SPAM filters can only tackle the distribution of malicious URLs through email and not to other distribution paths.

As the popularity of the search engines is growing over the years, the problem Web Spam is also arising. Web Spam are nothing but spam indexing or search spam, or search engine spam i.e. when we search for a query in the search engines it gives results based on query. Web spam can be very dangerous from user's perspective. Spam site can contain malware, when user open the site the malware silently get installed on the system. The site can also affect the financial status by stilling the private information like bank account number, password and other financial information. Becchetti et al. [3], performs a statistical analysis of a large collection of Web pages. In particular, he computes statistics of the links in the vicinity of every Web page applying rank propagation and probabilistic counting over the entire Web graph in a scalable way. He builds several automatic web spam classifiers using different techniques. Egele et al. [4] introduce an approach to detect web spam pages in the list of results that are returned by a search engine.

In a first step, Egele et al. [4] determines the importance of different page features to the ranking in search engine results. Based on this information, he develops a classification technique that uses important features to successfully distinguish spam sites from legitimate entries. By removing spam sites from the results, more slots are available to links that point to pages with useful content. Additionally, and more importantly, the threat posed by malicious web sites can be mitigated, reducing the risk for users to get infected by malicious code that spreads via drive-by attacks. A feature is a property of a web page, such as the number of links pointing to other pages, the number of words in the text, or the presence of keywords in the title tag. To infer the importance of the individual features, black-box testing of search engines was performed. More precisely, he creates a set of different test pages with different combinations of features and observes their rankings. This allows us to deduce which features have a positive effect on the ranking and which contribute only a little.

II PROBLEM FORMULATION

Not only has the amount of crime on the Web risen over the years, but also the types of attacks have changed significantly. While phishing emails and malicious attachments were the major infection vectors in the past, so called drive by-downloads on malicious Web sites now form the overwhelming majority of Web-based attacks [36]. That is, Internet users' workstations get infected with malicious software (malware) without their knowledge by simply browsing a compromised Web site. The malware installed on the user's workstation is mostly designed to either steal information such as bank account data or passwords, or can be used by the attacker to control a botnet. Especially in 2007-2008, more trojan programs were developed and distributed via Web sites than ever before. In fact, the virus analysts of Kaspersky Lab believe that the number of malicious Web sites and malware programs this year will even exceed the one from 2008 [37].

Given these facts, it is crucial to protect the users' workstations from being infected. Many organizations developed software and invented defence techniques against those attacks. However, most solutions such as anti-virus protection or software based firewalls are rather reactive and leave security updates to the user.

IP cloaking is a black hat method of gaining higher rankings in search engines by showing the spiders a different page of content that the user sees. It works by having a script on your server and when a page request comes to the server the HTTP header is checked to see where the request is coming from. If the request is coming from a search engine then a different page is presented than the normal one. This page will be purely for the search engine and will be highly optimized only for this purpose.

The need of proposed system is to detect spam and to identify malicious Web sites via a remote URL Blacklist as shown in Figure 2. The end-user clients in this scenario are common Web-browsers such as Firefox, Safari or the Internet Explorer.

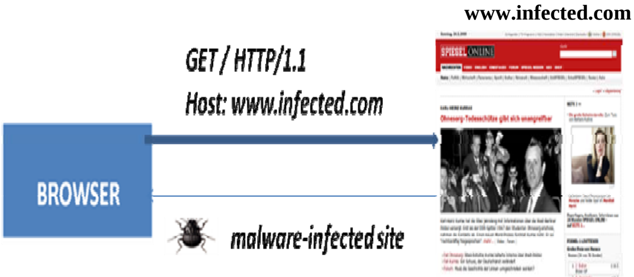

Figure 3: Problem description

III ARCHITECTURAL STATEGY

The proposed architecture for detection of spam and how to identify malicious Web sites via a remote URL Blacklist is shown in Figure 4.Blacklisting Service is based on a P2P network of interconnected nodes. Each node is an equal part of a distributed hash table and only stores the blacklist entries it is responsible for. The application splits in two major parts: The core network and the blacklisting service. While the Kademlia-based core network, called KadS, implements a trusted DHT and acts as data store, the blacklisting service is built on top of the DHT and simply uses the distributed storage. As Figure 4 shows, the core network consists of several KadS nodes. Each node stores a small part of the DHT in its local hash table and keeps track of other nodes in the local routing tree. In order to make the DHT accessible, it furthermore provides an API to add, alter and delete entries. The KadS network only provides a P2P-based storage environment, but does not validate the data it stores. For the blacklist service, however, a fixed data-structure is required: The blacklist nodes represent the actual user-accessible service and solve this problem by enforcing a business-logic as well as a fixed data structure for the hash table values. Each node encapsulates a KadS node and implements a secure TCP/UDP server to communicate with browser- and honeyclients.

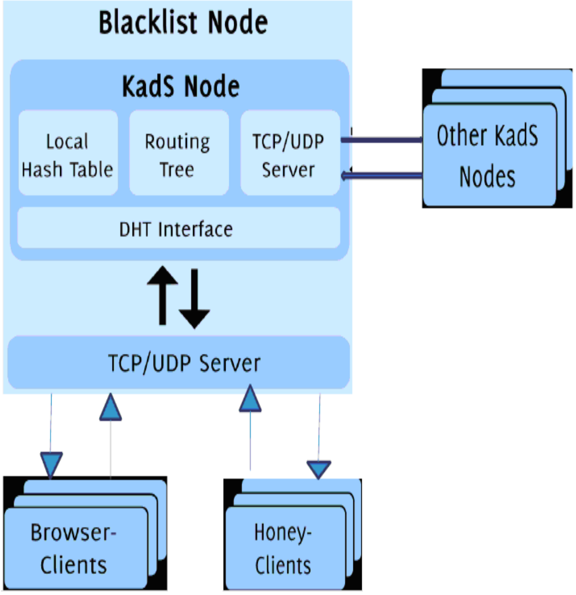

Figure 4: Proposed Architecture

A KadS: Core Network

The core network is an access-restricted, secure distributed hash table. Its design is based on a modified version of the Kademlia DHT protocol and extends it to a fully encrypted public key infrastructure. While the basic operations are almost identical to Kademlia, KadS restricts the access to the network on the one hand, and enforces nodes to encrypt their communication on the other hand. In order to do so, nodes need to authenticate in a handshake procedure before DHT-related messages can be exchanged. Only if they successfully verified each other's identity, they are able to exchange messages using a symmetric session-key. Thus, the major differences of KadS to the Kademlia protocol are access restriction and communication encryption.

B Blacklist Service

As Figure 4 shows, the blacklist service is wrapped around the core system and uses its interface to store and retrieve blacklist entries. In fact, it simply uses the secure DHT protocol as database and enforces a specific data-type for the hash table keys and values. To make the blacklist entries available to clients, it furthermore provides two different interfaces for browser plug-ins and the honeyclients. That is, the underlying core system provides almost all functionality while the actual blacklist service only uses its infrastructure for secure distributed data provisioning. In fact, the core system could be used for several different purposes at the same time as long as the on-top services such as the blacklist service generate differentiating hash table keys.

In order to receive and/or store entries in the DHT, each blacklist node needs to be connected to the KadS network. For this purpose, each of them encapsulates a KadS node and additionally provides outside interfaces for the blacklist clients. That is, the actual blacklist service does not provide any DHT-related functions, but simply uses the KadS node's methods to access the distributed hash table. Querying the network for the blacklist entry of the domain example.com, for instance, is nothing more but a simple call to the KadS node's get-method.

IV PROPOSED MODEL

The proposed work is detection of spam and how to identify malicious Web sites via a remote URL Blacklist. The framework of our proposed model is shown in Figure 5. The detail of each part in the model is illustrated below:

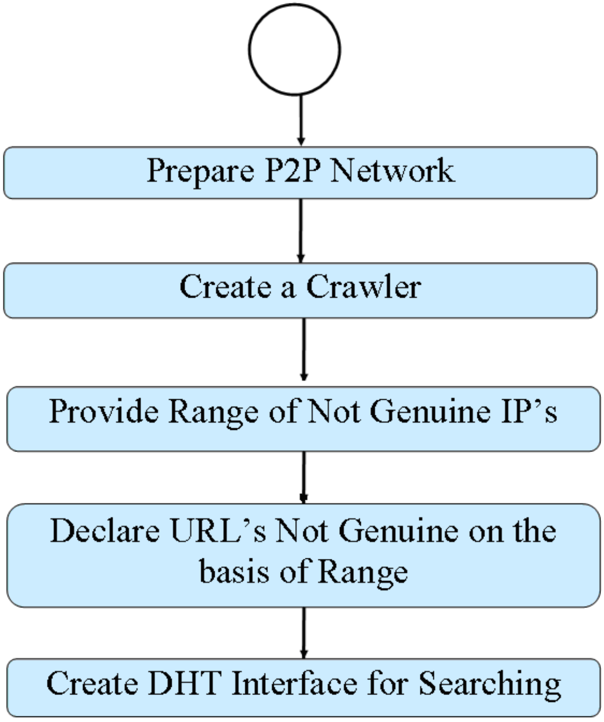
Prepare P2P Network
Create a Crawler
Provide Range of Not Genuine IP's
Declare URL's Not Genuine on the basis of Range
Create DHT Interface for Searching

Figure 5: Proposed Model

A. Prepare P2P Network

In this step generally creating a peer-to-peer network, in this there are a number of nodes (WebPages or Website) that create a network called P2P network. In this network when any node wants to join a network, there is a certificate authority that is designed by the network, provides the certificate to that node and after authorization that node join the network. In this network every node contains a certificate and public private key to encrypt or decrypt the message and local hash table to communicate to corresponding nodes. The advantages of creating a P2P Network are:

1. No single point of failure/attack: Due to the lack of a central server, it is more difficult for attackers to disrupt the service provided by the P2P network. Most P2P systems are designed to be redundant and the failure of few peers does not affect the service quality. In fact, P2P services mostly are more reliable and fault tolerant than client-server systems [29].

2. No resource bottleneck: In client-server based systems, a lack of resources such as processor time or memory shortage is more likely to occur. P2P networks distribute resources of interest equally amongst the participating peers and each node uses resources of the others.

3. Scalability and flexibility: In order to provide a flexible environment, P2P networks allow peers to join and leave the network as they like. Hence, if the network reaches a peak in terms of resource usage, one can simply add new peers to scale the application and balance the load among all peers.

B. Create a Crawler

In this step, a crawler is designed that is to used to crawl the website and provide the information to the P2P network, it generally collects the information of the domain name and it's regarding website/WebPages and sends it to the network.

C. List of IPs

In the list of IPs, there are two Types of IPs exist: 1). Genuine IP Address. 2). Non Genuine IP Addresses. This differentiation is based on the information that is collected by the crawlers. Crawler sends the information regarding IP Addresses then check that IP Address in the list,

D. If that IP Address comes in from the genuine IP Address then this will be accessed by the user & if this comes in the non genuine IP Addresses then it will harm your computer. All this information of non genuine IP addresses is stored in the database.

E. Create DHT Interface

To retrieve the information from the database, DHT interface is created through which the browser client can access the information through UDP and TCP servers.

Structured peer-to-peer systems mostly focus on providing a distributed, content-addressable data storage". Instead of identifying resources via their network location, the system is designed to store the content itself at a specific position in the network. This so called Distributed Hash Tables (DHT) has many advantages. Not only are they more fault-tolerant and reliable than unstructured approaches, they also outperform them in terms of scalability and performance.

V CONCLUSION

With the advancement of Internet rapidly, more and more criminal minds try to exploit it for their needs. Internet crime has become a dangerous threat to both home users and companies. Thus, there is a need for tools which can guarantees the Availability, Confidentiality and Integrity of the Information exchanged. The proposed approach is successfully Detecting Spam and identifying malicious Web sites via a remote URL Blacklist. The approach examined malicious spyware that are causing a significant threat to desktop security and are playing with the integrity of the system. The approach suggested prevents users from browsing malicious Websites by providing a service to check a Web site for malignity before the user opens it. Hence if a Web site has been reported to be malicious, the browser can warn the user and suggest not visiting it. In contrast to the obvious solution to realize the service on a classic client-server basis, the proposed system design uses a secure distributed hash table (DHT) to reduce the load of single systems and to be more resistant against denial-of-service attacks, or general failures.

The well-known P2P-based protocol Kademlia has been extended to an access-restricted secure distributed hash table: The new PKI-supported DHT protocol KadS implements certificate-based authentication and encrypts the communication between participating nodes. While traditional P2P networks allow every node to join and communicate with one another, KadS nodes have to possess a valid CA-signed public key certificate and a matching private key to join the network. Therefore, KadS allows the creation of a trustworthy P2P network and can be used to store confidential information.

REFERENCES

[1] Internet Crime Complaint Center,"*IC3 2008 Annual Report on Internet Crime,* " 2009. URL http://www.ic3.gov/media/2009/090331.aspx

[2] J. Baltazar, J. Costoya, and R. Flores, "The Real Face of KOOBFACE: The Largest Web 2.0 Botnet".

[3] Luca Becchetti, Carlos Castillo, Debora Donato, Ricardo Baeza Yates, Stefano Leonardi, "*Link Analysis for Web Spam Detection*".

[4] Manuel Egele , Clemens Kolbitsch , Christian Platzer, "*Removing web spam links from search engine results,*" in Springer-Verlag France 2009

[5] Wei Wang , Guosun Zeng , Daizhong Tang, "Using evidence based content trust model for spam detection‖ in Expert Systems with *Applications,"* 37 (2010) 5599–5606, Science Direct.

[6] Jun-Lin Lin, "*Detection of cloaked web spam by using tag-based Methods,"* in Expert Systems with Applications 36 (2009) 7493–7499, Science Direct.

[7] Luca Becchetti, Carlos Castillo, Debora Donato, Stefano Leonardi, and Ricardo Baeza Yates, "*Link-based characterization and detection of web spam,"* In Proceedings of the 2nd International Workshop on Adversarial Information Retrieval on the Web (AIRWeb), 2006

[8] C. Castillo, D. Donato, A. Gionis, V. Murdock, and F. Silvestri "*Know your neighbors: web spam detection using the web topology,"* Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, pages 423–430, 2007.

[9] Alexandros Ntoulas, Marc Najork, Mark Manasse, and Dennis Fetterly, "*Detecting spam web pages through content analysis*," In Proceedings of the 15th International World Wide Web Conference (WWW), pages 83–92, Edinburgh, Scotland, 2006.

[10] Gilad Mishne, David Carmel, and Ronny Lempel, "*Blocking blog spam with language model disagreement*," In Proceedings of the 1st International Workshop on Adversarial Information Retrieval on the Web (AIRWeb), Chiba, Japan, 2005.

[11] A. A. Benczúr, I. Bíró, and K. Csalogány, "*Detecting nepotistic links by language model disagreement,"* In Proceedings of the 15th International World Wide Web Conference (WWW), 2006.

[12] Guang-Gang Geng, Chun-Heng Wang, Qiu-Dan Li, Lei Xu and Xiao-Bo Jin, "*Boosting the Performance of Web Spam Detection with Ensemble Under-Sampling Classification*".

[13] Manuel Egele, Christopher Kruegel, Engin Kirda, "*Removing Web Spam Links from Search Engine Results*".

[14] Lourdes Araujo and Juan Martinez-Romo "*Web Spam Detection: New Classification Features Based on Qualified Link Analysis and Language Models*," in IEEE Transactions on Information Forensics And Security, VOL. 5, NO. 3, SEPTEMBER 2010.

[15] Juan Martinez-Romo, Lourdes Araujo, "*Retrieving Broken Web Links using an Approach based on Contextual Information*".

[16] J. Abernethy, O. Chapelle, and C. Castillo, "*Webspam identification through content and hyperlinks*," in Proc. Fourth Int. Workshop on Adversarial Information Retrieval on the Web (AIRWeb), Beijing, China, 2008, pp. 41–44.

[17] András A. Benczúr, Károly Csalogány, Tamás Sarlós, Máté Uher , "*SpamRank – Fully Automatic Link Spam Detection Work in progress,"* in Proc. First Int. Workshop on Adversarial Information Retrieval on the Web (AIRWeb, Chiba, Japan, 2005, pp. 25–38

[18] Jay M. Ponte and W. Bruce Croft, *"A Language Modeling Approach to Information Retrieval"* in Proc. 21st Annu. Int. ACM SIGIR Conf. Research and Development in Information Retrieval (SIGIR'98), New York, 1998, pp. 275–281, ACM.

[19] C. Castillo, D. Donato, L. Becchetti, P. Boldi, S. Leonardi, M. Santini, and S. Vigna, "*A reference collection for web spam,‖ SIGIR Forum,"* vol. 40, no. 2, pp. 11–24, 2006.

[20] Glen Murphy. New Firefox extensions. 2005. URL http://googleblog.blogspot.com/2005/12/new-firefox extensions.html.

[21] Mozilla Foundation. Firefox Phishing and Malware Protection. 2009. URL http://www.mozilla.com/en-US/firefox/phishing-protection/.

[22] Apple Inc. Software License Agreement for Safari. 2009. URL http://images.apple.com/legal/sla/docs/SafariMac.pdf.

[23] ZDNet. Study: IE8's SmartScreen leads in malware protection. 2009. URL http://blogs.zdnet.com/security/?p=2981.

[24] Mozilla Foundation. Phishing Protection: Design Documentation. 2009. URL http://code.google.com/p/google-safe-browsing/wiki/Protocolv2Spec.

[25] Mozilla Foundation. Phishing Protection: Design Documentation. 2009. URL https://wiki.mozilla.org/Phishing_Protection:_Design_Documentation.

[26] MacWorld.com MacJournals.com. Inside Safari 3.2's anti-phishing features.2008. URL http://www.macworld.com/article/137094/2008/11/safari_safe_browsing.html.

[27] Microsoft Corporation. Principles behind IE7s Phishing Filter. 2005. URL http://blogs.msdn.com/ie/archive/2005/08/31/458663.aspx.

[28] Microsoft Corporation. IE8 Security Part VIII: SmartScreen Filter Release Candidate Update. 2009. URL http://blogs.msdn.com/ie/archive/2009/02/09/ie8-security-part-viii-smartscreen-filter-release-candidate-update.aspx.

[29] Ralf Steinmetz, "Peer-to-peer systems and applications," 2005.

[30] Robert Morris Ion, "Stoica. Chord: Ascalable Peer-To-Peer lookup service for internet Applications," 2001. URL http://pdos.csail.mit.edu/papers/chord:sigcomm01/chord_sigcomm.pdf.

[31] Kaspersky Lab, "*Kaspersky Security Bulletin: Statistics 2008,"*Issue Date: March 2009. URL http://www.viruslist.com/en/analysis?pubid=204792052.

[32] Kaspersky Lab, "*Kaspersky Security Bulletin: Malware evolution* 2008," Issue Date: March 2009.URL http://www.viruslist.com/en/analysis?pubid=204792051.