PHISHING E-MAIL FILTERING MECHANISM USING HEURISTIC TECHNIQUE

M.K.P.Madushanka and AL.Hanees

Department of Mathematical Sciences, Faculty of Applied Sciences
South Eastern University of Sri Lanka

Abstract

Phishing is a new type of network attack where the attacker creates a replica of an existing Web page to fool users (e.g., by using specially designed e-mails or instant messages) into submitting personal, financial, or password data to what they think is their service provides' Web site. In this research paper, I proposed a novel method to phishing email filtering by the use of end-host based anti-phishing algorithm, which is call LinkGuard and content based filtering by the use of knowledge discovery by utilizing the generic characteristics of the hyperlinks in phishing attacks. These characteristics are derived by analyzing the phishing data archive provided by the Anti-Phishing Working Group (APWG). Because it is based on the generic characteristics of phishing attacks, Link Guard can detect not only known but also unknown phishing attacks. Our experimental analysis verified that Link Guard is effective to detect andprevent both known and unknown phishing attacks with minimal false negatives. This research also showed that Link Guard is light weighted and can detect and prevent phishing attacks in real time.

Keywords: Phishing, Hyperlink, email classification, Link Guard

Introduction

Phishing is a new word produced from 'fishing', it refers to the act that the attacker allure users to visit a faked Web site by sending them faked e-mails (or instant messages), and stealthily get victim's personal information such as user name, password, and national security ID, etc. This information then can be used for future targetadvertisements or even identity theft attacks (e.g., transfer money from victims' bank account).

The frequently used attack method is to send e-mails to potential victims, which seemed to be sent by banks, online organizations, or ISPs. In these e-mails, they will make up some causes, e.g. the password of your credit card had been miss-entered for many times, or they are providing upgrading services, to allure you visit their Web site to conform or modify your account number and password through the hyperlink provided in the e-mail. If you input the account number and password, the attackers then successfully collect the information at the server side, and is able to perform their next step actions with that information (e.g., withdraw money out from your account).Phishing itself is not a new concept, but it's increasingly used by phishers to steal user information and perform business crime in recent years. Within one to two years, the number of phishing attacks increased dramatically. The rest of paper is organized as follows: Section 2 introduces the literature survey; Section 3 introduces the motivation for this research and Section 4 analyze the system architecture; Section 5 shows the results and Section 6 discuss the conclusion and further work and finally attached the references for this research.

A phishing technique was described in detail as early as 1987, in a paper and presentation delivered to the International HP Users Group Interex.[1]. Email filters are a commonly applied defense mechanism impeding malicious links from reaching the potential victims' inboxes. These are typically use statistical techniques (DSPAM, SpamAssassin, *etc.*), URL blacklists and sender email information to identify spammed emails.

Phishing life cycle

A fake webpage generally contains a login form, and when a user opens the fake webpage and inputs personal information, this information is accessed by the attacker. Furthermore, the attackers use this information for some personal and financial gain [2]. The life cycle of a phishing attack is shown in Figure 1 .The following steps are involved in a phishing attack:

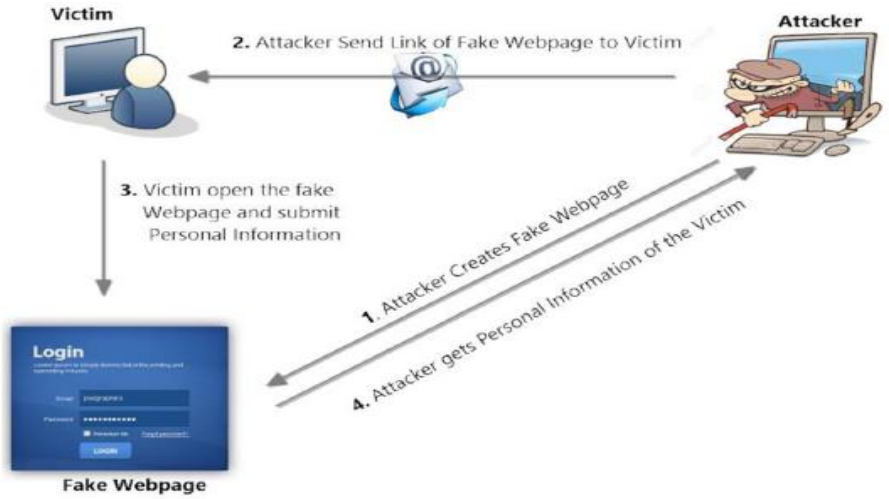


Figure 1: Phishing life cycle

Step 1: The attacker copies the content from the website of a well-known company or a bank and create a phishing website. The attacker keeps a visual similarity of the phishing website similar to the corresponding legitimate website to attract more users.

Step 2: The attacker writes an email and includes the link of the phishing website and sends it to a large number of users. In the case of spear phishing, a mail is sent to only select targeted users.

Step 3: The user opens the email and visits the phishing website. The phishing website asks the user to input personal information, for example, if the attacker mimics the phishing website of a well-known bank, then the users of bank are very likely to give up their credentials to the fake website.

Step 4: The attacker gets personal information of the user via the fake website and uses this information of the user for financial or some other benefits.

Most of the anti-phishing techniques are based on heuristics, which include the keyword frequently appearing in the phishing website [3][4]. If these techniques detect the keywords written in the English language, then they cannot detect other languages, e.g., Chinese, Hindi, Japanese, etc.

In general, phishing detection techniques can be classified as either user education or software-based anti-phishing techniques. Software-based techniques can be further classified as list-based, heuristic-based [3][4][5] and visual similarity-based techniques [6]. List-based anti-phishing techniques maintain a blacklist, white-list, or combination of both. In black-list based anti-phishing approach, a black-list is maintained which contains suspicious domain names and IP addresses. Black-lists are frequently updated; however, most of the black-list-based approaches are not effective in dealing with zero-hour phishing attacks [7].

Authors in [7] conclude that 47 % to 83 % of phishing domains update in the black-list after 12 h. Some of the approaches making use of black-lists are Google Safe Browsing API, DNS-based black-lists, and predictive black-listing. However, maintaining a black-list requires a great deal of resources for reporting and confirmation of the suspicious websites. Many people have proposed ways in which to eliminate spam emails (see, for example, [8][9]) Many of these approaches use a naive methodology, ranging from "bag-of-words" approaches, where the features of an email are the presence or absence of highly frequent and rare words, to analysis of the entropy of the messages. While these approaches looking at the text of the email appear to do well for spam, phishing messages still get through these filters.

The most promising methods utilize the general concept of feature-based phishing detection. In [10], key words are extracted from every e-mail, and then the web pages linked within the e-mail are compared with web sites which are close to these key words based on their visual layout. In a related approach, a browser plugin which analyzes the content of a website referred to from an email has been described in [11].
Another feature based approach called PILFER has been described in [12], where ten features are used for deciding whether an e-mail is considered a phishing message. For a binary classification of ham vs. phishing messages, an overall accuracy of 99.5%, 0.2% false positive rate, and 4% false negative rate is reported.

Similarly other researchers have tested the same or similar features using other classifiers such as logistic regression, classification and regression trees, random forests, neural networks, K-means, self-organizing maps, and a confident weighted online learning algorithm [13] [14] [15]. While these approaches have demonstrated the ability to detect phishing emails, phishers continue to evolve their attacks to bypass such filters.

Content-based detection techniques generally download the content hosted at the URL and use features extracted from the content to identify phish. These techniques require robust website scraping techniques in order to ensure the content is sufficiently retrieved. Content-based detection can combine techniques that draw features from the text of the main index page, characteristics of sets of component files, and measures of visual similarity among websites to identify phishing attacks [16] [17] [18].

System architecture and Implementation

The architecture of proposed solution included four main modules.

1. **Creation of a mail system and database operations.**

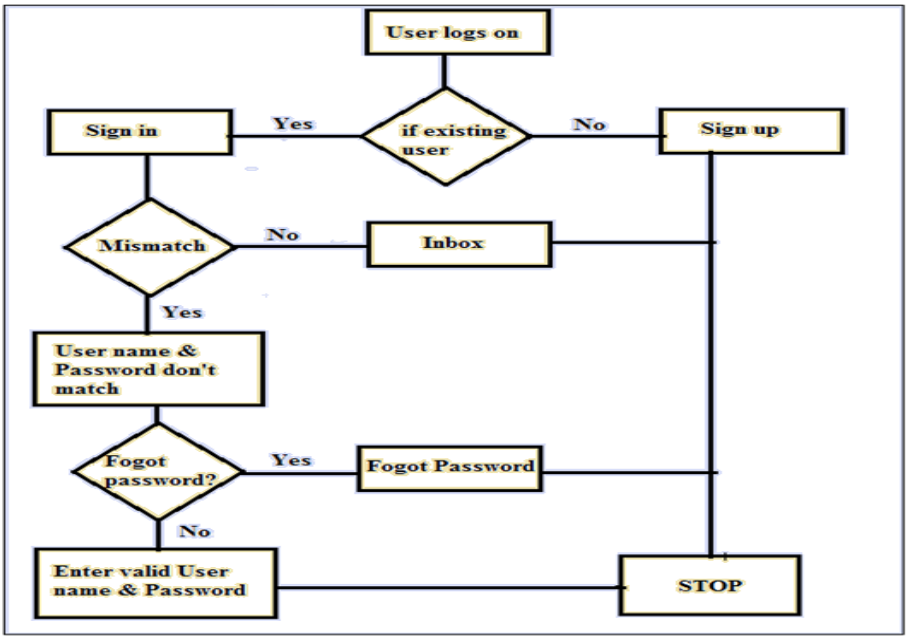


Figure 02: Creation of a mail system and database operations.

This module deals with the user interface for the home page, sign-in, sign-up and forgot your password pages. This module enables a new user to Sing-Up. It also enables an existing user to Sign-In. The user may use the Forget password link if he did forget his password.

The password is retrieved on the basis of security question and answer given by the user. Database operation manages the users. Every time a new user signs in his details are written in to the database.

2. **Composes, send and receive a mail**

The module 2 enables the user to compose and send a mail. It also allows the user to read a received mail. Once a mail is sent the date and the subject of the mail gets displayed. The received mail can be checked if it is phishing or not, the implementation of which is given in the next module. The compose mail option contains an option for spoof id. The spoof id allows the mail of the composer to be delivered with a different from address. This is being incorporated to demonstrate the Link Guard algorithm.

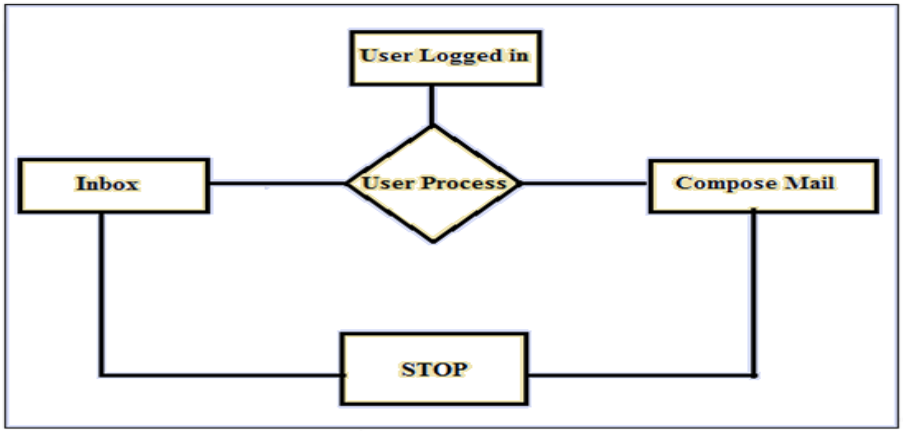


Figure 03: Composes, send and receive a mail

3. Implementation of the link guard algorithm.

It is possible to add domain names and categorize them as either white list or black list under settings. Whenever a mail is detected as phishing the domain name in that mail automatically gets added as black list. The Link Guard algorithm checks if the domain names fall under any of the above categories of hyperlinks for phishing emails. It also refers to the database of black and white list entries and sets the status of the mail as either Phishing or Non-Phishing. Once the mail is categorized as Phishing the user can take care that he does not open the link or submit any personal, critical information on to the website.The following terminologies are used in the algorithm.

v_link: visual link;
a_link: actual_link;
v_dns: visual DNS name;
a_dns: actual DNS name;
sender_dns: sender's DNS name.

```
1 .int Link Guard (v_link, a_link} {
2. v_dns = GetDNSName(v_link);
3. a_dns = GetDNSName(a_link);
4. if ((v_dns and a_dns are not
5. empty) and (v_dns != a_dns))
6. Return PHISHING;
7. if (a_dns is dotted decimal)
8. return POSSIBLE_PHISHING;
9. if (a_link or v_link is encoded)
10.{
11. v_link2 = decode (v_link);
12. A_link2 = decode (a_link);
13. return Link Guard(v_link2, a_link2);
14. }
15. /*analyze the domain name for possible phishing*/
16.  if (v_dns is NULL)
17. return AnalyzeDNS(a_link);
18. }
```

Figure 04: Description of the linkguard algorithm.

The LinkGuard algorithm works as follows. In its main routine *LinkGuard*, it first extracts the DNS names from the actual and the visual links (lines 1 and 2). It then compares the actual and visual DNS names, if these names are not the same, then it is phishing of category 1 (lines 3-5). If dotte decimal IP address is directly used in actual dns, it is then a possible phishing attack of category 2 (lines 6 and 7). We will delay the discussion of how to handle possible phishing attacks later. If the actual link or the visual link is encoded

```
19. int AnalyzeDNS (actual link) {
/return PHISHING;
*Analyze the actual DNS name according to the blacklist and whitelist
*/
20. if (actual_dns in blacklist)
21. if (actual_dns in whitelist)
22. Return NOTPHISHING;
23. return Pattern Matching(actual_link);
}
24. int Pattern Matching(actual_link){
25. if (sender_dns and actual_dns are different)
26. return POSSIBLE_PHISHING;
27. for (each item prev_dns in seed_set)
28. {
29. bv = Similarity(prev_dns, actual_link);
30. if (bv == true)
31. return POSSIBLE_PHISHING;
32. }
33. return NO_PHISHING;
}
34. float Similarity (str, actual_link) {
35. if (str is part of actual_link)
36. Return true;
37 int maxlen = the maximum string
38. Lengths of str and actual_dns;
39 int minchange = the minimum number of
40. Changes needed to transform str
41. To actual_dns (or vice verse);
42.  if (thresh<(maxlen-minchange)/maxlen<1)
43. return true/
44. return false;
45}
```

Figure 05. The subroutines used in the LinkGuard algorithm.

(Categories 3 and 4), we first decode the links, then recursively call *LinkGuard* to return a result (lines 8-13). When there is no destination information (DNS name or dotted IP address) in the visual link (category 5), LinkGuard calls *AnalyzeDNS* to analyze the actual dns (lines 16 and 17). LinkGuard therefore handles all the 5 categories of phishing attacks. *AnalyzeDNS* and the related subroutines are depicted in Fig.05. In *AnalyzeDNS*, if the actual dns name is contained in the blacklist, then we are sure that it is a phishing attack (lines 18 and 19). Similarly, if the actual dns is contained in the whitelist, it is therefore

not a phishing attack (lines 20 and 21). If the actual dns is not contained in either whitelist or blacklist, *PatternMatching* is then invoked (line 22).

We have implemented the LinkGuard algorithm in Windows 8.1 Pro. It includes two parts: a whook.dll dynamic library and a LinkGuard executive. The structure of the implementation is depicted in Fig 06.

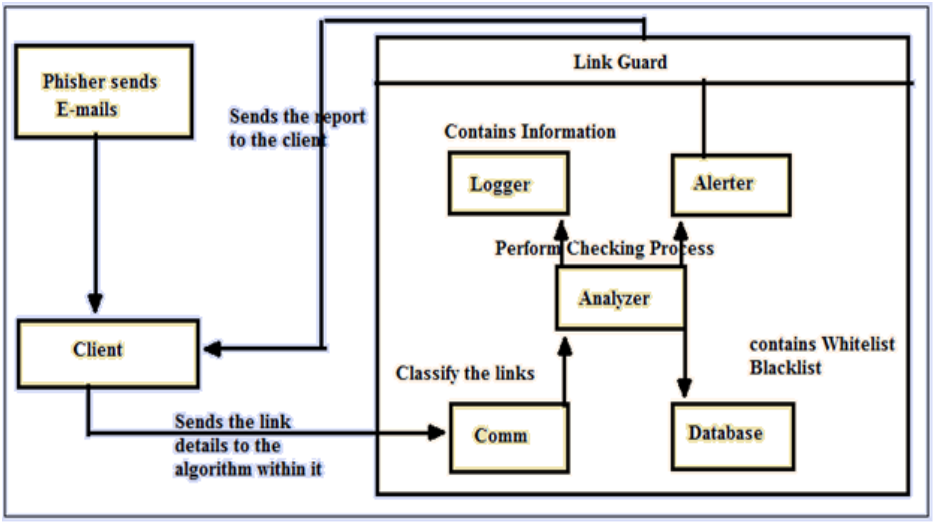


Figure 06: The structure of the Implementation of LinkGuard

Whook is a dynamic link library, it is dynamically loaded into the address spaces of the executing processes by the operating system. Whook is responsible for collecting data, such as the called links and visual links, the user input URLs. More specifically, whook.dll is used to:

1) install a BHO (browser helper object) for IE to monitor user input URLs; 2) install an event hook with the *SetWinEventHook* provided by the Windows operating system to collect relevant information; 3) retrieve sender‟s e-mail address from Outlook; 4) analyze and filter the received windows and browser events passed by the BHO and the hook, and pass the analyzed data to the LinkGuard executive. LinkGuard is the key component of the implementation. It is a standalone windows program with GUI (graphic user interface).

Analyzer, Alerter, Logger, Comm, and Database. The functionalities of these 5 parts are given below:

Comm: Communicate with the whook.dll of all of themonitored processes, collect data related to user input fromother processes (e.g. IE, outlook, firefox, etc.), and send these data to the Analyzer, it can also send commands (suchas block the phishing sites) from the LinkGuard executiveto whook.dll. The communication between the LinkGuard process and other processes is realized by the shared memory mechanism provided by the operating system.

Database: Store the whitelist, blacklist, and the user input URLs.

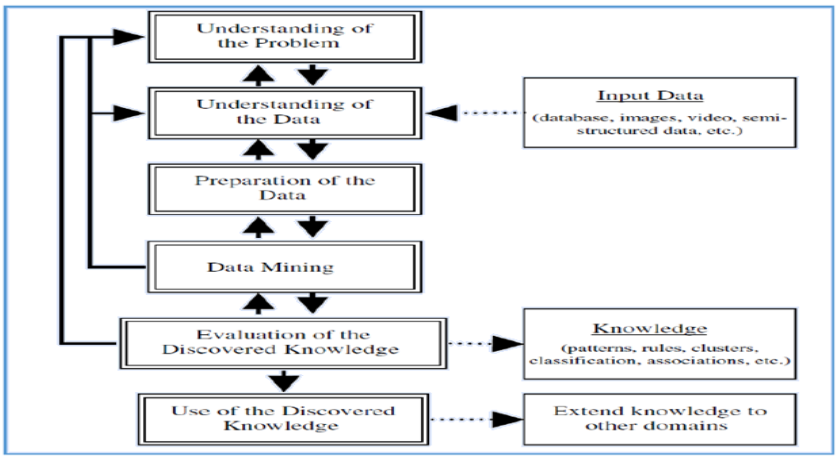
Analyzer: It is the key component of LinkGuard, which implements the LinkGuard algorithm, it uses data provided by Comm and Database, and sends the results to the Alertand Logger modules.

Alerter: When receiving warning messages from Analyzer, it shows the related information to alert the users and send back the reactions of the user back to the Analyzer.

Logger: Archive the history information, such as use revents, alert information, for future use. After implemented the LinkGuard system, we have designed experiments to verify the effectiveness of our algorithm. Since we are interested in testing Link Guard"s ability to detect unknown phishing attacks, we set both whitelist and black list to empty in our experiments. Our experiments showed that Phishing Guard can detect 195 phishing attacks out of the 203APWG archives (with detection rate 96%). For the 8 undetected attacks, 4 attacks utilize certain Web site vulnerabilities.

Hence the detecting rate is higher than 96% if category 5 is not included. Our experiment also showed that our implementation used by small amount of CPU time and memory space of the system. In a computer with 1.6G Pentium CPU and 512MBmemory, our implementation consumes less than 1% CPU time and its memory footprint is less than 7MB.Our experiment only used the phishing archive provided by APWG as the attack sources. We are planning to use LinkGuard in daily life to further evaluate and validate its effectiveness. Since we believe that a hybrid approach may be more effective for phihsing defense, we are also planning to include a mechanism to update the blacklist and whitelist in real-time.

Implementation of content based filtering.
The proposed approach for phishing email classification employs the model of knowledge discovery (KD) and data mining for building an intelligent email classifier that is able to classify a new email message as a legitimate or spam; the proposed model is built by applying the iterative steps of KD to identify and extract useful features from a training email data set, the features are then fed to a group of data mining algorithms to identify the best classifier.



Results

Figure 07: Login Page

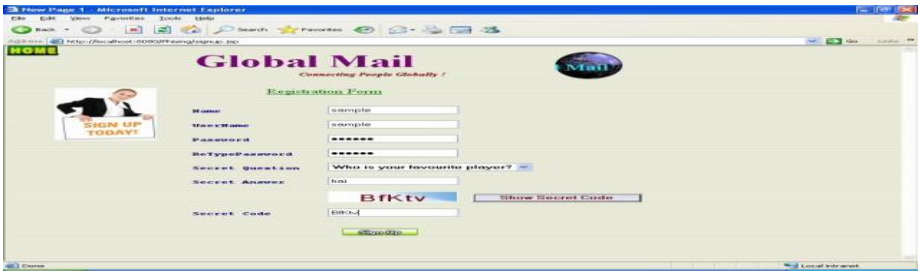


Figure 08: Registration Form

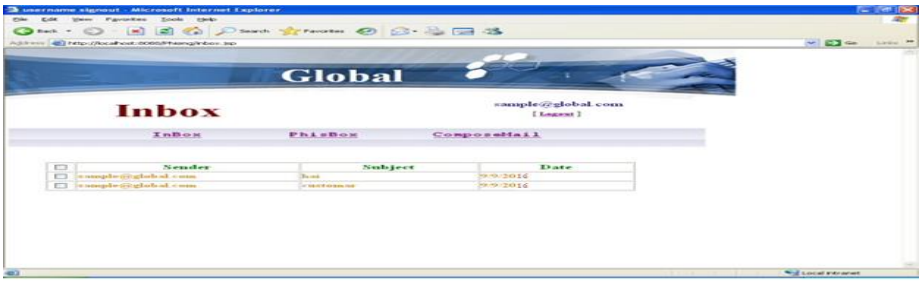


Figure 09. Inbox

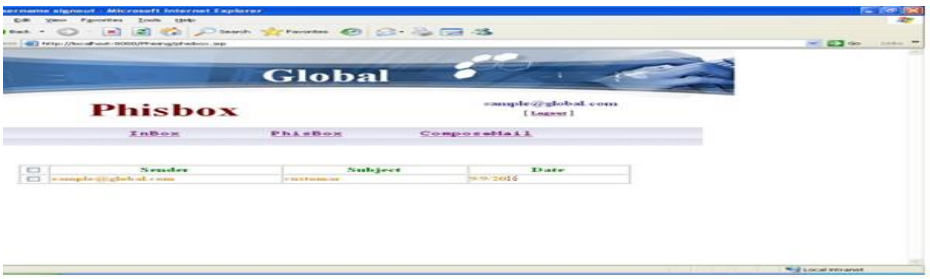


Figure 10: Phisbox

Figure 11: Compose mail

Conclusion and Future work

Phishing has becoming a serious network security problem, causing financial loss of billions of money to both consumers and e-commerce companies. And perhaps more fundamentally, phishing has made e-commerce distrusted and less attractive to normal consumers. In this project, we have studied the characteristics of the hyperlinks that were embedded in phishing e-mails and how phishing can be identified by the contents of the email body. Then We designed an anti-phishing algorithm, Link Guard, based on the derived characteristics and content based filtering using the knowledge discovery studies. Since Phishing Guard is characteristic based, it can not only detect known attacks, but also is effective to the unknown ones.

Our experiment showed that Link Guard is light-weighted and can detect up to 96% unknown phishing attacks in real-time. We believe that Link Guard is not only useful for detecting phishing attacks, but also can shield users from malicious or unsolicited links in Web pages and Instant messages.

As future work, the proposed model could be further enhanced by developing an adaptive mechanism to reflect the contributions of analyzing new emails term frequency and applying enhanced linguistic processing techniques to strengthen the similarity between phishing emails terms such that a better classification results are obtained and also including the filtering process for the images that embedded with the spam text.

References

[1] Spam Slayer: Do You Speak Spam? PCWorld.com. Retrieved on August 16, 2006

[2] A Almomani, BB Gupta, T Wan, A Altaher, Phishing Dynamic Evolving Neural Fuzzy Framework for Online Detection Zero-Day Phishing Email. Indian J.Sci. Technol. 6, no. 1, 3960–3964 (2013)

[3]M Moghimi, AY Varjani, New rule-based phishing detection method. Expert Syst. Appl. 53, 231–242 (2016)

[4] R Gowtham, I Krishnamurthi, A comprehensive and efficacious architecture for detecting phishing webpages. Comput. Secur. 40, 23–37 (2014)

[5] GA Montazer, S Yarmohammadi, Detection of phishing attacks in Iranian e-banking using a fuzzy–rough hybrid system. Appl. Soft Comput.35, 482–492 (2015)

[6] W Liu, X Deng, G Huang, AY Fu, An antiphishing strategy based on visual similarity assessment. IEEE Internet Comput. 10(2), 58–65 (2006)

[7] S Sheng, B Wardman, G Warner, L Cranor, J Hong, and C Zhang, An empirical analysis of phishing black-lists, in Proceeding of the Sixth Conference on Email and Anti-Spam, CEAS, 2009

[8] B. Leiba and N. Borenstein, "A multifaceted approach to spam reduction," in *Proceedings of the First Conference on Email and Anti-Spam (CEAS)*, 2004. [Online]. Available: http://www.ceas.cc/papers-2004/127.pdf

[9] W. Cohen, "Learning to classify English text with ILP methods," in *Advances in Inductive Logic Programming*, L. De Raedt, Ed. IOS Press, 1996, pp. 124–143. [Online]. Available:citeseer.ist.psu.edu/cohen96learning.html

[10] W. Liu, X. Deng, G. Huang, and A. Y. Fu. An antiphishing strategy based on visual similarity assessment. IEEE Internet Computing, 10(2):58–65, 2006.