ISSN:0975-9646
Anuj Vujayrania et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 3 (2) , 2012,3313-3317

Application of DHT Protocol in IP Cloaking

Anju Vijayrania,Abdul Rahman
Department of Computer Science
Lingaya's University, Faridabad, India

***Abstract*-The paper aims at examining malicious spyware that 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 spamdexing 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. RELATED WORK

Related Work deals with detection of spam and how to identify malicious Web sites via a remote URL Blacklist. The end-user clients in this scenario are common Web-browsers such as Firefox, Safari or the Internet Explorer.

A. Identifying Spam

Wei Wang et al. [5] present use the notion of content trust for spam detection, and regard it as a ranking problem. Besides traditional text feature attributes, information quality based evidence is introduced to define the trust feature of spam information, and a novel content trust learning algorithm based on these evidence is proposed. Finally, a Web spam detection system is developed and the experiments on the real Web data are carried out, which show the proposed method performs very well in practice. Jun-Lin Lin et al. [6] Work presents three methods of using difference in tags to determine whether a URL is cloaked. Since the tags of a web page generally do not change as frequently and significantly as the terms and links of the

web page, tag based cloaking detection methods can work more effectively than the term- or link-based methods. The Proposed methods are tested with a dataset of URLs covering short-, medium- and long-term users' interest.

Experimental results indicate that the tag-based methods outperform term- or link-based methods in both precision and recall. Moreover, a Weka J4.8 classifier using a combination of term and tag features yields an accuracy rate of 90.48%.Becchetti et al [7] presents a study of the performance of each of these classifiers alone, as well as their combined performance. Using this approach he is able to detect 80.4% of the Web spam in our sample, with only 1.1% of false positives. Castillo et al. [8] demonstrate three methods of incorporating the Web graph topology into the predictions obtained by our base classifier:

1. clustering the host graph, and assigning the label of all hosts in the cluster by majority vote,
2. propagating the predicted labels to neighboring hosts, and
3. using the predicted labels of neighboring hosts as new features and retraining the classifier.

Ntoulas et al. [9] considers some previously undescribed techniques for automatically detecting spam pages, examines the effectiveness of these techniques in isolation and when aggregated using classification algorithms. Mishne et al. [10] follow a language modeling approach for detecting link spam in blogs and similar pages. They examine the use of language in the blog post, a related comment, and the page linked from the comment. In the case of comment spam, these language models are likely to be substantially different. Benczúr et al. [11] propose method fights a combination of link, content and anchor text spam. He catches link spam by penalizing certain hyperlinks and compute modified PageRank values. Guang-Gang Geng et al. [12] focuses on how to take full advantage of the information contained in reputable websites (web pages). Manuel Egele et al. [13] determine 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.

Lourdes Araujo et al. [14] present an efficient spam detection system based on a classifier that combines new link-based features with language-model (LM)-based ones. These features are not only related to quantitative data extracted from the Web pages, but also to qualitative properties, mainly of the page links. They consider, for instance, the ability of a search engine to find, using information provided by the page for a given link, the page that the link actually points at. Juan Martinez-Romo et al. [15] propose an algorithm based on information retrieval techniques to select the most relevant information and to rank the candidate pages provided for the search engine, in order to help the user to find the best replacement. Jacob Abernethy et al. [16] present an algorithm, witch, that learns to detect spam hosts or pages on the Web. Unlike most other approaches, it simultaneously exploits the structure of the Web graph as well as page contents and features. The method is efficient, scalable, and provides state-of-the-art accuracy on a standard Web spam benchmark.

Benczúr et al. [17] proposed a novel method based on the concept of personalized PageRank that detects pages with an undeserved high PageRank value without the need of any kind of white or blacklists or other means of human intervention. He assumes that spammed pages have a biased distribution of pages that contribute to the undeserved high PageRank value. He define SpamRank by penalizing pages that originate a suspicious PageRank share and personalizing PageRank on the penalties. Jay M. Ponte et al. [18] proposes aapproach significantly outperforms standard tf.idf weighting on two different collections and query sets. His component of a probabilistic retrieval model is the indexing model, i.e., a model of the assignment of indexing terms to documents. WEBSPAM-UK2006[19] collection, a large set of Web pages that have been manually annotated with labels indicating if the hosts are include Web spam aspects or not. This is the first publicly available Web spam collection that includes page contents and links, and that has been labeled by a large and diverse set of judges.

B. Identifying Malicious Web Sites Via A Remote URL Blacklist

Two existing solutions are: Google's Safe Browsing for Firefox/ Safari and Microsoft's SmartScreen for the Internet Explorer.

1. Google Safe Browsing as shown in Figure 1: Initially designed and developed by Google and distributed as part of the Google Toolbar, the former Safe Browsing extension [20] is now licensed under the Mozilla Public License and an essential part of Firefox and Safari [21, 22]. The component checks the sites a user visits against regularly downloaded lists of reported phishing and malware sites. If a URL or domain matches an entry in the list, a warning message is displayed to the user. In the newest version, it also supports live lookups with up-to-the-minute fresh lists for every URL instead of using the cached local versions. Since the protocol is well-structured and openly defined, the provided lists could come from any server that implements the system. However, due to its origin, both browsers use the Google servers by default. That is, the lists of phishing and malicious Web sites are maintained by Google which, according to ZDNet, uses a combinationof automatic (honey clients) and community-driven efforts to analyze a Web site" [23]. The protocol is based on simple HTTP request/response-cycles and supports blacklists as well as white-lists. It differentiates between malware-, phishing- and white-lists and supports various list formats, including regular expression lists or hashed lists of URLs or domains, respectively. It typically updates them every 30 minutes and usually only compares the visited URLs to the local lists. However, if a Web site matches a local list entry, it double-checks the URL using a live lookup to make sure that the entry is still up-to- date [24, 25].

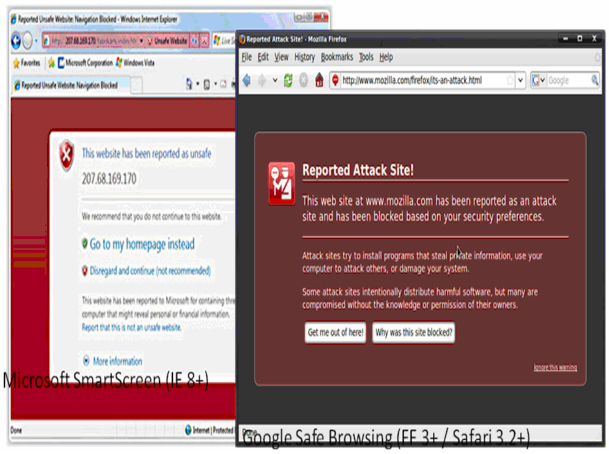

Figure 1: Google's Safe Browsing for Firefox/Safari and Microsoft's Smart Screen for the Internet Explorer

Surprisingly, Google's competitor Apple also uses the technology in its proprietary browser Safari. Apple silently included Safe Browsing in version 3.2 and only mentions it in its License Agreement [22, 26].

2. Internet Explorer Smartscreen as shown in Figure 1: Since version 7 of the Internet Explorer, Microsoft also included phishing protection in its browser. The so called Phishing Filter detects only phishing attempts, but does not protect users from drive-by downloads on malicious Web sites [27]. The recently released Internet Explorer 8 extends the Phishing Filter by the missing anti-malware protection and has been rebranded to SmartScreen [28].

Like its predecessor and in contrast to Google's Safe Browsing API, SmartScreen mostly relies on live lookups to determine if a Web site has been reported to be a phishing site or distributes malware. Although it also keeps a regularly downloaded list of known safe sites, it queries Microsoft's server for most of the visitedWeb sites. That is, SmartScreen and Phishing Filter only check \sites that aren't in IE's downloaded `known-safe' list" [27] and hence are able to use up-to-date information for most of the Web sites. In addition to the blacklist approach, SmartScreen also statically analyzes each visited Web site for characteristics associated with known phishing attempts and warns if sites are suspicious.

III 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.

www.infected.com

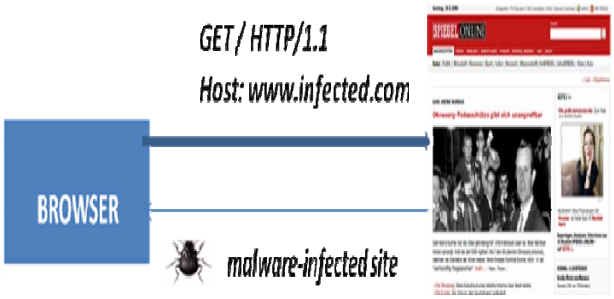

Figure 3: Problem description

IV PROPOSED MODEL

The inspiration of our 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 3. The detail of each part in the model is illustrated below:

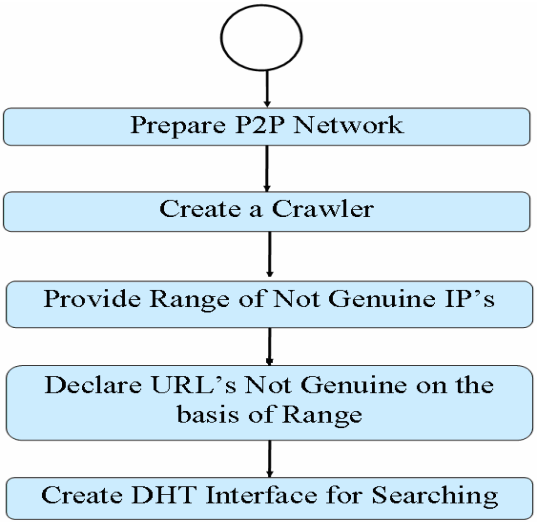

Figure 3: Proposed Model

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. Especially the latter differentiate the system from first-generation P2P networks. Since most operations of common DHT protocols have a complexity of O(log N) or O(log2 N), adding many peers to the network hardly changes the performance at all [29].

The first DHT protocols, i.e. Chord [30], Pastry [31], CAN [32], and Tapestry [33], were designed in 2001 when the research community realized their enormous potential. They all differ in data management and routing strategies, but essentially follow the general paradigm of consistent hashing [34]: In contrast to classic hash tables in which changing the number of array slots results in the recalculation of all hash- keys, consistent hashing allows resizing the table without having to change the keys. It was originally designed to solve the problem of a varying number of machines in a network and is now used by DHT protocols. The idea is to assign each node a k-bit identifier and divide the address space, typically

$\{0, 1\}^k$ for $k > 0$, in roughly equally sized segments (or buckets). Each node is assigned to a segment and is responsible for storing all the data items with hash values that fall within its assigned segment" [35]. This fixed structure makes it possible for peers to locate the responsible node(s) for a given key, and thus store or retrieve data items.

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.

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. URLhttp://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_browsi ng.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] P. Druschel A. Rowstron, "*Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems*," 2001. URL http://research.microsoft.com/~antr/PAST/pastry.pdf.

[32] Paul Francis Sylvia Ratnasamy, "A *scalable content addressable network,*" 2001. URL http://www.cs.cornell.edu/people/francis/p13- ratnasamy.pdf.

[33] John Kubiatowicz Ben Y. Zhao, "*Tapestry: An infrastructure for fault- tolerant wide-area location and routing*," 2001. URL http://cs- www.cs.yale.edu/homes/arvind/cs425/doc/tapestry.pdf.

[34] E. Lehman, D. Karger, "*Consistent Hashing and Random Trees: Distributed Caching Protocols for Relieving Hot Spots on the World Wide Web*," 1997. URL http://tinyurl.com/akamai-com.

[35] K. Borner, G. Fletcher, H. Sheth, "Unstructured Peer-to-Peer Networks: Topological Properties and Search Performance," 2005.

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

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