Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 54 (2015) 147 – 156

Procedia Computer Science

Eleventh International Multi-Conference on Information Processing-2015 (IMCIP-2015)

PhishShield: A Desktop Application to Detect Phishing Webpages through Heuristic Approach

Routhu Srinivasa Rao* and Syed Taqi Ali

Department of Computer Engineering, National Institute of Technology, Kurukshetra, Haryana, India

Abstract

Phishing is a website forgery with an intention to track and steal the sensitive information of online users. The attacker fools the user with social engineering techniques such as SMS, voice, email, website and malware.

In this paper, we implemented a desktop application called PhishShield, which concentrates on URL and Website Content of phishing page. PhishShield takes URL as input and outputs the status of URL as phishing or legitimate website. The heuristics used to detect phishing are footer links with null value, zero links in body of html, copyright content, title content and website identity. PhishShield is able to detect zero hour phishing attacks which blacklists unable to detect and it is faster than visual based assessment techniques that are used in detecting phishing. The accuracy rate obtained for PhishShield is 96.57% and covers a wide range of phishing web sites resulting less false negative and false positive rate.

Peer-review under responsibility of organizing committee of the Eleventh International Multi-Conference on Information Processing-2015 (IMCIP-2015)

Keywords: Anti phishing; Copyright; Footer HTML; Phishing; Website identity.

1. Introduction

Phishing is criminal activity which aims to trace the user's sensitive information such as username, password, bank account number, credit card details and social security number without his/her permission. This activity is possible by designing a new website mimicking the trusted website in both content and design attributes. Phishers takes care such that user is unaware of entering into phishing zone and leading to disclosure of his/her sensitive information.

Anti-Phishing Working group (APWG) is an organization which collects the phishing information from various sources like APWG feed, company contributions, China Internet Network Information Center (CNNIC), Anti-phishing Alliance of China (APAC) and private sources across the world. APWG generates and releases reports in quarterly and half yearly describing the statistics of malicious domains and phishing attacks in different regions of the world. According to APWG Global Phishing survey report 1H2014[1], it received 123,741 unique phishing papers from January to June 2014, compared to around 115,565 in 2H2013[2] survey of previous year. The report states that average uptime of phishing sites is 32 hours and 32 minutes and the median of the life spans of phishing sites is 8 hours and 42 minutes. This report indicates that half of the phishing sites are getting shut down in less than a day but the average uptime clears that attackers are using sophisticated techniques to bypass heuristic antiphishing solutions.

*Corresponding author.
E-mail address: routh.srinivas@gmail.com

Peer-review under responsibility of organizing committee of the Eleventh International Multi-Conference on Information Processing-2015 (IMCIP-2015)
doi:10.1016/j.procs.2015.06.017

The sources of phishing attacks are mostly from email, websites and malware. In *email based phishing*, attacker sends millions of emails to millions of users such that at least thousands of them would fall for it. Mostly emails claims to be arriving from a trusted organization. The links provided in phishing emails draws user into entering phishing website. In *website based phishing*, website is duplicated targeting trusted website users into revealing sensitive information. Users may reach to phishing sites through some social networking sites like Facebook or Twitter. In *malware based phishing*, malicious software like Trojan horse is inserted into compromised legitimate site and when the user clicks on the link, the malicious software is installed into system and then software tracks the sensitive information within the computer and sends it to the attacker. The malware can be inserted into legitimate site via links or an audio file or a video file. Most of the recent malwares are multifunctional i.e. they can steal the data, make the victim's computer as a part of botnet or download and install other malicious software without user's notice.

Attackers sometimes target specific group of people or organizations or companies or roles to get intellectual information, business secrets or military information etc. instead of financial gain. This variation of general phishing is called Spear Phishing. Mostly these attacks are carried out by attackers who are having deep knowledge about the internet experience of user. Attacker tracks the user's frequency of visits to a legitimate site and then try to compromise the legitimate site so that the community of people will get affected. The main intention of spear phishing is to steal sensitive information where as general phishing is to engage financial frauds. Whaling is a type of spear phishing where the target of group is a bigger fish like executive officer of private business and government agencies.

Due to the rapid growth of sophisticated phishing techniques developed by sophisticated attackers, fresh phishers are easily able to create phishing websites through phishing toolkits[3] which are available in the internet. Hence use of antiphishing techniques like blacklist, whitelist, heuristic and visual similarity based approaches have become less effective in detecting phishing websites. *Blacklist*[4,5] or *whitelist*[6] works only on the URLS that are recorded in the respective list and fails to detect if the URLS falls under out of list scenario. This approach gives either false positive (whitelist) or false negative (blacklist) rate. Frequent updating of lists is must in these approaches.

Heuristic approach[7–9] studies structure of phishing websites content and URL, extracts the features of phishing and designs a model to detect phishing sites on the basis of extracted features. This approach has less false positive or false negative and faster than list based approaches but these are less accurate compared to list based approach. An attacker can bypass the heuristic filter and can reach his goal of stealing credentials after getting aware of heuristic technique.

Visual similarity based approach[10–13,21] compares the suspicious website visual content like images, text and styles with trusted domain visual content. If the similarity between the websites is above a certain threshold it is treated as phishing site otherwise as legitimate. This approach is slow compared to above two approaches because it needs an initial visual content database of all trusted websites to be compared with the suspicious website and also comparison operation of visual content is costlier than URL comparison.

To bypass filtering of antiphishing techniques, phishers use variety of techniques such as replacing of web content like text, links and styles. Therefore antiphishing techniques which are based on text[8,14], links[9,15,16] and styles[9] may stop detecting such types of phishing attacks. The antiphishing techniques based on image comparisons used to counter the above phisher tricks but may result in high response time and tests patience of online user due to high computation cost of image comparison of suspicious site and legitimate image database.

To reduce the growth of phishing attacks we need a solution consisting of combination of above antiphishing approaches. In this paper, a novel heuristic approach to detect phishing webpages has been proposed. Here we presented the working of our approach and implemented the same as a desktop application. Our application is able to detect normal phishing sites and also sites replacing content with images. In this paper, we term phishing sites replacing content with images as *image phishing site.*

The rest of the paper is organized as follows. Section 2 reviews the related work on anti-phishing techniques and their comparison with our proposed solution is given in a table. Section 3 explains methodology and working of our application. Section 4 shows the experimentation and results. Finally, we concluded the paper by giving key points of our entire work in section 5.

2. Related Work

To gain the knowledge on phishing concepts, techniques and anti-phishing techniques we did a literature survey on phishing by reading more than hundred papers. In this section we describe outlines of some anti-phishing techniques and comparison of these techniques with our work is shown in Table 1.

Table 1. Comparison of phishing detection techniques with PhishShield application.

Techniques	Language Independent	Zero day phishing attacks	Image based phishing attacks	Whitelist	Blacklist	Heuristics	Visual similarity
Google Safe Browsing [4]	Yes	No	No	No	Yes	No	No
PhishNet [5]	Yes	No	Yes	No	Yes	Yes	No
PhishGaurd [7]	Yes	Yes	Yes	No	No	Yes	No
Cantina [8]	No	Yes	No	No	No	Yes	No
SpoofGaurd [9]	Yes	Yes	No	No	No	Yes	No
BaitAlarm [11]	Yes	Yes	No	Yes	No	No	Yes
Visual similarity based phishing detection [12]	Yes	Yes	Yes	Yes	Yes	No	Yes
Liu *et al.* [13]	Yes	Yes	Yes	No	No	No	Yes
PhishShield	Yes	Yes	Yes	Yes	No	Yes	No

Google Safe Browsing[4] uses blacklist antiphishing technique to detect phishing technique. The suspicious URL is checked in the blacklist for its presence. The Suspicious URL is classified as phishing site if it is found in blacklist otherwise classified as legitimate website. The limitation in this approach is that phishing sites which are not listed in blacklist are not detected. These type of non-blacklisted phishing sites are called as Zero day phishing sites. This technique may lead to high false negative rate. A small change to the blacklisted URLs would result in no match with blacklist and hence cannot be recognized by the tool.

PhishNet[5] technique takes blacklist as input and predicts variations of each URL based on five URL variation heuristics such as Replacing Top Level Domain (TLD), Directory structure similarity, IP address equivalence, Query string substitution and Brand name equivalence. This technique covers the exact match limitation which is stated above in Google safe browsing. However, this technique also has same limitation of not detecting zero day phishing attacks.

PhishGaurd[7] extension feeds the large number of random generated credentials to the login form, restricts user's original credentials from submitting and based on the responses of server it chooses to feed the users credentials. If the response of server to bogus credential is success then user is alerted with a warning of phishing message. But the extension may create a worry to online user thinking that he/she already given his credentials to the phishing site. The extension also violates the first line of defense i.e. preventing phishing websites reaching to online users.

Cantina[8] technique depends on the textual content of the website. Term Frequency–Inverse Document Frequency (TF-IDF) algorithm applied on the textual content combined with additional heuristics used to detect phishing attacks. The top five tokens with highest TF-IDF is submitted to search engine followed by comparison of suspicious link with search engine results. This approach fails when the text of website is replaced with images or addition of invisible text which matches background color of the website.

SpoofGaurd[9] plugin works, based on the phishing symptoms of suspicious website. Some of the phishing symptoms considered are host name check, host name sensitivity, URL check, Image check, Password field check and links check. These symptoms or heuristics are assigned with weights of same value or different value. If total score of all heuristics of a suspicious website exceeds a threshold then it is classified as phishing website otherwise as legitimate. It has an advantage of detecting zero day phishing attack but has a limitation of high false positive rate.

BaitAlarm[11] uses visual features comparison to classify phishing and legitimate websites. Phishers must use same styles to imitate the graphics of legitimate website so authors considered Cascading Style Sheets (CSS) for detecting phishing websites. Authors taken a legitimate site and compared with a large number of phishing sites indicating need of whitelist. The limitation of BaitAlarm is that computation cost of CSS style comparison with whitelist database is too high.

Hara *et al.*[12] developed a technique which classifies the suspicious websites based on image similarity. Authors used ImgSeek application for comparison of legitimate and suspicious image. This technique can auto update the whitelist with addition of suspicious websites that are classified as neither legitimate nor phishing. The limitation of this approach is very high false positive and high false negative rate. Image comparison at client side leads to delay in browser's experience.

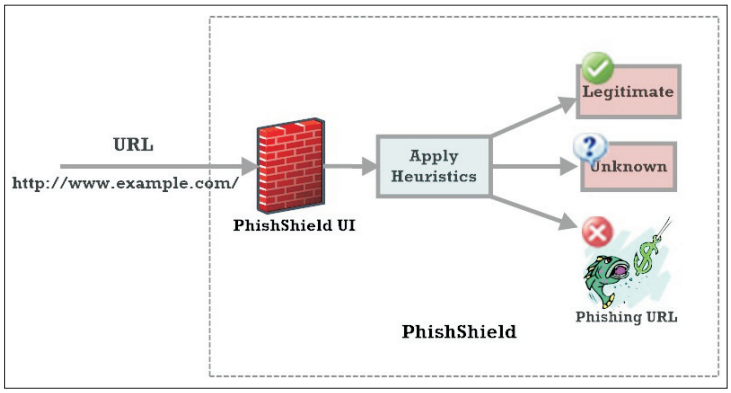

Fig. 1. Architecture of proposed solution.

Liu *et al.*[13] proposed an approach that detects phishing based on the visual features of a suspicious websites. The visual features such as block level (text and images), layout similarity (DOM) and overall style (cascaded style sheet) are compared with respective features of legitimate website. Each feature is assigned with weights according to the priority played in designing a legitimate website. Authors classify a suspicious website as phishing websites if its visual similarity is above a threshold value otherwise classified as legitimate. The limitation in this technique is high response time i.e. this scheme needs large legitimate image database and visual comparison of suspicious website with image database is too costly.

3. Proposed System

Figure 1 shows the architecture and Table 2 shows algorithm of our proposed work. We divide our work into five modules which acts as filters to verify legitimacy of the URL. These five modules are considered as five levels of detection. The name of the application called PhishShield takes URL as input and gives output as status of website i.e. phishing or legitimate or unknown.

We also calculated the identity of the URL based on the maximum frequency of domain[15] that are extracted from the hyperlinks of HTML.

3.1 Use of whitelist

In the first level of detection, domain of the URL is compared with trusted website list called Whitelist. If comparison is successful, the URL is classified as legitimate otherwise PhishShield continues to second level of detection. Before entering into the second level of the detection, HTML webpage is parsed and stored as a Document Object Model (DOM) element.

3.1.1 Detection of login page

Phishers use phishing tool kits[3] in creating fake login forms to steal sensitive information. As online users reveal sensitive information mostly in a login page therefore we mainly focus on only login page websites for phishing detection. The login page existence is found through parsing the html of website for input type = “password”. In the presence of password type field, the application PhishShield continue the execution otherwise stops the execution process as the user does not have a way to enter his/her confidential information. This filtration would prevent phishing detection process on ordinary websites not containing login forms. If needed the detection process can be extended to ordinary websites with minor alteration in the proposed system.

3.2 Zero links in body portion of HTML

In legitimate sites, the presence of at least one link is certain in the body of web page. For example, if the website contain login form then the body of the page may contain hyperlink texts as either signup or forgot password or others. In phishing pages, the page content sometimes may be replaced with images and may not contain any links in the body section of html. Sometimes replaced text images are referenced to NULL links or trusted domain links but this property is addressed in Null footer links and website identity heuristic phishing detection.

In this second level of detection, image phishing sites are filtered with a heuristic factor i.e. zero number of links in the body of HTML. If number of links is zero, it is classified as phishing site based on image phishing detection otherwise application proceeds to next level of detection.

3.3 Footer links pointing to NULL (#)

We term links in footer section of website pointing to NULL value or value starting with NULL character as NULL footer links. An anchor tag pointing to NULL value is called as NULL anchor. It indicates link is redirecting to its own page. From this fact of information we derived third level of detection heuristic.

In this third level of detection, we consider footer links present in the website and calculate the value of links. Phishers mainly focus on making user to stay in the login form. Therefore they might design login form page with some or all links as NULL links, leads users directing to a page consisting login form. Hence, in papers[16,17] authors have considered proportionality of the null links with total number of links for filtering the phishing sites but many of the legitimate sites also includes the null links such as logo link pointing to NULL (#) so it may sometimes create wrong classification. By our experimentation on various legitimate sites, we found none having null links in the footer section of the website. Hence, from this observation we derived a heuristic factor to filter the phishing sites.

If the anchor tag in the footer section is pointing to null i.e.

<a href = “#”>

<a href = “#skip”>

<a href = “#content”>

then the URL is treated as Phishing URL otherwise PhishShield forwards to next level of detection. Figure 2, shows the footer section with hyperlinks pointing to “#” value.

3.4 Use of copyright and title content

In the fourth level of detection, the tag containing copyright section and <title> tag containing title content is extracted from DOM object. In legitimate sites, copyright and title text in the page mostly contains the domain information of the website, we use this information to detect phishing with the help of whitelist. The copyright content is extracted and tokenized into terms. Each term is compared with white list for the match. If the match is successful then the entered URL is classified as phishing site otherwise parsed content is forwarded to next filter. Figure 2, shows the copyright content consisting domain information i.e. Amazon, it is extracted and compared with whitelist for checking legitimacy status.

3.5 Website identity

Website identity is determined based on the frequency of hyperlinks with in the website. In legitimate website, frequency of the hyperlinks pointing to its own domain is high when compared to frequency of the hyperlinks pointing to foreign domain. As phishers try to imitate the behavior of legitimate sites, they insert the links in their websites pointing to the target domain. This information is used to identify the website identity of the given URL by calculating domain of the link with maximum frequency. If the domain of input URL of PhishShield application does not match with domain having maximum frequency (website identity) then input URL is considered as phishing site targeting to domain with maximum frequency. For example, www.abc.com is the input URL to the application and www.ebay.co.in

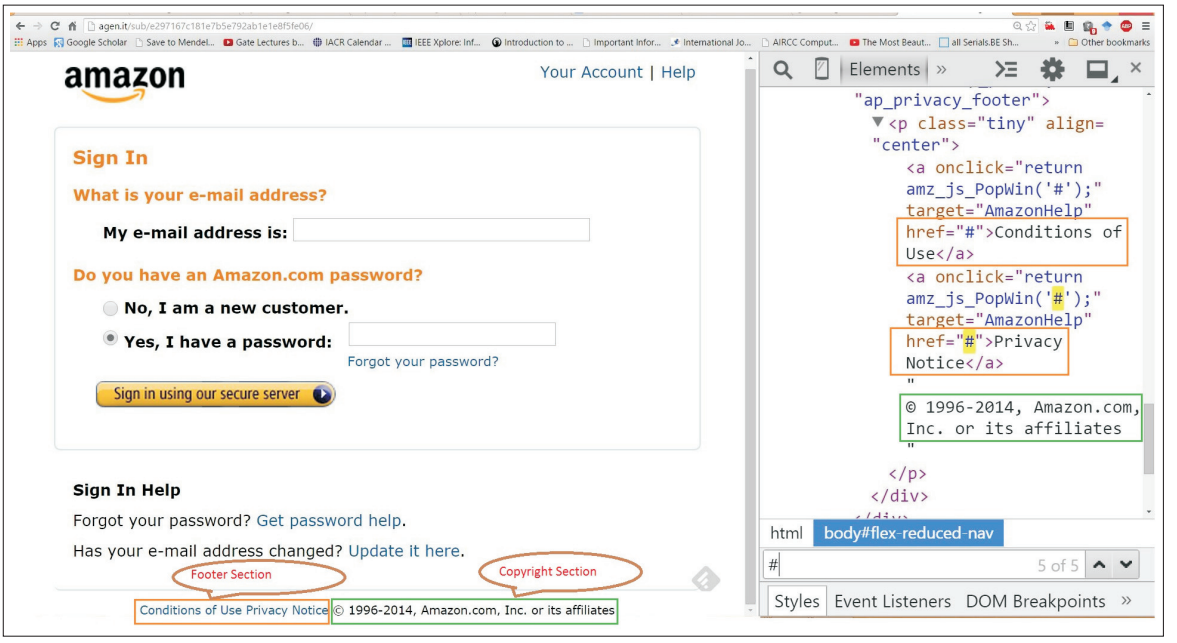

Fig. 2. Screenshot of phishing site along with footer and copyright section.

is the domain with high frequency, as both of the domains differ, www.abc.com is considered as phishing site targeting www.ebay.co.in.

This filter is used not only to detect phishing websites but also identifies phishers target domain that is being imitated. The parsed HTML content is passed through this filter mandatorily even if the phishing has been detected from the above filters so that target website is revealed to the user.

4. Experimentation and Results

We implemented PhishShield which is based on URL content and Web content of the URL such as null footer links, copyright and title content. To develop our tool we used NetBeans 8.0.2 IDE, JAVA compiler, JSoup API and firebug tool.

JSoup[19] is used for parsing the html contents of webpage and extracting html contents such as links in footer, copyright, title, CSS etc. Firebug is an open source Firefox extension which is used for debugging, editing, and monitoring of any website's CSS, HTML, DOM, XHR, and JavaScript.

The whitelist we used for the experiments is based on the target list from PhishTank[20]. PhishTank is an antiphishing website where anyone can submit, verify, track or share phishing data. It maintains a phishing archive consisting of valid, unknown, online or offline phishing sites. To evaluate the performance of our PhishShield application in detecting phishing websites, we collected 1600 phishing valid, invalid, unknown, offline, online phishing sites URL. We also collected 250 legitimate websites, out of which 176 are taken from PhishTank and remaining are considered randomly.

4.1 Evaluation metrics

In order to calculate the accuracy of our proposed system we used following evaluation parameters[18].

- False Positive (F_{Pos}):

Table 2. PhishShield algorithm.

Algorithm of our proposed work

Input: an URL
Output: label (legitimate = 0, phishing = 1, unknown = 2.
Algorithm:

1. **ValidateWebsite(String URL)** //validate URL using Whitelist
   - 1.1. Domain=Extract_Domain (URL);
   - 1.2. for each Host name (Host) in Whitelist
   - 1.3. Status=Compare (Host, Domain)
   - 1.4. If (status) return 0;
   - 1.5. else goto step 2

2. **int PhishShield (String URL)**
   - 2.1. Document Doc= Jsoup.Connect (String URL) /*Parse the html of website using JSoup and store the content in a document object Doc */
   - 2.2. If (Parsing == Successful) // JSoup Connection is successful
     - 2.2.1. If (Doc has input type = =password) // validate login
       - 2.2.1.1. int label=0;
       - 2.2.1.2. **ImageBasedPhishing (Document Doc)**//Check for number of links (n1) in body of html
         - 2.2.1.2.1. n1= doc.body().select("a");
         - 2.2.1.2.2. If (n1! = 0) goto step 2.2.1.3
         - 2.2.1.2.3. elselabel = 1; //indicating image phishing website.
         - 2.2.1.2.4. goto step 2.2.1.5;
       - 2.2.1.3. **NullFooterLinks (Document Doc)** //Check for number of footer links equalling to null
         - 2.2.1.3.1. Elements f1 = doc.select("div[id~=bottom|footer] a");
         - 2.2.1.3.2. for each link in f1
         - 2.2.1.3.3. f2=checkforNullLinks(Elements f1)// we compared each link with '#" value
         - 2.2.1.3.4. if(f2==0) goto step 2.2.1.4
         - 2.2.1.3.5. else label = 1 // indicating phishing sites having footer links to null
         - 2.2.1.3.6. goto step 2.2.1.5;
       - 2.2.1.4. **CopyrightTitle (Document Doc)**//Extract the copyright and title section from html.
         - 2.2.1.4.1. Tokenize the copyright or title section content
         - 2.2.1.4.2. Compare each token with whitelist
           - 2.2.1.4.2.1. If comparison successful
           - 2.2.1.4.2.2. label = 1
           - 2.2.1.4.2.3. else goto step 2.2.1.5
       - 2.2.1.5. **WebsiteIdentity (Document Doc)** //calculate the frequency of each domain in links of the webpage each link and
         - 2.2.1.5.1. Webidentity= CalculateDomainwithMaximumFrequency(Doc); /* we counted frequency of found maximum frequency domain*/
         - 2.2.1.5.2. If (domain of input URL! = web identity)
         - 2.2.1.5.3. then label = 1 // i.e. input URL is targeting website identity.
       - 2.2.1.6. return label
     - 2.2.2. Else Stop Executing PhishShield Application // case of absence of password field
   - 2.3. Else return 2; // case of parsing failure

This measures the rate of legitimate sites (L) wrongly classified as phishing sites (P).

$$F_{\text{Pos}} = \frac{L \rightarrow P}{(L \rightarrow P) + (L \rightarrow L)} \tag{1}$$

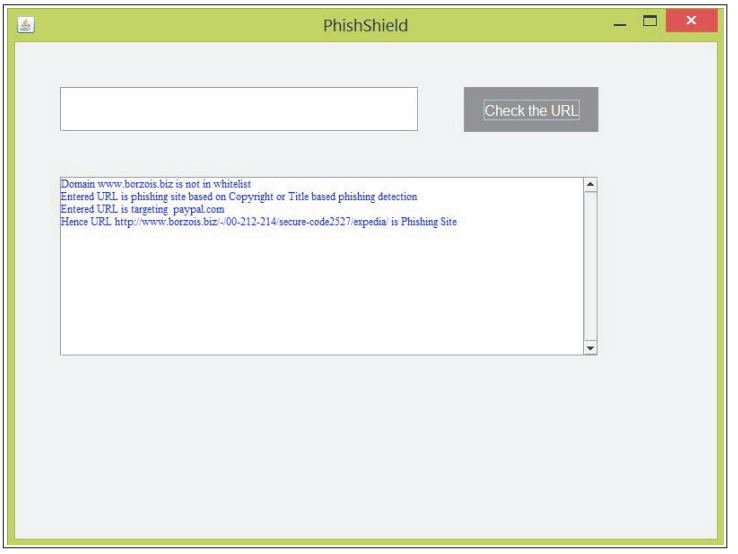

Fig. 3. Verification of suspicious URL through PhishShield application.

- False Negative (F_{Neg}):
  This measures the rate of phishing sites (P) wrongly classified as legitimate sites (L).

$$F_{Neg} = \frac{P \to L}{(P \to L) + (P \to P)} \tag{2}$$

- True Positive (T_{Pos}):
  This measures the rate of phishing sites (P) correctly classified as Phishing sites (P).

$$T_{\text{Pos}} = \frac{P \to P}{(P \to P) + (P \to L)} \tag{3}$$

- True Negative (T_{Neg}):
  This measures the rate of legitimate sites (L) correctly classified as legitimate sites (L).

$$T_{Neg} = \frac{L \to L}{(L \to L) + (L \to P)} \tag{4}$$

- Accuracy (Acc):
  This measures the overall rate of correctly detected phishing and legitimate instances in relation to all instances.

$$Acc = \frac{(L \to L) + (P \to P)}{(L \to L) + (L \to P) + (P \to L) + (P \to P)} \tag{5}$$

  where $L \to P$ is number of legitimate sites misclassified as phishing, $L \to L$ is number of legitimate sites correctly classified as legitimate, $P \to L$ is number of phishing sites misclassified as legitimate, $P \to P$ is number of phishing sites correctly classified as phishing.

On experimenting 1600 phishing sites and 250 legitimate sites we could get the below values of metrics as shown in Fig. 4. Column chart. Figure 3 shows verification of suspicious website with PhishShield.

The effectiveness of our proposed method depends on the right input. Our method fails to detect phishing when all of the filters are bypassed by the phishers. There are some types of phishing sites, requests sensitive information on

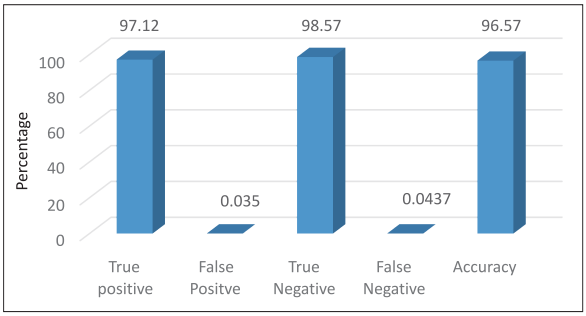

Fig. 4. Performance results of PhishShield application.

pages that do not mimic any legitimate webpage. PhishShield fails to detect these category of phishing sites because these sites does not use targeted legitimate content to display webpage hence bypassing of filters takes place in these scenarios. Of course if legitimate website is not mimicked to request sensitive information then even an unsophisticated user can identify the difference and get a suspicion on it. PhishShield also fails to detect phishing sites when JSoup parsing failure occurs.

5. Conclusion

In this paper we have proposed a novel heuristic solution to detect phishing attacks and developed it as an application called PhishShield. The heuristic features presented in the paper are extracted from website without intervention of user using JSoup. We have explained our method with algorithm and shown that our solution detects phishing based on heuristics (copyright, null footer links, zero links of body html, link with maximum frequency domain) and whitelist.

The main advantage of our Application is that it can detect phishing sites which tricks the users by replacing content with images, which most of the existing anti-phishing techniques not able to detect, even if they can, they take more execution time than our application.

We believe there are many possible ways to improve these results in terms of performance and computation cost. In future, we may involve developing additional heuristics combined with other techniques like genetic algorithms, neural networks to increase the accuracy and improve the response time of the application.

References

[1] APWG, Phishing activity trends paper. [Online]. http://docs.apwg.org/reports/APWG_ Global_ Phishing_ Report_ 1H_ 2014.pdf
[2] APWG, Phishing activity trends paper. [Online]. http://docs.apwg.org/reports/APWG_ GlobalPhishingSurvey_ 2H2013.pdf
[3] Sophos, Do-it-yourself phishing kits found on the internet, reveals Sophos, Technical paper, Sophos, August (2004). [Online]. http://www.sophos.com/pressoffice/news/articles/2004/08/sa_ diyphishing.html
[4] Safe Browsing API – Google Developer, [Online] Available at https://developers.google.com/safe-browsing/
[5] P. Prakash, M. Kumar, R. R. Kompella and M. Gupta, Phishnet: Predictive Blacklisting to Detect Phishing Attacks, In *INFOCOM, 2010 Proceedings IEEE*, pp. 1–5, March (2010).
[6] Y. Cao, W. Han and Y. Le, Anti-Phishing based on Automated Individual White-List, In *Proceedings of the 4th ACM Workshop on Digital Identity Management ACM*, pp. 51–60, October (2008).
[7] Y. Joshi, S. Saklikar, D. Das and S. Saha, PhishGuard: A Browser Plug-In for Protection from Phishing, In *2nd International Conference on Internet Multimedia Services Architecture and Applications, IMSAA 2008, IEEE*, pp. 1–6, December (2008).
[8] Y. Zhang, J. I. Hong and L. F. Cranor, Cantina: A Content-Based Approach to Detecting Phishing Web Sites, In *Proceedings of the 16th International Conference on World Wide Web, ACM*, pp. 639–648, May (2007).
[9] N. Chou, R. Ledesma, Y. Teraguchi and J. C. Mitchell, Client-Side Defense Against Web-Based Identity Theft, In *NDSS*, February (2004).
[10] A. Y. Fu, L. Wenyin and X. Deng, Detecting Phishing Web Pages with Visual Similarity Assessment based on Earth Mover's Distance (EMD), *IEEE Transactions on Dependable and Secure Computing*, vol. 3(4), pp. 301–311, (2006).

[11] J. Mao, P. Li, K. Li, T. Wei and Z. Liang, Baitalarm: Detecting Phishing Sites using Similarity in Fundamental Visual Features, In *5th International Conference on Intelligent Networking and Collaborative Systems, INCoS 2013, IEEE*, pp. 790–795, September (2013).
[12] L. Wenyin, G. Huang, L. Xiaoyue, Z. Min and X. Deng, Detection of Phishing Webpages based on Visual Similarity, In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web, ACM*, pp. 1060–1061, May (2005).
[13] L. Wenyin, G. Huang, L. Xiaoyue, Z. Min and X. Deng, Detection of Phishing Webpages based on Visual Similarity, In *14th International Conference on World Wide Web (WWW): Special Interest Tracks and Posters*, (2005).
[14] G. Xiang, and J. I. Hong, A Hybrid Phish Detection Approach by Identity Discovery and Keywords Retrieval, In *Proceedings of the 18th International Conference on World Wide Web*, ACM, pp. 571–580, April (2009).
[15] G. Ramesh, I. Krishnamurthi and K. S. S. Kumar, An Efficacious Method for Detecting Phishing Webpages through Target Domain Identification, Decision Support Systems, vol. 61, pp. 12–22, (2014). [Online]. Available at: http://www.sciencedirect.com/science/article/pii/S0167923614000037
[16] M. He, S. J. Horng, P. Fan, M. K. Khan, R. S. Run, J. L. Lai and A. Sutanto, An Efficient Phishing Webpage Detector, *Expert Systems with Applications*, vol. 38(10), pp. 12018–12027, (2011).
[17] R. M. Mohammad, F. Thabtah and L. McCluskey, An Assessment of Features Related to Phishing Websites using an Automated Technique, In *2012 International Conference for Internet Technology and Secured Transactions, IEEE*, pp. 492–497, December (2012).
[18] M. Khonji, Y. Iraqi and A. Jones, Phishing Detection: A Literature Survey, *Communications Surveys & Tutorials, IEEE*, vol. 15(4), pp. 2091–2121, (2013).
[19] jSoup – A Java API, Available at http://jsoup.org/
[20] PhishTank, [Online]. Available at https://www.phishtank.com/
[21] R. S. Rao and S. T. Ali, A Computer Vision Technique to Detect Phishing Attacks, In *5th International Conference on Communication Systems and Network Technologies (CSNT)*, 2015, IEEE, Paper Presented at the Meeting of Conference on Communication Systems and Network Technologies (CSNT), Gwalior, India, (2015).