EURASIP Journal on
Information Security

RESEARCH Open Access

A novel approach to protect against phishing attacks at client side using auto-updated white-list

Ankit Kumar Jain and B. B. Gupta*

Abstract

Most of the anti-phishing solutions are having two major limitations; the first is the need of a fast access time for a real-time environment and the second is the need of high detection rate. Black-list-based solutions have the fast access time but they suffer from the low detection rate while other solutions like visual similarity and machine learning suffer from the fast access time. In this paper, we propose a novel approach to protect against phishing attacks using auto-updated white-list of legitimate sites accessed by the individual user. Our proposed approach has both fast access time and high detection rate. When users try to open a website which is not available in the white-list, the browser warns users not to disclose their sensitive information. Furthermore, our approach checks the legitimacy of a webpage using hyperlink features. For this, hyperlinks from the source code of a webpage are extracted and apply to the proposed phishing detection algorithm. Our experimental results show that the proposed approach is very effective for protecting against phishing attacks as it has 86.02 % true positive rate while less than 1.48 % false negative rate. Moreover, our proposed system is efficient to detect various other types of phishing attacks (i.e., Domain Name System (DNS) poisoning, embedded objects, zero-hour attack).

Keywords: Cyber security, Phishing, Black-list, White-list, Social engineering, DNS poisoning, Hyperlinks

1 Introduction

Phishing is a cyber security threat which is performed with the help of social engineering techniques to trick Internet users into revealing personal and secret information [1]. Detection and prevention of phishing attacks is a big challenge as the attacker performs these attacks in such a way that it can bypass the existing anti-phishing techniques [2, 3]. Moreover, sometimes an educated and experience user may also fall under this attack [4]. In this attack, the attacker makes a fake webpage by copying or making a little change in the legitimate page, so that an internet user will not able to differentiate between phishing and legitimate webpages. One of the effective solutions to prevent a phishing attack is to integrate security features with the web browser which can raise the alerts whenever a phishing site is accessed by an internet user. Generally, web browsers provide security against phishing attacks with the help of list-based solutions. The list-based solutions contain either black-list or white-list. These list-based solutions match the given domain with the domains present in the black-list or white-list to take the appropriate decision [5, 6]. The combination of technical experts and security software verify when a new domain needs to be added in this list. Security software checks the various features of a webpage to verify identity [7].

According to the anti-phishing working report in the second half of 2014, 123,972 unique phishing attacks were found worldwide between July to December 2014 [8]. E-commerce, banks, and money transfer companies are the most targeted industries by these attacks. Seventy-five percent of phishing websites used five top-level domains namely .com, .tk, .pw, .cf, and .net. The median uptime of phishing websites in the second half of 2014 increased to 10 h and 6 min (i.e., half of all phishing attacks stay active for slightly more than 10 h) [8]. Internet services providers (ISPs) were the most targeted industry sector during the first three quarters of 2015, surpassing the banking and financial service sectors coming in second and third during the 9-month period [9]. The

* Correspondence: gupta.brij@gmail.com
National Institute of Technology Kurukshetra, Kurukshetra, India

attackers targeted Internet services providers because ISP account contains the identification details of users, credit card information, and secret information regarding the domain name [10, 11]. An attacker can also send the spam mail from the hacked user's account. One of the major problems of 2015 is the Business Email Compromise (BEC) scam [9]. In this scam, the attacker fools industries into transferring large amounts of money using spear-phishing techniques.

1.1 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 [12]. The life cycle of a phishing attack is shown in Fig. 1. The following steps are involved in a phishing attack:

Step 1: The attacker copies the content from the website of a well-known company or a bank and creates 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 selected 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.

1.2 Phishing attack classification

Phishing scams can be carried out using technical subterfuge and social engineering [6]. Social engineering strategies use "spoofed" emails to guide users to fraudulent websites. Phishing URL can also be spread by Internet Relay Chat (IRC) and instant messaging (IM), forum, blogs, etc. The phishers send the same email to thousands of users and asking them to input personal information. Some of the major phishing attacks to fool an internet user are explained below:

The zero-hour phishing attack: A zero-hour vulnerability refers to a hole in anti-phishing technique that is unknown to the vendor. This security hole is then exploited by hackers before the vendor becomes aware and hurries to fix it.
Embedded objects: A real webpage is downloaded to build the phishing webpage which appears just similar to a genuine webpage in appearance. Attackers cover the address bar by using an image or script which makes the online user believe that they have input to the right website. Attackers also use the embedded objects (flash, images, etc.) instead of HTML codes to avoid phishing detection techniques.
Domain Name System (DNS) attack: DNS cache poisoning exploits vulnerabilities in the domain name system. In this attack, attackers divert Internet traffic from the legitimate website to the phishing website.
Language dependent: Most of the anti-phishing techniques are based on heuristics, which include the

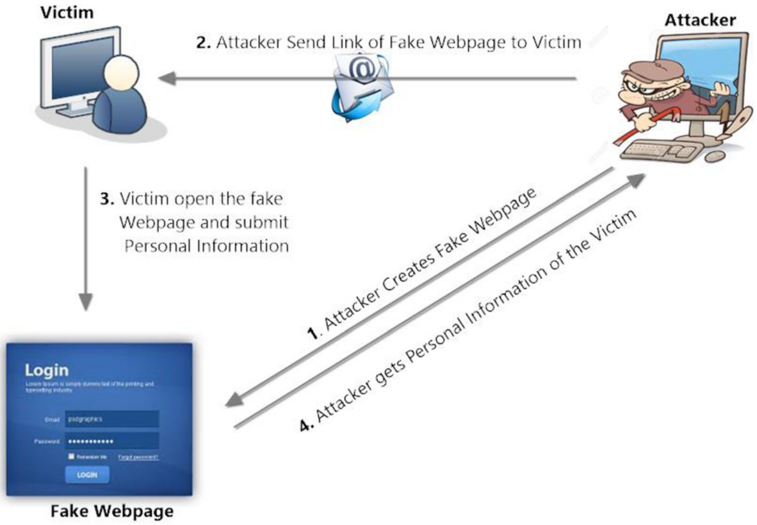

Victim
2. Attacker Send Link of Fake Webpage to Victim
Attacker
3. Victim open the fake Webpage and submit Personal Information
1. Attacker Creates Fake Webpage
4. Attacker gets Personal Information of the Victim
Login
Email
Password
LOGIN
Fake Webpage

Fig. 1 Phishing life cycle

keyword frequently appearing in the phishing website [13, 14]. If these techniques detect the keywords written in the English language, then they cannot detect other languages, e.g., Chinese, Hindi, Japanese, etc.

In this paper, we propose a novel approach against phishing attacks using auto-updated white-list of legitimate sites accessed by the individual user. A white-list is a set of approved legitimate domains or URLs. A white-list contains the information of those sites which are legitimate and the user wishes to access. On the other hand, a black-list contains the information of those sites which are fake and the user does not wish to access. Moreover, the white-list data is small and more accurate as compared to the black-list. In our proposed solution, the current domain is matched with predefined legitimate domains called white-list. If the user tries to open any website which is not available in the white-list, then our system checks the legitimacy of the website. Furthermore, our approach uses the hyperlink features which are extracted from the source code of a webpage to make the decision. After checking the legitimacy of a webpage using hyperlink features, the system updates the domain in the white-list. When the user accesses the same domain next time, the system only matches the domain name and IP address. Our proposed approach comprises two major components. (i) First is the domain and IP address matching module. This module matches the present domain and IP address in the white-list. Matching of IP address protects against DNS poisoning or pharming attacks. We use third-party DNS to match domain name with IP address. (ii) The second module runs if the domain is not matched with the white-list. This module examines the features from the hyperlinks to take the decision.

To summarize, the major contributions of our paper are mentioned below:

- A practical and real-time technique is proposed which can protect a user from phishing attacks on client site effectively.
- Detect the phishing attack by analysing only one effective feature (i.e., hyperlinks present in the webpage).
- Detection of zero-hour phishing attack without any prior training.
- DNS attack is also detected by matching the IP address of the suspicious site using Google Public DNS.

The remainder of this paper is organized as follows. Section 2 describes the background and previous anti-phishing approaches, their advantages and drawbacks. Section 3 presents the overview of our proposed approach and phishing detection algorithm. Section 4 presents the implementation detail, evaluation metrics, and results to judge the proposed anti-phishing system. We conclude the paper and present future scope in Section 5.

2 Related work

There have been several techniques given in the literature to detect phishing attacks. In this section, we present an overview of detection approaches against phishing attacks. 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 [13–15], and visual similarity-based techniques [16].

List-based anti-phishing techniques maintain a black-list, 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 [17]. Authors in [17] 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. As thousands of phishing webpages are created every day, updating every phishing webpage in the black-list is a challenging task. Some of the anti-phishing solutions given in the literature to protect user from phishing attacks are mentioned below:

Google provides a service for safe browsing [18] that allows the applications to verify the URLs using a list of suspicious domains which is regularly updated by Google. It is an experimental API but is used with Google Chrome and Mozilla Firefox, and it is very easy to use. The Safe Browsing Lookup API [18] allows the clients to send the suspicious URLs to Safe Browsing service which tells whether the URL is legitimate or malicious. The client API sends the URLs with GET or POST requests, which are checked using the malware and phishing lists provided by Google. Some of the shortcomings of Safe Browsing Lookup API are as follows: (i) no hashing is performed before sending URL and (ii) there is no limit on the response time by the lookup server.

A DNS-based blackhole list (DNSBL) [19] is a zone that contains resource records for the identification of hosts present in the black-list and uses the DNS protocol. The hosts undergo an IP address or domain name transformation to be encoded into DNSBL zones. There is an A record (for IPv4 address) and TXT record which gives the reason for black-listing for each entry in the DNSBL [19]. The standard value of A record contents is 127.0.0.2, but they may have other values too. DNSBLs can use the same TXT records for all entries or a different for each entry. A single DNSBL can have both IPv4 and IPv6 addresses. Domain names are less frequently used by DNSBLs than the IP addresses. The interpretation of A records and TXT are the same as the interpretation of the IPv4 DNSBLs.

PhishNet [20] examines the black-listed URLs and used some heuristics to create new variations of that URL. The author replaced top-level domains (TLD) with 3209 different TLDs resulting into child URLs which are required to be examined. To generate new URLs, clusters of host equivalence classes having the same IP address are maintained, and all combinations of these hostnames and path are then used to create new URLs. The URLs with the same directory are grouped together, and the new URLs are created by exchanging filenames within that group. If two URLs have the same directory structure with different query parts, the query part can be swapped to create new URLs.

The automated individual white-list [21] keeps records of the legitimate Login User Interfaces (LUIs) of webpages. Whenever the user submits their credentials to LUI, the white-list is checked for it and if it is not on the list, then a warning is given to the user. AIWL has two primary components. First is the white-list of legitimate LUIs. It is used to check whether a URL is familiar or suspicious so that the warning is suppressed. In the white-list, each LUI is stored as a vector that comprises of URL address, webpage feature, DNS-IP mapping. The second component is the automated white-list maintainer: It is a naive Bayes classifier which decides whether to store an LUI in the white-list. The white-list maintainer checks the number of logins for a specific LUI, if it exceeds a certain threshold, then the LUI is white-listed. We borrow the idea of maintaining the white-list for individual users from this paper.

Liu et al. [16] proposed an anti-phishing technique using visual features. This technique compares the visual similarity between the current site and the stored legitimate website. The proposed approach has taken a variety of visual features for comparison. To detect a phishing website, the system consists of two modules. The first module runs on the local server to detect suspicious URLs and keywords from email. The second module compares the visual similarity between the suspicious webpage and stored genuine webpage.

Liu et al. [22] present an approach which can detect the zero-hour phishing attack. The system extracts directly associated webpage and indirectly associated webpage. Directly associated webpages are extracted using the hyperlinks present in the source code of a webpage. Most frequent keywords (using the term frequency-inverse document frequency (TF-IDF) algorithm) including title word are searched using a reliable search engine to extract indirectly associated webpages. After extracting directly and indirectly associated webpage, the system compares the suspicious webpage with the associated webpage using link relation, ranking relation, text similarity, and layout similarity relations.

Zhang et al. [23] proposed a content-based phishing detection technique called CANTINA, which takes feature set from various fields of a webpage. The proposed technique calculates TF-IDF of the content of a website and creates a lexical signature. Then, the top five terms with highest TF-IDF values are submitted to the search engine. The top “*n*” results are used to check the legitimacy of a website, though the performance of CANTINA is affected by the language used in the website.

Xiang et al. [7] present CANTINA+, an effective, rich feature-based machine learning approach to detect phishing webpages. The rich features are taken from the various field of a webpage like Document Object Model (DOM) tree and the URL of a website. They filtered the website without login forms in the first step to decrease false positive rate. CANTINA+ achieved a true positive rate of 92 % and a false positive rate of 0.4 %.

Reddy et al. [24] present an anti-phishing technique which protects user at client side against phishing attacks. The proposed technique provides facility for the user to select specific image corresponding to every website he/she visits. Next time, when a user visits the same website and if the images do not match, then the system will alert the user. However, maintaining the image database required a lot of memory, and matching the images of suspicious sites with the stored images required a lot of time.

In a real-time environment, the detection of a phishing attack should be effective and very fast. Black-list-based approaches are very fast, but they cannot detect the zero-hour phishing attack. Visual similarity-based approaches are time consuming, require a lot of memory, and fail to detect the zero-hour attack. Heuristic-based approaches can detect zero-hour attack but their performance depends on the feature set, training data, and classifier [25, 26]. Therefore, in this paper, we proposed an approach based on auto-updated white-list to protect against phishing attacks effectively.

3 Proposed framework

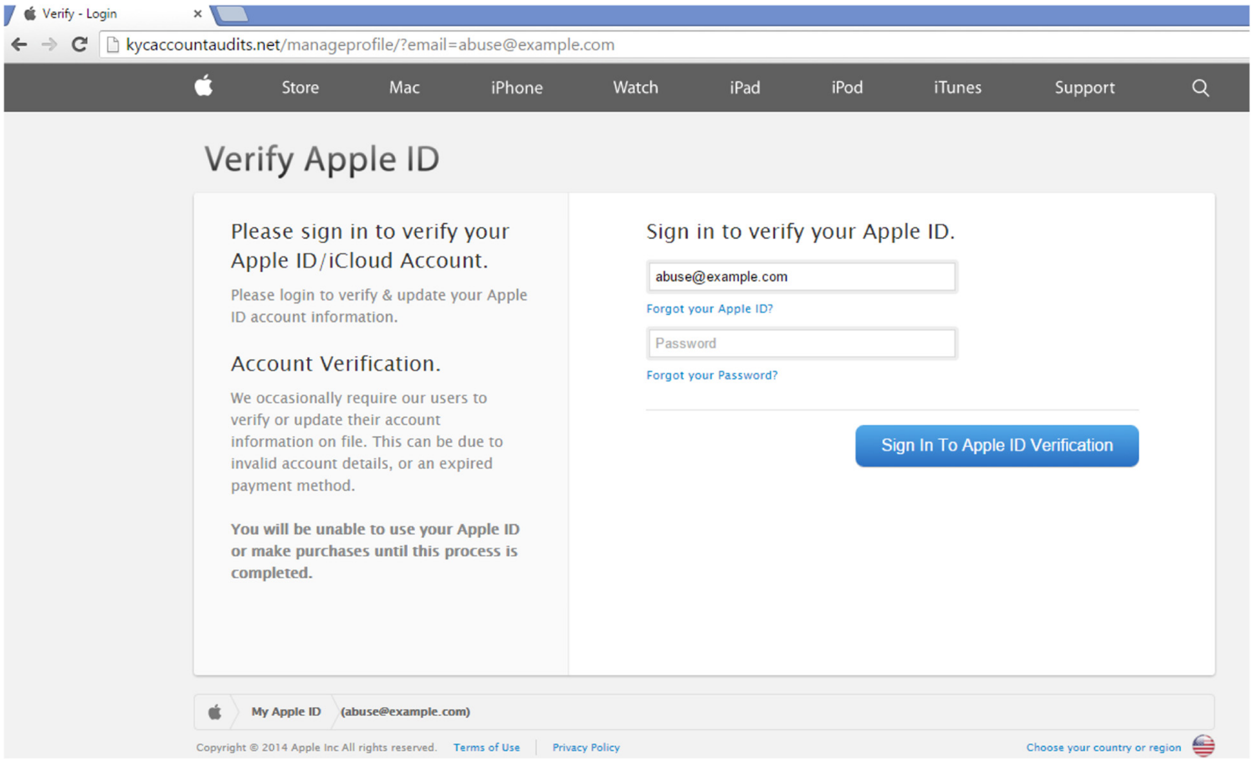
In this section, we will discuss our proposed phishing detection system.

Phishing webpages always have the same visual designs as their corresponding real websites because visual style is the most important characteristic, which is observed by a maximum number of users. However, a phishing webpage does not provide services similar to the corresponding legitimate webpage. An attacker can download the real webpage to build the phishing webpage. The phishing webpage may contain some links which redirect the users to the corresponding legitimate webpage (i.e., if a user finds any difficulty to access his/her account, he/she click on a help link then the webpage may redirect to the help section of the targeted legitimate webpage). To verify the hyperlinks relation, we have checked over 1120 phishing webpages taken from PhishTank [27] and found that 410 webpages contain direct hyperlinks to their legitimate source page.

Figure 2 shows the phishing webpage of Apple which is a verified phishing webpage taken from PhishTank [27]. The phishing page contains many links which can redirect the users to apple.com. If the user finds any difficulty in accessing the login account, then the user clicks on a link like "Forget your apple id". Subsequently, the webpage is redirected to the Apple genuine webpage. When the user signs in by entering user id and password, the webpage is then redirected to the genuine apple.com and credentials are passed to another fake domain.

3.1 System architecture

The architecture of our proposed system is divided into two modules as shown in Fig. 3. The first module is the URL and DNS matching module which contains a white-list, which is used to increase the running time and decrease the false negative rate. Our white-list maintains two parameters, domain name and corresponding IP address. Whenever a user accesses a website, then the system matches the domain name of the current website with white-list. If the domain of the current website is matched with the white-list, then the system matches the IP address to take the decision. When the user access a website which is already present in the white-list, then our system matches the IP address of the corresponding domain to check the DNS poisoning attack. Our white-list starts with zero; it means that at the beginning, there is no domain in the list and the white-list starts increasing once a user accesses the new webpages. When a user accesses a website, then there are two possibilities, either the user is accessing the website for the first time or it is already visited by the user. If the user is accessing the website for the first time, then the domain of the website will not be present in the white-list. In that case, our second module starts working. The second module is the phishing identification module, which checks whether a webpage is phishing. We extract the hyperlinks from the webpage and apply our phishing detection algorithm (the phishing detection algorithm is explained in Section 3.2). Our phishing detection algorithm examines the features from the hyperlinks to take



Verify Apple ID

Please sign in to verify your Apple ID/iCloud Account.

Please login to verify & update your Apple ID account information.

Account Verification.

We occasionally require our users to verify or update their account information on file. This can be due to invalid account details, or an expired payment method.

You will be unable to use your Apple ID or make purchases until this process is completed.

Sign in to verify your Apple ID.

Fig. 2 Phishing webpage of Apple

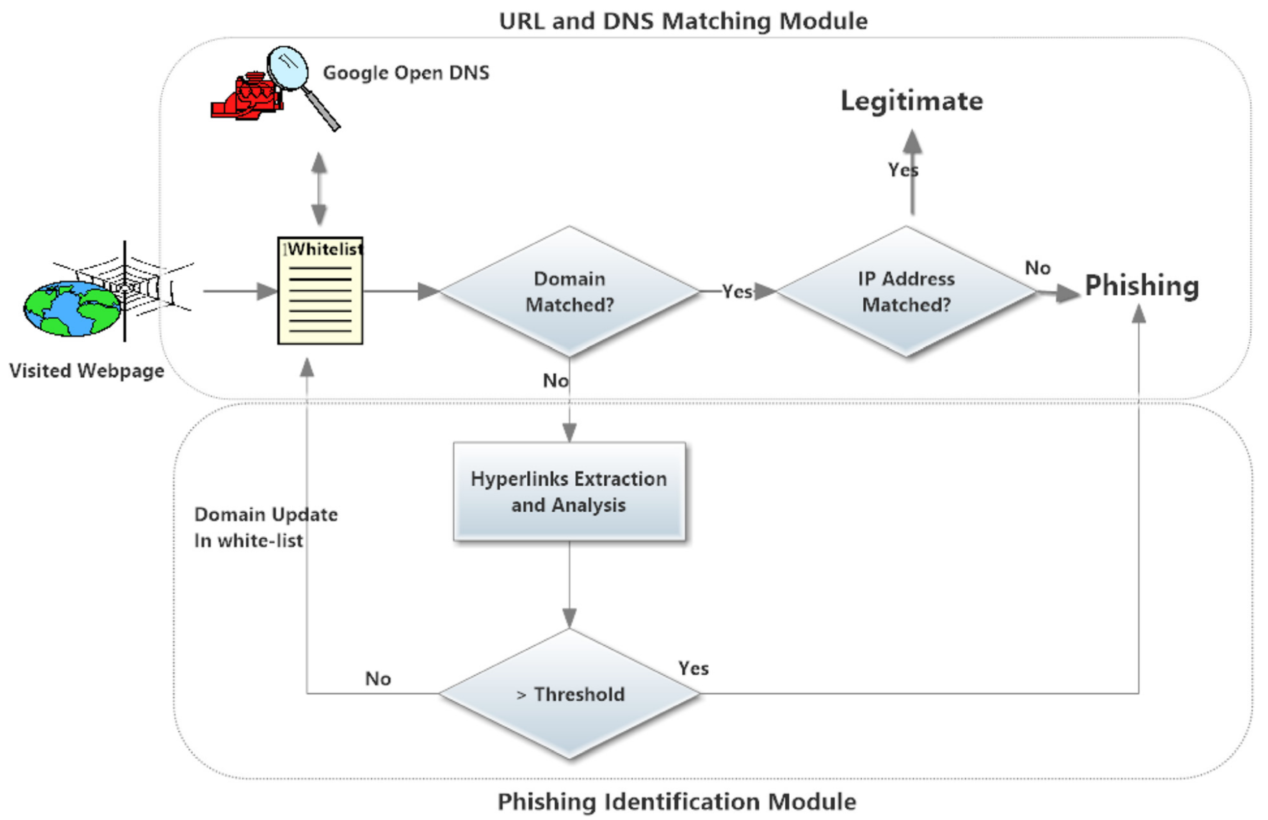
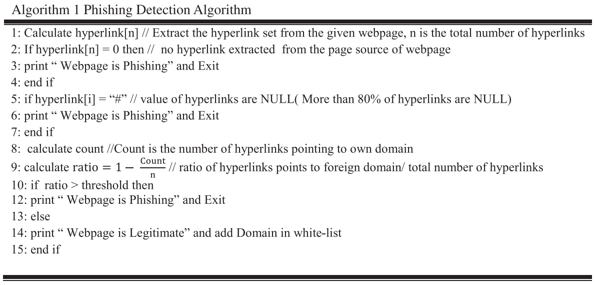

URL and DNS Matching Module
Google Open DNS
Legitimate
Yes
Whitelist
Visited Webpage
Domain Matched?
Yes
IP Address Matched?
No
Phishing
No
Hyperlinks Extraction and Analysis
Domain Update In white-list
No
> Threshold
Yes
Phishing Identification Module

Fig. 3 Architecture of the proposed anti-phishing system

the decision. After checking the legitimacy, if the website is phishing, then the system shows the warning to the user. Moreover, if the website is legitimate, then the system updates the domain in the white-list.

3.2 Phishing detection algorithm

Our proposed phishing detection algorithm takes the decision on the basis of hyperlink information extracted from the page source of the suspicious webpage. The phishing detection algorithm is shown in Algorithm 1. The motivation behind the hyperlink extraction is that the phishing webpage copies the page content from the targeted legitimate webpage and it may contain many hyperlinks in the mimicked fake webpage which generally point to the corresponding legitimate webpage such as help and forget user id or password. Sometimes phishing webpage also contains the logo of the targeted legitimate webpage. Hyperlinks are extracted from the DOM object's properties, which include href, src, and alt attributes of anchor, image, script, and link tags. There may be chances that some of the hyperlinks extracted come out to be relative links; in that case, we replaced those links by their hierarchically known absolute links. Many of the hyperlinks present in the phishing webpage may redirect to the corresponding legitimate website, but if the webpage is genuine, then it never points to a phishing webpage. Our phishing detection algorithm takes the decision based on three parameters of hyperlinks. These parameters are webpage that does not contain hyperlinks, null links present in the source code, and foreign links present in source code. We discuss all three parameters in following subsections.



Algorithm 1 Phishing Detection Algorithm

```
Calculate hyperlink[n] // Extract the hyperlink set from the given webpage, n is the total number of hyperlinks
If hyperlink[n] = 0 then //  no hyperlink extracted  from the page source of webpage
    print " Webpage is Phishing" and Exit
end if
if hyperlink[i] = "#" // value of hyperlinks are NULL( More than 80% of hyperlinks are NULL)
    print " Webpage is Phishing" and Exit
end if
calculate count //Count is the number of hyperlinks pointing to own domain
calculate ratio $= 1 - \frac{\text{Count}}{n}$ // ratio of hyperlinks points to foreign domain/ total number of hyperlinks
if  ratio > threshold then
    print " Webpage is Phishing" and Exit
else
    print " Webpage is Legitimate" and add Domain in white-list
end if
```

3.2.1 Webpage that does not contain any hyperlinks

Writing the source code in HTML is easily traceable, and anti-phishing techniques can easily extract features from the page source code. Therefore, to trick the anti-phishing techniques, an attacker can design a webpage in such a way that he/she will be able to bypass the phishing or malware detection system. An attacker uses server site script (as mentioned below) to hide the webpage source content.

```
<a href="serversite.php?id=1234">server site</a>
```

serversite.php is a script on the server that knows that id=1234 refers to www.xyz.com and could redirect the browser to it. The script would only work if it is called from the server (otherwise, someone could simply copy the source code and modify the path to serversite.php). Hackers also use the frameset (as mentioned below) to hide the source code of a webpage, and they give the reference of php file stored on the server site.

```
<frameset rows="0,* border="0">
<frame src="UntitledFrame-1" name="header" scrolling="no" noresize
target="main">
<frame name="main" src="cadastro.php">
<noframes>
</frameset>
```

From our analysis, we have seen that if the website is legitimate, we can extract at least one hyperlink. In addition, if the total links extracted from the page source are zero, then it shows that the website is a phishing site. Therefore, our phishing detection algorithm declares the webpage as phishing if no hyperlink is extracted from the page source.

3.2.2 Webpage that contains null pointer

Null pointer or null link means the hyperlink is not pointing to any other document or webpage. Null pointer in href tag is denoted by <a href="#">. The use of null pointer is to back on the same webpage after clicking on the hyperlink. Attackers create null pointer in the fake webpage for the following two reasons:

1- The first reason is to create the live hyperlink which goes nowhere. A genuine website contains lots of webpages, but a fake website contains very limited webpages. Therefore, to pretend like the legitimate webpage, the attacker creates a fake webpage and put the null values in hyperlinks. When the user scrolls the mouse over the null links, it seems that they are active.

2- Hackers use the javascript with null links to exploit the vulnerability of a web browser. An attacker creates a hyperlink in such a way that when a user scrolls the mouse over it, it shows something else rather than the actual one. In the example (as shown below), the link looks like www. example1.org, but actually, the real domain is http://example2.org. By using href="#", the link is activated and pointing the same, so the onClick attribute is able to activate.

```
<a href="#" onClick="location.href=unescape ('http://www.example1.
org%01@example2.org');" onMouseOver="window.status='example1.
org';return true;" onMouse Out="window.status=";return false;">This is
the hyperlink Text </a>
```

Therefore, our phishing detection algorithm can declare the webpage as phishing if most of the hyperlinks (greater then threshold) extracted from the page source are NULL.

3.2.3 Number of links pointing to own domain and foreign domain

After checking the no link and null link attributes, our algorithm can take the decision based on the extracted hyperlink set. In the legitimate webpage, most of the hyperlinks point to the same domain but in the phishing webpage, most of the hyperlinks point to its targeted domain (corresponding legitimate site) or some another foreign domain. Our algorithm calculates the ratio of the total number of links pointing to a foreign domain and the total links extracted from the source code of the webpage. We choose the appropriate threshold value to take the decision using various experiments. The phishing detection algorithm takes decision based on the following equation.

$$\text{Ratio} = \frac{\sum L - \mathrm{ND}_i}{\sum L} \tag{1}$$

where ND_i is the total number of links pointing to the own domain and $\sum L$ is the total number of links extracted from the page source of the suspicious webpage.

3.3 Records in the white-list

In the proposed approach, two records are kept in the white-list. First is the domain name and another is the IP address of the corresponding domain name. Whenever a user accesses the webpage for first time, the identity of the webpage is checked by the hyperlink relationship to make the decision. After making the decision, if the website is legitimate, then the system stores the detail in the white-list. Next time, when the user accesses the same domain name, the system extracts the IP address corresponding to that and matching is performed. We extract the IP address from the third party to protect the user from the DNS poisoning attack.

3.4 Third-party services

DNS cache poisoning exploits vulnerabilities in the domain name system. In this attack, hackers divert Internet traffic from the legitimate website to the phishing website. DNS is a distributed database because it becomes complex when all the Internet information is stored in a single place. To speed up the performance, one server

Table 1 Database uses to test system

S. number	Database	Number of URLs	Phishing/legitimate	URL of dataset
1	PhishTank [27]	1120	Phishing	https://www.phishtank.com
2	Alexa [31]	200	Legitimate	http://www.alexa.com/topsites
3	Stuffgate [32]	150	Legitimate	http://stuffgate.com/stuff/website/top-sites
5	Online payment service provider	55	Legitimate	http://en.wikipedia.org/wiki/List_of_online_payment_service_providers

caches the recently queried data from another server. ISP, home router, and user's personal computer also maintain the DNS cache, so they can also solve the DNS query rather than refer to DNS server again and again for the same information. Whenever the user's computer contacts to a domain name like "paypal.com," a DNS resolver contacts to the nearest DNS server. The DNS server responds with an IP address. The user's computer then connects directly to that numerical IP address corresponding to paypal.com. An attacker can change the information in the DNS cache and makes it poisoned. If the attacker gets control over DNS server and change some information, e.g., an attacker changed the IP address corresponding to paypal.com and put its own website IP address which is a phishing website, then the attacker gets all the information input by the user. In this attack, an Internet user always sees the correct URL in the web browser. Our phishing detection system compares the IP address of a suspicious site after matching the domain name. We retrieve the target's IP address by performing DNS lookup. The purpose of using DNS lookup is to resolve the issue of DNS poisoning as sometimes the attacker changes the entry in DNS cache. Attempting to match these two IP addresses will reduce the false negative rate.

4 Implementation and results

In this section, we will discuss implementation details and experimental results of our proposed phishing detection system. Our proposed system can provide the personal protection for Internet users as a browser plug-in for accessing the websites at the client side. When the user input any URL in the web browser, our phishing detection system declares the URL as either phishing or legitimate.

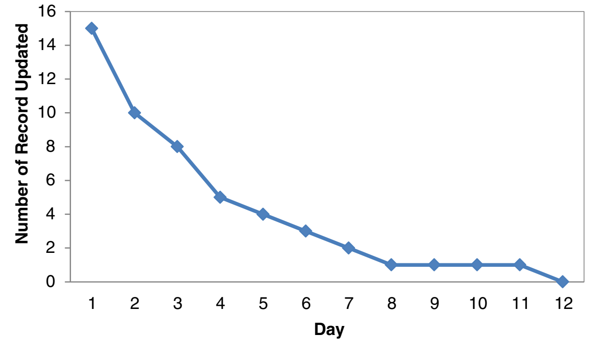


Fig. 4 Growth of white-list

4.1 Tool used

Our phishing detection system is implemented in Java platform standard edition 7 (JDK 1.7). It takes the URL of the suspicious webpage as an input to checks its legitimacy. The parent domain of the input URL is checked with the white-list. If the suspicious webpage comes out as a phishing one, the system gives warning to the user. The hyperlinks present in the webpage are extracted using Jsoup [28] by parsing the HTML file of the webpage, and a pattern matching scheme is used to obtain the links from the webpages which are not well formed. We have used Guava libraries [29] to find out the parent domains of the hyperlinks. The IP addresses of the parent domains of the suspicious webpages are found using Google Public DNS [30]. Then, the legitimacy of the suspicious webpage is verified by comparing both stored and extracted IP addresses. If Google Public does not find any IP address corresponding to the domain, then we can declare the webpage as phishing.

4.2 Dataset used

To evaluate the performance of the proposed approach, we have taken the dataset of 1525 (1120 phishing and 405 legitimate) webpages. Our dataset consists of both

Table 2 Ratio of hyperlinks pointing to a foreign domain versus total hyperlinks

Threshold (%)	Phishing webpages (%)	Legitimate webpages (%)
10	77.92	31.11
20	75.64	19.75
30	73.05	6.91
36	71.42	1.48
40	68.99	1.48
50	62.01	0.98
60	49.02	0.49
70	40.90	0.25
80	31.98	0
90	20.12	0

Table 3 Details of the hyperlink features

Webpages	Total instances	No. of webpages that contain no hyperlinks	No. of webpages that contain null links	No. of webpages pointing to a foreign domain (≥ threshold)
Phishing	1120	279	245	440
Legitimate	405	0	0	7

phishing and legitimate webpages. The phishing webpages are collected from the PhishTank [27] which is a benchmark dataset of verified phishing URLs. We have collected the phishing URLs during the period of 6 months (June 2015 to November 2015). The life of a phishing site is very short; therefore, we took phishing sites in our experiment when they are live. Legitimate webpages are taken from three different sources as shown in Table 1. Legitimate datasets consist of the variety of webpages like payment gateway, banking sites, e-commerce, blogs, forum, and social network websites.

4.3 Evolution metrics

We have calculated the true positive rate, false positive rate, true negative rate, false negative rate, and accuracy of our phishing detection system. These are the standard metrics to judge any phishing detection system. Let N_{L} denote the total number of legitimate websites and N_{P} denote the total number of phishing websites. Now, $N_{\mathrm{L}\to\mathrm{L}}$ are the legitimate websites classified as legitimate, and $N_{\mathrm{L}\to\mathrm{P}}$ are the legitimate websites misclassified as phishing. $N_{\mathrm{P}\to\mathrm{P}}$ are the phishing websites classified as phishing and $N_{\mathrm{P}\to\mathrm{L}}$ are the phishing websites misclassified as legitimate. Performance of a phishing webpage detection system can be evaluated in the following manner:

True positive rate (TPR): true positive rate is the rate of phishing websites classified as phishing out of the total phishing websites.

$$\mathrm{TPR} = \frac{N_{\mathrm{P}\to\mathrm{P}}}{N_{\mathrm{P}}} \times 100 \tag{2}$$

False positive rate (FPR): false positive rate is the rate of phishing websites classified as legitimate out of the total phishing websites.

$$\mathrm{FPR} = \frac{N_{\mathrm{P}\to\mathrm{L}}}{N_{\mathrm{P}}} \times 100 \tag{3}$$

False negative rate (FNR): false negative rate is the rate of legitimate websites classified as phishing out of the total legitimate websites.

$$\mathrm{FNR} = \frac{N_{\mathrm{L}\to\mathrm{P}}}{N_{\mathrm{L}}} \times 100 \tag{4}$$

True negative rate (TNR): true negative rate is the rate of legitimate websites classified as legitimate out of the total legitimate websites.

$$\mathrm{TNR} = \frac{N_{\mathrm{L}\to\mathrm{L}}}{N_{\mathrm{L}}} \times 100 \tag{5}$$

Accuracy (A) measures the rate of phishing and legitimate websites which are identified correctly with respect to all the websites.

$$\mathrm{Accuracy} = \frac{N_{\mathrm{L}\to\mathrm{L}} + N_{\mathrm{P}\to\mathrm{P}}}{N_{\mathrm{L}} + N_{\mathrm{P}}} \times 100 \tag{6}$$

4.4 Experiment results and discussion

Various experiments are performed to evaluate the performance of our proposed phishing detection system. We also compared our proposed system with other popular and standard anti-phishing approaches. Our system can detect the phishing webpage based on hyperlinks information. The overall true positive rate of the system is 86.02 % and false negative rate is 1.48 %.

The growth of the white-list: Our approach has used the white-list to detect the phishing URLs. Therefore, we have designed the white-list-based solution which constructed a list of legitimate sites accessed by an individual user. From our experiments, we have analyzed that a particular user do access very limited websites in a day. After verifying the webpage, the system updates the domain in the white-list. Figure 4 shows the average number of records updated day by day in the list by an individual user. Initially, our white-list is empty, and the system needs to verify each webpage accessed by a user. After verification of the webpage using the phishing detection algorithm, the system updates the webpage in the white-list. The system only needs to verify a very few webpages after some days for a particular individual user. Our white-list is used to increase the running time

Table 4 Results of our anti-phishing system

Total phishing	Total legitimate	Phishing website classified as phishing	Phishing website classified as legitimate	Legitimate website classified as legitimate	Legitimate website classified as phishing	True positive rate	False negative rate	Accuracy
1120	405	964	156	398	6	86.07 %	1.48 %	89.38 %

Table 5 Comparison between anti-phishing solutions

Approach/attack	Zero-hour protection	Embedded objects	DNS attack	Language independent
CANTINA [24]	Yes	No	No	No
CANTINA+ [29]	Yes	No	No	No
PhishNet: predictive black-listing [20]	Yes	No	No	Yes
DNS-based black-list (DNSBL) [19]	No	No	No	Yes
Automated individual white-list [21]	Yes	Yes	No	Yes
Visual signature [16]	No	Yes	No	Yes
Automatic detection of phishing target [22]	Yes	No	No	No
Our approach	Yes	Yes	Yes	Yes

and decrease the false negative rate. We have conducted the experiments with 10 participants in which three were faculty members, three were Ph.D. students, and four were graduate students. After installing the white-list in the system, the rate of the number of records updated in the white-list was maximum on the first day; gradually, it decreases with time and after few days, no new record was updated in the white-list.

Our algorithm analyzed the three parameters of a hyperlink namely no hyperlink, null hyperlinks, and ratio of hyperlinks pointing to a foreign domain out of the total hyperlink present in the webpage. To bypass the anti-phishing technique, 24.91 % (279 out of 1120) of phishing webpages contain no hyperlinks in their source code as we have discussed in Section 3 that an attacker uses server site scripting to fool anti-phishing techniques. 21.87 % (245 out of 1120) phishing webpages contain most of the null hyperlinks in the source code to pretend the active links in the webpage or the attacker uses vulnerable java script with the null links. After checking the no link and null link attributes, our algorithm computes the ratio of links pointing to a foreign domain versus the total links. Hyperlinks in more than 70 % of the remaining phishing webpages (excluding the webpages which contain the no link and null link in the source code) have pointed to another domain (threshold is 36 %) while hyperlinks in only 1.48 % of legitimate webpages pointed to the foreign domain. We have taken the threshold value of Eq. 1 experimentally as shown in Table 2, e.g., threshold 20 % means 20 % or more than 20 % of hyperlinks present in the webpage have pointed to the foreign domain. Details of hyperlink features in legitimate and phishing webpages are shown in Table 3.

Selection of the appropriate threshold to detect more number of phishing websites is a challenging task. Our aim is to design a system in which true positive rate should be high and false negative rate should be as minimum as possible. If a threshold increases, then the false negative rate decreases but at the same time, the true positive rate also decreases. Moreover, if we decrease the threshold, then the true positive rate increases but the false negative rate also increases. A good anti-phishing system requires both low false negative rate and high true positive rate. We checked it manually and adjusted the threshold to 36 %.

We have calculated the true positive rate, false negative rate, and accuracy for our anti-phishing system as shown in Table 4. The comparison of our approach with other anti-phishing approaches is shown in Table 5. Hyperlink threshold versus percent of webpages pointing to the foreign domain is shown in Fig. 5.

Although our proposed approach is very effective in dealing with the verity of phishing attacks, however, some of the recommendations are mentioned below which can be explored in the future to enhance the performance of the system.

Our phishing detection algorithm works on hyperlink features. Accuracy of detection can be improved by adding certain more features. However, extracting other features from the page source will increase the running time complexity of the system. Moreover, some features (e.g., age of domain) also required the third-party services which are not reliable. Secondly, the accuracy of detection may improve by using the machine learning to train hyperlink features instead of the phishing detection algorithm. However, using the machine learning algorithm,

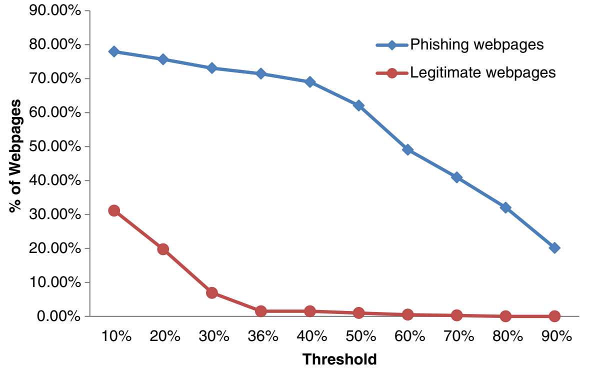


Fig. 5 Hyperlink threshold versus percent of webpages pointing to a foreign domain

reliable labeled dataset is required and performance of the system depends on the learning algorithm.

5 Conclusions

In this paper, we proposed a novel approach to protect against phishing attack using auto-updated white-list of legitimate sites accessed by the individual user. Furthermore, our approach is able to check the legitimacy of a webpage using hyperlink features. Our experimental results showed that the proposed approach is very effective in protecting against phishing attacks as it has 86.02 % true positive rate with a very less false positive rate of 1.48 %. Moreover, our proposed system is efficient to detect various other types of phishing attacks (i.e., DNS poisoning, embedded objects, zero-hour attack). Moreover, our approach is suitable for a real-time environment. In the future, the performance of the proposed system can be improved by taking the other features along with the hyperlinks; however, extracting other features will increase the running time complexity of the system.

Competing interests

The authors declare that they have no competing interests.

Authors' contributions

AKJ carried out the conception and design of the proposed phishing detection approach, acquisition of datasets, analysis and evaluation of the proposed approach and drafted the manuscript. BBG conceived of the study and participated in its design, implementation, evaluations, obtaining the results, and coordination and helped to draft the manuscript. All authors read and approved the final manuscript.

Received: 5 January 2016 Accepted: 21 April 2016
Published online: 06 May 2016

References

1. A Almomani, BB Gupta, S Atawneh, A Meulenberg, E ALmomani, A survey of phishing email filtering techniques. IEEE Commun. Surv. Tutorials **15**(4), 2070–2090 (2013)
2. A Mishra, BB Gupta, Hybrid solution to detect and filter zero-day phishing attacks, in *proceeding of Emerging Research in Computing, Information, Communication and Applications (ERCICA-14)*, Bangalore, India, August 2014
3. K Parsons, A McCormac, M Pattinson, M Butavicius, C Jerram, The design of phishing studies: challenges for researchers. Comput. Secur. (2015). doi:10.1016/j.cose.2015.02.008
4. S Sheng, B Magnien, P Kumaraguru, A Acquisti, LF Cranor, J Hong, and E Nunge, Anti-Phishing Phil: the design and evaluation of a game that teaches people not to fall for phish, in *Proceedings of the 3rd symposium on Usable privacy and security*, July 18-20, Pittsburgh, Pennsylvania, 2007 pp. 88-99
5. A Tewari, AK Jain, and BB Gupta, Recent survey of various defense mechanisms against phishing attacks. J. Inf. Privacy Sec. 1-11. **12**(1), 3–13 (2016)
6. BB Gupta, A Tewari, AK Jain, and DP Agrawal, Fighting against phishing attacks: state of the art and future challenges. Neural Comput. & Applic. 1-26 (2016). doi:10.1007/s00521-016-2275-y
7. G Xiang, J Hong, C Rose, L Cranor, Cantina+: a feature-rich machine learning framework for detecting phishing web sites. ACM Trans Inf Syst Secur (TISSEC) **14**(2), Article no. 21 (2011)
8. APWG report available at: http://www.antiphishing.org/download/document/245/APWG_Global_Phishing_Report_2H_2014.pdf. Accessed 30 Nov 2015.
9. APWG Q1-Q3 2015 Report available at: http://docs.apwg.org/reports/apwg_trends_report_q1-q3_2015.pdf. Accessed 7 Mar 2016.
10. RM Saad, A Almomani, A Altaher, BB Gupta, S Manickam, ICMPv6 flood attack detection using DENFIS algorithms. Indian J. Sci. Technol. **7**(2), 168–173 (2014)
11. Alomari, Esraa, Selvakumar Manickam, BB Gupta, Prashant Singh, and Mohammed Anbar, Design, deployment and use of HTTP-based botnet (HBB) testbed, in Advanced Communication Technology (ICACT), 2014 16th International Conference on, pp. 1265-1269. IEEE, 2014
12. 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)
13. M Moghimi, AY Varjani, New rule-based phishing detection method. Expert Syst. Appl. **53**, 231–242 (2016)
14. R Gowtham, I Krishnamurthi, A comprehensive and efficacious architecture for detecting phishing webpages. Comput. Secur. **40**, 23–37 (2014)
15. 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)
16. W Liu, X Deng, G Huang, AY Fu, An antiphishing strategy based on visual similarity assessment. IEEE Internet Comput. **10**(2), 58–65 (2006)
17. 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
18. Google safe browsing API Available at: https://developers.google.com/safe-browsing/. Accessed 30 Nov 2015.
19. DNSBL Information - Spam Database Lookup. Available at http://www.dnsbl.info. Accessed 1 May 2016
20. P Prakash, M Kumar, RR Kompella, and M Gupta, Phishnet: predictive blacklisting to detect phishing attacks, in *proceedings of the 29th conference on information communications*. Piscataway, NJ, USA, pp. 346–350, 2010
21. 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, New York, NY, USA: ACM, pp. 51–60, 2008
22. G Liu, B Qiu and L Wenyin, "Automatic detection of phishing target from phishing webpage," in proceeding of 20th International Conference on Pattern Recognition (ICPR), pp. 4153-4156, 2010
23. Y Zhang, J Hong and L Cranor, CANTINA: a content-based approach to detecting phishing websites, in Proceedings of the 16th International World Wide Web Conference (WWW2007), Banff, Alberta, Canada, May 8-12, pp. 639-648, 2007
24. VP Reddy, V Radha, M Jindal, Client side protection from phishing attack. Int J adv. Eng. Sci. Technol. **3**(1), 039–045 (2011)
25. A Jain, BB Gupta, PHISH-SAFE: URL features based phishing detection system using machine learning, in proceeding of CSI-2015, New Delhi, India, December 2015
26. V Ramanathan, H Wechsler, phishGILLNET—phishing detection methodology using probabilistic latent semantic analysis, AdaBoost, and co-training. EURASIP J Inf Security Security **2012**(1), 1–22 (2012)
27. Verified Phishing URL Available at : https://www.phishtank.com. Accessed 30 Nov 2015.
28. Jsoup HTML parser, Available at: http://jsoup.org/apidocs/org/jsoup/parser/Parser.html. Accessed 30 Nov 2015.
29. Guava libraries, Google Inc., Available at: https://github.com/google/guava. Accessed 30 Nov 2015.
30. Google Public DNS, Available at: https://developers.google.com/speed/public-dns/. Accessed 30 Nov 2015.
31. Alexa Top 500 sites, Available at: http://www.alexa.com/topsites. Accessed 30 Nov 2015.
32. Stuffgate Top 1000 sites, Available at: http://stuffgate.com/stuff/website/top-1000-sites. Accessed 30 Nov 2015.