COMPUTERS & SECURITY 62 (2016) 213–228

Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

A phish detector using lightweight search features

Gaurav Varshney [a,*], Manoj Misra [a], Pradeep K. Atrey [b]

[a] Department of Computer Science and Engineering, Indian Institute of Technology Roorkee, India
[b] Department of Computer Science, State University of New York at Albany, NY, USA

ARTICLE INFO

Article history:
Received 5 February 2016
Received in revised form 23 July 2016
Accepted 10 August 2016
Available online 16 August 2016

Keywords:
Phishing
Search engine
Lightweight
Google
Chrome

ABSTRACT

Web phishing is a well-known cyber-attack which is used by attackers to obtain vital information such as username, password, credit card number, social security number, and/or other credentials from Internet users via deception. A number of web phishing detection solutions have been proposed and implemented in the recent years. These solutions include the use of phishing black list, search engine, heuristics and machine learning, visual similarity techniques, DNS, access list and proactive phishing URLs detection based techniques. However, the current solutions are quite heavy in terms of their computational and communication requirements. Most of these solutions are dependent on third parties and require dedicated servers for their operation. It has been observed that search engine based solutions are the most lightweight and viable. This paper advances search engine based antiphishing research and presents the lightest possible phishing detection system, named the Lightweight Phish Detector (LPD). The LPD can run on client browsers for phishing detection. The development of LPD was done using the Google Chrome browser. Exhaustive testing has been performed to evaluate its accuracy and effectiveness. Comparisons are performed with currently available search engine based antiphishing approaches and other approaches that are currently used by popular browsers such as Chrome, Firefox, Internet Explorer, Netcraft toolbar and Cascaded Phish Detector. For testing, phishing sites reported from the Phishtank and normal sites available from Alexa ranking are used in the experiments. A true negative rate varying from 92.4% to 100% was obtained from the Alexa dataset of normal URLs while a true positive rate of 99.5% was recorded from the Phishtank URLs. Results show that the proposed scheme is very accurate. A competitive response time and intelligent action-response mechanism makes LPD a fast and intelligent antiphishing solution.

1. Introduction

1.1. Phishing

Phishing is a deception attack over the Internet wherein fraudsters create webpages looking similar to the webpages of the targeted organization and spread their URL links via email and/or some other communication mechanisms to the victims. Novice users, who have not yet acquired the computer knowledge to check domain names and other technical information related to these links will follow the instructions in the phishing message based on the similar look of the fraudulent website. These actions may include entering their username and

* Corresponding author.
E-mail address: gauravdtsi@gmail.com (G. Varshney).
http://dx.doi.org/10.1016/j.cose.2016.08.003
0167-4048/

password, and asking for other personal information such as their: (1) e-mail account information, (2) social networking site credentials, (3) banking information, etc. The victims unknowingly reveal their personal information, which is utilized by fraudsters for a variety of malicious purposes, including identity theft, online credit card and banking frauds etc (Varshney et al., 2012; Zhuang et al., 2012).

These websites are hosted anonymously on the attacker's own web server or, more commonly, on third party servers on a pay-per-use basis, which prevents users from easily tracing them (APWG, 2016).

1.2. Statistics

Phishing was first identified as a cyber-threat in 1996. Table 1 shows recent attack trends in the area of phishing. It can be seen that 158,544 phishing websites were functional even in Quarter 4 of 2015. The sheer amount of these websites demonstrates the difficulty of creating an accurate and complete solution to detect and thwart phishing attacks. The industry most affected by phishing is payment and financial services, as financial information allows the attacker to access the victim's monetary assets online (APWG, 2016). Research is in progress in the areas of phishing prevention, user training and phishing detection (Varshney et al., 2012).

This paper analyzes existing antiphishing schemes and identifies their drawbacks. These drawbacks include excess computational, communication and storage needs, third party dependencies and non-real-time detection of active phishing websites. Subsequently, the paper presents a lightweight phish detector (LPD) that has not only improved accuracy, but has also reduced computational and communication cost and third party dependencies in comparison to existing phishing detection systems. Furthermore, the proposed LPD is implemented on the client side and it provides real-time protection against phishing attacks.

The core idea behind LPD is: If the right set of features associated with a webpage is used to search an authentic webpage using a popular search engine, then the URL of the correct webpage should appear in the top T URLs in the search results. To check the authenticity of a webpage, LPD searches for the webpage over a popular search engine using two features: (1) the domain name in its URL and (2) the page title. This is done concurrently when the user is visiting the website. We performed a series of experiments to identify the best features to be used in a search query and we identified that the two features mentioned above give the most accurate results. Though Google search engine is used in our implementation, the functioning of LPD does not depend on any specific search engine. The major advantages of the LPD compared to existing antiphishing solutions are that: (1) it is lightweight and resource efficient, (2) it is plug and play, i.e. it has client side availability, (3) it has no dedicated resources for phishing detection, (4) it has real-time or zero hour phishing detection capabilities, (5) it has better detection accuracy, (6) it has platform independence, (7) it has faster response times, (8) it performs proactive detection and thwarting of malicious payload execution from phishing sites, and (9) it has automatic redirection capability to the corresponding authentic website after phishing website detection.

The rest of the paper is organized as follows: Section 2 describes existing phishing detection schemes and identifies their drawbacks and research gaps. Section 3 provides a detailed overview of the proposed search engine based lightweight phishing detection solution. Section 4 presents the analysis of experiments done for verification and validation. Section 5 compares LPD with the existing search engine based proposals. Section 6 concludes the paper and describes the future enhancements that are in progress.

2. Related work

2.1. Classification of existing web phishing detection schemes

Phishing detection schemes can essentially be classified into six different categories: Search Engine Based (SEB), Heuristics and Machine Learning Based (HMLB), Phishing Blacklist and White list Based (PBWB), Visual Similarity Based (VSB), DNS Based (DNSB), and Proactive Phishing URL Detection Based (PPUDB) detection schemes.

This classification is based on the underlying technique used for phishing website identification. A brief description of individual schemes is given below and an analysis of these schemes under various parameters is summarized in Table 2.

2.1.1. SEB

Search engine based techniques (Chang et al., 2013; Dunlop et al., 2010; Huh and Kim, 2012; Ramesh et al., 2014; Xiang and Hong, 2009; Xiang et al., 2011) extracts features such as text, image or URL from the websites and search for them using single or multiple search engines. The collected information is used to detect phishing.

2.1.2. HMLB

These techniques (Aburrous et al., 2009, 2010; Barraclough et al., 2013; Eshete, 2013; He et al., 2011; Ma et al., 2009; Mohammad et al., 2014; Nguyen et al., 2014; Shahriar and Zulkernine, 2012; Xu et al., 2013; Zhuang et al., 2012) extract text, image or URL-specific information from normal and abnormal websites and

Table 1 – Recent phishing attacks launched against payment service websites in the USA, where the most targeted top level domains were .com.

Parameter	Quarter 3 of 2014	Quarter 4 of 2014	Quarter 1 of 2015	Quarter 2 of 2015	Quarter 3 of 2015	Quarter 4 of 2015
Phishing sites	92473	47094	136347	253007	241140	158544
Phishing emails	163333	197252	221211	417472	395015	380280

Table 2 – Analysis of existing schemes.

Scheme	Lightweight	Client side	Response time	Third party dependence	Cons
SEB	Yes	Yes as browser add-ons	Fast	Low and only on SE	Fail if SE are compromised to rank phished websites
HMLB	No	No, need server support	Medium	Medium, depends on feature set and training algorithm	Need feature updates
PBWB	Yes, very much	Yes as browser add-ons	Fast	High, depend on third party blacklist/whitelist	Non-real time, high communication and storage cost
VSB	No	Yes as browser add-ons	Medium	High, need OS and extra hardware/software support for image capture/comparison	Image comparison is tedious
DNSB	Yes	Yes as browser add-ons	Slow	Medium, DNS	Fail if DNS poisoning happens, communication expensive and extra load on DNS server
PPUB	No	No, need server support	Fast	Medium, depends on probable URL generation algorithms	Intensive to generate and check large no of URLs, will not detect phishing URLs dissimilar to probable phishing URLs generated

use heuristics and/or machine learning algorithms for phishing detection.

2.1.3. *PBWB*

White list containing normal websites and blacklist containing anomalous websites are used to detect phishing (G. Developers, 2014a; Google, 2014; Li et al., 2014).

2.1.4. *VSB*

This technique (Chen et al., 2014; De Ryck et al., 2013; Lam et al., 2009; Mao et al., 2013) uses the visual similarity between phishing and authentic webpages and similar visual matching strategies to detect phishing.

2.1.5. *DNSB*

DNS information is used to infer phishing attacks. For example, DNS is used to detect that the IP address over which the phishing website is running is not an authentic website IP (Bin et al., 2010; Chen et al., 2011).

2.1.6. *PPUDB*

Detects probable phishing URLs by generating variants of existing authentic URLs and then verifying their involvement in phishing related activities on the web (Basnet and Sung, 2012; Marchal et al., 2012).

2.2. *Research gaps*

Some of the major research gaps identified during the study of existing schemes are:

2.2.1. *Resource utilization/dependency/availability*

The majority of phishing detection schemes are computationally or communication intensive (Aburrous et al., 2009, 2010; Barraclough et al., 2013; Basnet and Sung, 2012; Bin et al., 2010; Chen et al., 2011, 2014; De Ryck et al., 2013; Eshete, 2013; He et al., 2011; Lam et al., 2009; Ma et al., 2009; Mao et al., 2013; Marchal et al., 2012; Mohammad et al., 2014; Nguyen et al., 2014; Shahriar and Zulkernine, 2012; Xu et al., 2013; Zhuang et al., 2012). Heuristics and machine learning based or visual similarity based solutions are costly as they must extract a considerable amount of features (text, visual graphics etc.) from the webpages to detect phishing attacks. Also, implementing machine learning techniques is computationally intensive, as the client side module must perform self-learning over time. The proactive phishing detection solution is similar in that it needs deep web mining, which requires a large amount of computational resources for detecting phished URLs on the web. Such solutions cannot be implemented on the client side as browser extensions or as a lightweight IDS solution over client; they need dedicated server resources for their operations.

2.2.2. *Real time/zero hour detection*

Many solutions do not detect phishing with zero delays (Basnet and Sung, 2012; G. Developers, 2014a; Google, 2014; Li et al., 2014; Marchal et al., 2012). Examples include PBWB solutions which are dependent on a third party to provide a phishing blacklist or white list, PPUDB solutions can also miss the detection of an active phishing URL if it does not match with the probable phishing URLs which were generated and proactively mined over web. Also, solutions that use heuristics and feature extraction need frequent updates as phishers keep changing the features that they use for phishing in order to escape detection.

2.2.3. *Search engine based schemes optimizations*

Search engine based techniques (Chang et al., 2013; Dunlop et al., 2010; Huh and Kim, 2012; Ramesh et al., 2014; Xiang and Hong, 2009; Xiang et al., 2011) are new and lightweight. They can be used either on the client side or on the server side. However, these techniques are still evolving and research is needed to determine the set of features that must be used for a web search, and the optimal utilization of the results obtained from the search for phishing detection. Optimal search engine query selection and intelligent and optimized use of search engine results is still a topic of research.

2.2.4. *Intelligent action and response*

None of the existing search engine based proposals (Chang et al., 2013; Dunlop et al., 2010; Huh and Kim, 2012; Ramesh

et al., 2014; Xiang and Hong, 2009; Xiang et al., 2011) have worked in the area of active client side protection so that the malicious payload executions via phishing websites can be thwarted. Furthermore, none of the techniques have discussed how an intelligent response should be generated after detecting a phishing website so that a novice user can be redirected to an authentic website (Basnet and Sung, 2012; Bin et al., 2010; Chang et al., 2013; Chen et al., 2014; G. Developers, 2014a; Google, 2014; Huh and Kim, 2012; Li et al., 2014; Marchal et al., 2012; Nguyen et al., 2014; Ramesh et al., 2014; Xiang and Hong, 2009).

3. LPD: security model, design and implementation

This section describes the detailed design and implementation of the proposed LPD under the given assumptions and security model and requirements.

3.1. Overview

LPD is designed to run as a client side browser extension. Whenever a user U visits a URL, the LPD sniffs the URL and page title of the webpage in the background concurrently while the page is loading. The domain name is extracted from the URL and is concatenated with the page title to make a search string. The search string is then provided as an input to the Google search engine (denoted by $\mathcal{S}$) via Google web search API in the background. The domain names of the top T search results are extracted and compared with the domain name of the URL visited by U. If a match occurs, the webpage is considered authentic. Subsequently, the background of the webpage is made green and the webpage is modified to show a text notification indicating the authenticity of the page. A no match indicates the possibility of a phishing web site. The webpage background is changed to red and an alert pops up advising the user to be wary of the website and suggests closing it. The architecture and working of the LPD is shown in Fig. 1.

3.2. Assumptions

The following assumptions have been made in designing LPD:

a All authentic domain names contain a page title that can be extracted. Missing page title information on a webpage will be considered phishy as authentic websites generally have a page title and this trick can be utilized by attackers to bypass LPD phishing detection capability.
b The first domain name from the obtained search results is given as a choice for redirection via LPD to the user when he/she reaches a phishing website. This redirection might not take the user to the website that they intended to visit, but it will always ensure that the user is directed to the most appropriate and authentic website. This is because only authentic domain names can be the top search result of any search engine query and most often it will resemble the one that user intended to visit.

However the set of assumptions does not increase the false positive rate. Regarding the first assumption: In case of larger corporate sub domains redirects, each sub domain is expected to have a page title for their web page and even if they do not have a page title their primary domain name will definitely have. As LPD automatically identifies sub domains related to a primary domain name and do not check them again, they will be flagged as benign removing the possibility of sub domains being marked as phishing when they do not maintain a page title. Also the second assumption has no effect on the false positive rate as it is a part of the response mechanism.

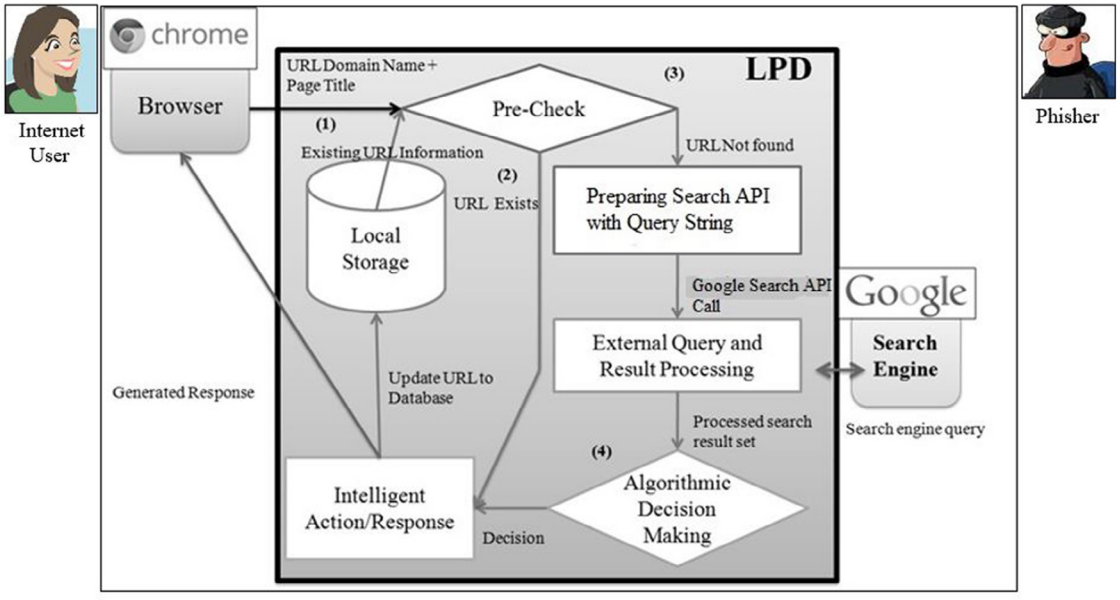


Fig. 1 – LPD architecture.

3.3. Security model and requirements

The security model outlines the honest, semi-honest/semi-malicious and malicious entities in the LPD architecture.

3.3.1. Honest entities

- User U: is a novice and honest user, and legitimately uses the LPD.
- Browser B: is honest and it neither attempts to automatically disable the LPD nor intentionally works in favor of the phishing websites.
- Search engine $\mathcal{S}$: (Google in this case) is not compromised and does not return search results to LPD in favor of phisher's interest.

3.3.2. Semi-honest/semi-malicious entities

LPD must be safeguarded against various kinds of network attacks. Hence, network attackers (denoted by $\mathcal{N}$) are considered as semi-honest/semi-malicious entities. The $\mathcal{N}$ works as an adversary, i.e. $\mathcal{N}$ can launch various network attacks such as DDoS, but it does not necessarily target the LPD. This category also includes physical attackers, denoted by $\mathcal{P}$, who try to manually disrupt LPD services by disabling its functionality over the browser as well as those who disable the functionality remotely by breaking the user's trust as a friend or a third party.

3.3.3. Malicious entities

Malicious entities in our case are phishers (denoted by Φ). They try to interrupt the LPD service and/or disable its functionality in order to deceive U. The LPD is designed to be secure against the above mentioned semi-malicious and malicious entities, as will be shown in 4.5.

Apart from these security and availability specific requirements, the LPD must also conform to the efficiency requirements in order to be considered lightweight and superior to existing search engine based solutions. We list the efficiency requirements for LPD as follows and will evaluate these requirements in Section 4.5:

- α-efficiency: A phishing detection solution is considered efficient if it detects and alerts about the loading of a phishing website on the browser before the user performs any action over it such as entering a username and password. Ideally, an efficient phishing detection solution should start its operations while the page is loading and must complete its operations before the page is fully loaded, so that the user can be alerted before he performs any actions requested by the phisher. Let T_p be the time it takes for a complete webpage to load over the browser and T_a be the time it takes for the LPD to create an alert about a phishing website, then α is defined as the difference of T_a and T_p. In order to consider the solution efficient it is preferable to have either $T_p - T_a > 0$ or $\alpha = T_a - T_p < 1000$ milliseconds, assuming that it takes approximately one second for a user to react after the page loads. A value of $\alpha < 1000$ milliseconds is required in order to call LPD α efficient.
- β-dependency: The dependency of a phishing detection solution is the number of independent services which the solution depends, apart from the services utilized at the client side. A greater value signifies that the solution is resource-intensive and will have greater dependency, thus making it less portable. A good solution must always hold a low value of β.
- γ-updatability: The frequency with which the solution must be updated in order to maintain its detection accuracy. A zero value of γ represents no updates needed in the solution and its logic. A higher value indicates a need for frequent updates. A good solution must require minimal or no updates to be considered lightweight.

3.4. Design and implementation

3.4.1. Data structure

Safe list is a data structure created and maintained by LPD that contains all domain names identified as normal by LPD to date. Safe list in implementation exists as a space-separated string of domain names. Whenever a new domain name visited by the user is identified as normal, it is concatenated with the existing list in a space-separated manner. During pre check operations, LPD splits the space-separated list and transforms it into an array, which is then searched for the current domain name being pre checked. An example representation of the safe list data structure is shown in Table 3:

3.4.2. Algorithm

The algorithm pseudo code used by LPD is given in Table 4:

Algorithm function description.:
Prompt user (Message X): Displays a pop up on the browser with an alert message X with two options: YES and NO.
UpdateBrowserTab (URL X): Updates the current browser tab URL with the given URL X and reload the tab in the browser.

3.5. LPD: implementation

LPD has been written in Javascript and Google APIs are used for its implementation. Google Chrome provides a set of APIs (G. Developers, 2014b) for developing extensions that can run over its platform. These APIs include tab related APIs that provide tab specific data for the browser, and storage APIs that allow extensions to store and access local data in the browser's memory. LPD can be easily ported to run over other browsers such as Firefox, Internet Explorer and Safari by keeping the Java script APIs intact and replacing the browser specific APIs used in Google chrome with those available in other browsers.

3.6. LPD: intelligent action and response mechanism

LPD not only detects phishing but also takes preventive measures to avoid potential damage to the victim. Unlike other search engine based solutions which only detect and alert users

Table 3 – Safe list example.

Key	Value
Whitelist	google.com facebook.com nic.in iitr.ac.in flipkart.com jabong.com

Table 4 – LPD working algorithm.

Step 1. Fetch URLs domain name and page title
a Get the URL of the webpage requested by user with respective browser APIs. b Extract domain name from URL. c Extract webpage title from DOM (Document Object Model).
Step 2. Pre-check domain name existence in safe list
a. Search domain name in safe list. i If (Found) (1) Set page background to Green. (2) Display text notification of authenticity on the webpage. (3) Exit. ii Else (1) GOTO Step 3
Step 3. Prepare search query and perform search
a Search Query = domain name + page title. b Search Results [] = Top six Google search results returned from search query. c TopSixDomainName [] = Extract six domain names from Search Results [].
Step 4. Decision making and response
a EntryFlag = 0 b For (i = 1 to i < =T i++) //Assuming that Array index starts from 1 and T is threshold value i If (TopSixDomainName [i] = =domain name) (1) EntryFlag = 1 c If (EntryFlag = =1) (1) Set page background to Green. (2) Display text notification of authenticity on the webpage. (3) Save URL s domain name in safe list (4) Exit d Else if (EntryFlag = = 0) (1) Set page background to Red. (2) Stop webpage loading (3) Prompt user ("Unsafe Website. Exception. Add to safe list?") i If (Reply = = "YES") (1) Save domain name in safe list (2) Set page background to Green. (3) Restart webpage load (4) Exit ii If (Reply = = "NO") (1) Redirect user to corresponding authentic website • UpdateBrowserTab (TopSixDomainName [1]) (2) Exit e End of If

of phishing, LPD also stops the malicious execution of the website on the user's browser immediately after detecting a phishing website. It also provides suggestions to the user about an authentic website he may wish to visit while alerting them of a phishing attack. The first (instead of only the first, T top results can be suggested in future) top search result's domain name is the website to which the user is redirected if he chooses to do so. This helps the user to reach an authentic website corresponding to the phished one with high accuracy because the search parameters of page title and domain name URL will most likely lead to the correct target domain of the authentic website. This is due to the fact that the page title used by the phisher generally corresponds to the authentic website page title, and the phished domain name being non popular (as it is a phished one) will not impact the search query in deciding the final search results, making the results solely dependent on the page title which results in the authentic domain name appearing in the top search results. Even if this is not the case, the user will reach a normal website because only a normal and long running website can appear as the first result of any search string.

3.7. *How LPD is better than its peers*

The proposed scheme is not only resource effective in terms of computational and communication cost but also overcomes the following shortcomings of the existing schemes:

a LPD not only gives a pop up alert, but also displays the actual domain name of the visited page and a text message about its authenticity. This is an enhancement over existing solutions which use only pop ups for alerts or those which does not provide a better notification of a websites authenticity. LPD also intelligently suggest the user, the URL of the actual authentic website corresponding to the phishing website. Most of these features have not been discussed or are implemented in the previous proposals (Basnet and Sung, 2012; Bin et al., 2010; Chang et al., 2013; Chen et al., 2014; G. Developers, 2014a; Google, 2014; Huh and Kim, 2012; Li et al., 2014; Marchal et al., 2012; Nguyen et al., 2014; Ramesh et al., 2014; Xiang and Hong, 2009).

b During our experiments (as described in subsection 5.3) it was identified that LPD performs all of its operations in nearly 1530 milliseconds (on an average), before the complete webpage loads over the browser. This makes LPD faster in its response time than other existing proposals. Not only does LPD detect phishing, but it also stops loading its malicious webpage contents. This avoids any ongoing malicious script execution on the client's machine via the phishing website. LPD is intelligent in the sense that it locally stores domain names of authentic (non-phishing) URLs visited by the user (in a safe list) that remain in the browser's local storage. This reduces the burden of checking the authenticity of a URL again if the user tries to visit a URL in the safe list.

c Unlike other existing schemes such as Huh et al. (Huh et al (Huh and Kim, 2012). and Ramesh et al (Ramesh et al., 2014).), instead of using complete URLs, LPD uses only the top level domain name to reduce its antiphishing check over individual URLs and sub domains of an authentic targeted domain name. This is because all URLs are assumed to be running over the top level domain's servers with a set of security policies which remains the same across all of its subdomains and URL. This reduces redundant checking of URLs of the same domain name, thus increasing storage efficiency, as only domain names are saved, rather than including sub domains and individual URLs in the safe list.

4. Experimental setup, testing, results and security analysis

Before confirming domain name + page title as the final search string, various experiments were performed: (1) to determine

the set of webpage features that should be utilized by LPD in its search query, (2) to determine T, which is the number of URLs in the top search results that should be matched with the current URL visited by the user. Finally we also did experiments to compare the effectiveness of LPD with that of other known proposals. This section covers the results of these experiments.

4.1. *Experimental setup*

All experiments were performed on an Intel Core i7 3770 CPU @ 3.40 GHz desktop computer with a 64 bit Windows 7 professional Operating system. The Google chrome 44.0.2403.130 m version was used for the experimentation. The size of the complete working LPD extension in .crx (Chrome extension archive file) format was 47 KB. The desktop was connected to the Internet with a link speed of 100 Mbps and an average network utilization of 0.02%. For the experiments, phishing sites reported over Phishtank and normal sites available from Alexa dataset were utilized.

Alexa (2016) provides web traffic data and analytics. It obtains browsing information of users by various means and analyze it for web traffic reporting thus providing rankings and other information of around 30 million websites on Internet. Alexa provides a dataset of normal websites as a raw text file. Each line of the file mentions a website rank over the Internet and its domain name separated by a comma. As Alexa dataset has already been used by previous researchers (Ramesh et al., 2014; Xiang and Hong, 2009) as a source of normal dataset, it has been utilized in our experiments to maintain transparency. Similarly, Phishtank (2015) is a community based system that verifies phishing websites. A variety of users and third parties from all around the world submits suspected phishes which are then voted by another set of users for whether they are a valid phish or not. Phishtank hence provides a real time dataset of Phishing websites running over the Internet. Phishtank provides a dataset of phishing websites in CSV file format. Each line of the file contains details about a unique phish reported. This includes : phish Id, phish URL, phish detail URL, submission time, verified status, verification time, online status and target URL. Past researchers (Ramesh et al., 2014; Xiang and Hong, 2009) have used Phishtank phish archive for their experimental testing and validation and hence we have utilized the same for comparison.

4.2. *Performance metric*

For the performance evaluation of LPD we used following three parameters: True Positive Rate (TPR), True Negative Rate (TNR) and Accuracy, derived from a 2×2 confusion matrix with two classes: predictive and actual, as shown in Table 5. A "positive" result indicates a phishing website whereas a "negative" result indicates a legitimate one.

Table 5 – Confusion matrix.

		Actual	
		Positive	Negative
Predictive	Positive	a	b
	Negative	c	d

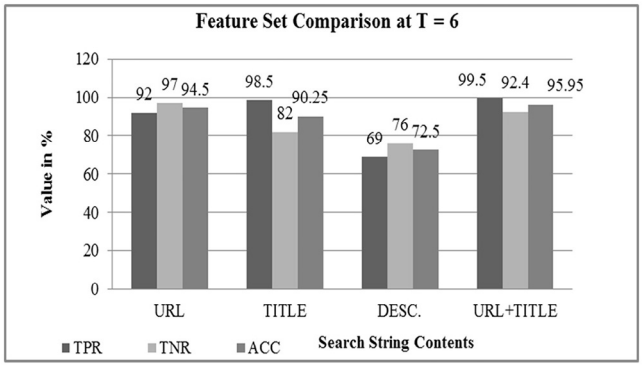


Fig. 2 – TPR, TNR, accuracy obtained with various feature set as search.

The three parameters: Sensitivity or TPR, Specificity or TNR and Accuracy, can be defined as follows:

$$TPR = \frac{a}{a+c} \quad (1)$$

$$TNR = \frac{d}{b+d} \quad (2)$$

$$Accuracy = \frac{a+d}{a+b+c+d} \quad (3)$$

Where a, b, c and d are:

a = Number of websites correctly detected as phishing via LPD (True Positive)
b = Number of normal websites incorrectly detected as phishing via LPD (False Positive)
c = Number of phishing websites incorrectly detected as normal via LPD. (False Negative)
d = Number of websites correctly detected as normal via LPD (True Negative)

4.3. *Determining search string parameters*

Experiments were performed with various combinations of URL, text and meta features as search engine queries to differentiate authentic and phishing websites. After the experimental analysis (in subsection 4.3.4), it was found that the combination of the domain name of a website and its page title provides enough information to uniquely identify an authentic website (highest accuracy for this combination in Fig. 2). The experimental analysis was performed with Google search engine as a reference. A detailed analysis of individual features as a search string parameter is given below.

4.3.1. *Domain name as a feature*

This subsection studies and analyzes the importance of domain name in the search string.

A search with domain name as search string using any search engine should show the domain name in the top results,

Table 6 – Alexa normal website dataset divided into groups G1 to G5.

Group Name	G1	G2	G3	G4	G5
Ranking (start–end)	0–500	1000–1500	10,000–10,500	200,000–200,500	500,000–500,500

if it has been crawled and indexed by the search engine and has a high number of hits. A study (Moore and Clayton, 2007) shows that the phishing sites remain up for a much shorter time because they are frequently shutdown by popular antiphishing organizations. Hence, it is nearly impossible for a phishing website to appear in the top results of a search that uses a domain name as a search query. While, using only the domain name as a search query can be an optimal solution, it can still be defeated by phishers. A phishing organization or group can keep a domain name running for a longer time by abstaining from doing any phishing attacks. They can subsequently launch phishing sites on these domain names. Thus, a search engine query with only the domain name as a search string may show the domain name of the phishing site in the top results of the search engine. Hence, the domain name cannot be used as the only search parameter.

4.3.2. *Discovering other features: the "description" meta tag feature*

As it was identified that domain name alone cannot make the best search string, other meta tags and features were tested. This subsection studies and analyzes the importance of "description" and other meta tags in the final search string.

To counter the drawbacks of using a domain name as the only search string parameter, meta features including "description", "referrer", "robots", etc. were also tested during the study and analysis. However, the results of these experiments show that these features are not very helpful in identifying an authentic website. For example, we observed that the "description" of a website may not always be closely related to its domain name. For example, when the string "Headquartered in Mumbai, India, SBI is the biggest bank in India", which is the "description" meta tag attribute value of the domain onlinesbi.com, was searched, the domain onlinesbi.com was in the top search results. However, when the same attribute with the value "A new welcome to Yahoo. The new Yahoo experience makes it easier to discover the news and information that you care about most. It's the web ordered for you" was used as a search string for the domain "yahoo.com", it was not observed in the top ten search results. This shows that "description" alone is not sufficient as a search string to link to the authentic website's domain name. This is because the value of the "description" attribute is generally a very long string whose subsets match with other popular websites and does not precisely link it to the website's domain. Also, many websites do not have the "description" attribute value in direct relation to the website's domain name, and many sites might not store the "description" meta tag information. Hence, the use of the "description" meta tag as a combination was ruled out in the first phase of feature identification. Other meta tag features were also tested but none were found to be uniquely associated with a website's identity, as they were available on all the websites.

4.3.3. *Using page title as a feature*

As "description" and other meta tags were found to be inconsistent and overly generic the need for other features which can be more unique and readily available was identified. This subsection studies and analyzes the importance of page title as a feature in the final search string.

After experimenting with many combinations of other meta features to identify their effectiveness, it was observed that a light and accurate feature is the page title. The page title alone links a webpage to its domain name in an accurate manner in most cases. For example, if we search in Google for "Yahoo India" having the page title "in.yahoo.com" we get the "in.yahoo.com" domain in the top search results (in this case, the first result). This is true for most of the websites, as the page title is small and is generally selected carefully based on a website's intended purpose. Hence, the page title precisely maps to its domain name for most of the sites. Only when the page title is itself contained in the page title of other websites does differentiation become a problem. Furthermore, the page title is maintained by all popular websites and a phishing site needs to have the same or nearly the same page title as that of the target domain name it is spoofing, in order to make others believe its authenticity. This makes the page title a good alternative search string, but it must be used in combination with another search parameter.

4.3.4. *Experiments*

The results of the experiments, with the following set of features used as the search string, are shown in Fig. 2: (1) URL domain name (2) Page title (3) Description (4) URL domain name + page title. TPR was calculated over 500 phishing websites from the Phishtank phish archive (Phishtank, 2015) and a group of normal websites from the Alexa dataset at a value of $T = 6$ (T represents the number of top search results that need to be matched). Authentic websites with an Alexa ranking from 0 to 500000 were included in various groups from G1 to G5, containing 500 websites each, as described in Table 6 (Jamey, 2015). For the phishing websites, the reporting date of all 500 websites from the Phishtank phish archive was the date on which experiments were performed in order to consider a real-time phishing detection scenario. These results show that domain name + page title as a search string identifies nearly all websites with the highest accuracy of 95.95%.

4.3.5. *Final search string*

Based on the experimental results above, the domain name concatenated with the page title was used as the final search query. A phishing domain name is not popular on a search engine such as Google and it must surely be using the page title of the targeted website. Therefore, the search will result in a list of authentic websites in the top results instead of the phishing domain.

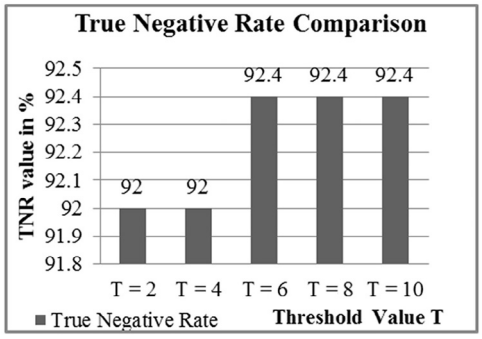


Fig. 3 – TNR comparison at various values of T.

4.4. *Determining optimum value of T*

The LPD detects phishing sites by considering the top T search results of the search query, extracting their domain name and matching them with the domain name currently being visited by the user to confirm the website's authenticity. If a complete match occurs between any of the top T domains and the domain name being visited by the user, the website is considered authentic, otherwise it is recorded as a phishing website. Finding the correct value of T is important to increase the accuracy and efficiency of any SE based antiphishing solution. Past proposals (Chang et al., 2013; Dunlop et al., 2010; Ramesh et al., 2014; Xiang and Hong, 2009; Xiang et al., 2011) used a arbitrary value of T instead of using an optimized and experimentally validated value of T. The use of arbitrary values results in increased computational cost and may lead to decreased TPR in some cases. To get optimal value of T we did following experiments for multiple values of T from $T = 2$ to $T = 10$ at intervals of 2.

4.4.1. *TNR evaluation at various values of T*

A group of websites with different Alexa rankings as per Table 6 was used during the experiments to capture the effect of ranking on the accuracy of search engine based schemes, and it can be observed from Fig. 3 that the average TNR over G1 to G5 remains stable after a value of $T = 6$. In a separate experiment using Alexa dataset top 500 URLs, we achieved an average TNR of 100% over G1-G5 at all values of T.

4.4.2. *TPR evaluation at various values of T*

Each value of T was also tested in detecting phishing websites to find the TPR. 500 phishing websites from Phishtank database were reported on the day of the experiment (websites tested were reported to Phishtank within a period of 30 minutes before the start of the experiments). A TPR of 99.5% was obtained for all values of T.

4.4.3. *Response time at various values of T*

As the values $T = 6$ to $T = 10$ can all be used, it is important to understand the effect of choosing a value of T, on the classification time. A set of experiments was performed and the time needed by LPD to report its results was recorded. There was a set of ten runs performed at varying values of T (from 2 to 10) and the average, median and mode of each were recorded. Fig. 4 shows these results in a graphical format. It can be seen from the above graph that the time taken for response via LPD is directly proportional to the value of T used for matching. As the T increases, from 2 to 10, the response time of LPD increases from 1.27 to 1.87, on average. $T = 6$ was selected as the optimal value of T as Fig. 3 shows that there is no gain in TNR beyond $T = 6$ and Fig. 4 shows that the response time increases with T and TPR remains the same at all values of T.

4.5. *LPD security analysis*

This section discusses the security of the LPD against malicious and semi-malicious entities, as stated in Section 3.3.

1. *Threat*: A malicious Φ or semi-malicious $\mathcal{N}$ entity performs service disruption attacks such as DDoS over the honest entity $\mathcal{S}$, whose services are utilized via LPD, eventually making LPD non-functional over a targeted system.

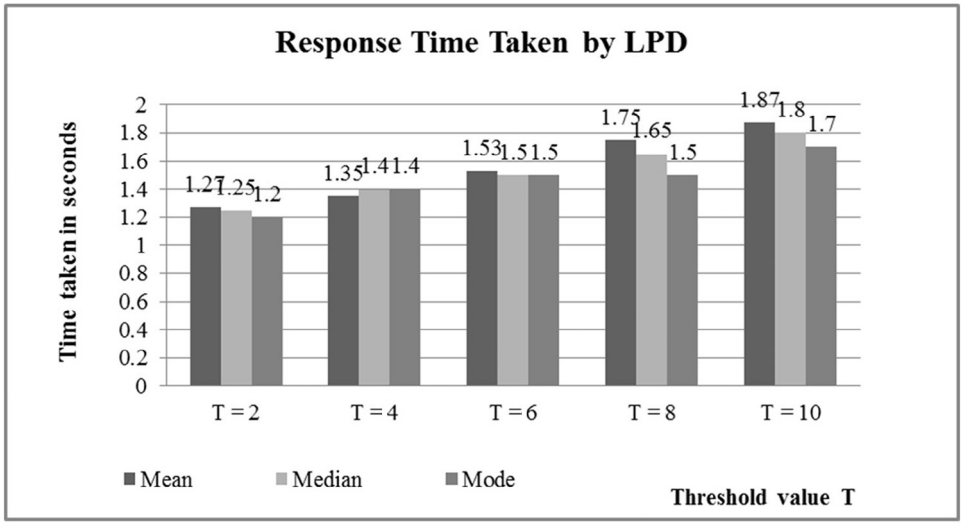


Fig. 4 – Mean, Median, Mode comparison of the response time taken by LPD.

Analysis: As the LPD does not utilize a dedicated server of its own like Netcraft, CPD and other search engine based antiphishing solutions, there is no possibility of a network based DDoS attack from either Φ or $\mathcal{N}$ that can be directly aimed to disrupt LPD functionality. However, there is a possibility of a DDoS attack over $\mathcal{S}$, which is considered to be an honest entity in the LPD environment. This can eventually affect LPD services, but in a study it has been verified that the uptime of $\mathcal{S}$ is nearly 100% at all times (Uptrends, 2015), making the scenarios of DDoS attacks on $\mathcal{S}$ affecting LPD functionality and services nearly impossible.

2. *Threat*: A semi-malicious entity $\mathcal{N}$ performs active or passive eavesdropping over the communications performed via LPD to obtain information about the honest entity U, such as the website that U is visiting, the website content that LPD extracts and communicates to $\mathcal{S}$, etc. the attacker can also affect the LPD operations by actively manipulating the data that the LPD sends or receives to $\mathcal{S}$ in order to compromise its functionality. $\mathcal{N}$ can specifically intercept the communication between $\mathcal{S}$ and B and add their website's URL at the top of the results returned by $\mathcal{S}$.

Analysis: Most of the browser-based antiphishing solutions communicate and obtain results from a dedicated server on an unencrypted channel. For example, Netcraft and CPD communicate with their dedicated servers on an unencrypted channel, which can be sniffed for information passively. Therefore $\mathcal{N}$ can also change the results communicated by the server to these extensions as they will have access to unencrypted information. Such attacks can be disastrous as no matter what the detection results are, they can be actively manipulated by $\mathcal{N}$ at specific places, as per their needs. For example: $\mathcal{N}$ can actively insert their website's URL at the top of the results returned by $\mathcal{S}$.

LPD thwarts such an attack scenario as it does not communicate with any dedicated servers of its own and the query and response via entity $\mathcal{S}$ to entity B are communicated over an https channel, making both passive eavesdropping and active manipulation to the results returned by $\mathcal{S}$ nearly impossible.

3. *Threat*: A semi-malicious entity $\mathcal{P}$ accesses the machine of the targeted honest entity U either physically or remotely and with or without his/her consent to disable the LPD functionality over browser.

Analysis: It is very likely that one could uninstall the LPD which runs as a browser extension. $\mathcal{P}$ can physically use the computer for some time with or without the permission of the entity U and can disable the LPD services. The LPD is not currently vulnerable to such attacks because there is a noticeable amount of other security extensions, for example: "*Add-ons, Settings, History Locker*" (C. E. Developer, 2015), which allows additions/deletions/enabling/disabling operations over browser extensions only to authentic users with the application of a password mechanism. Hence, the LPD cannot be disabled by a random semi-malicious entity $\mathcal{P}$ for carrying out spear phishing over the computer of a known and specific honest entity U.

4. *Threat*: A semi-malicious $\mathcal{N}$, $\mathcal{P}$ or a malicious entity Φ can access the LPD storage by subverting the honest entity B, and can obtain the targeted valuable information as per their needs or can manipulate the stored whitelist to include phishing websites to evade their detection.

Analysis: The LPD stores the safe list data structure at the local storage of the honest entity B. Any other web application or extension cannot access the other's application specific storage. Also, as whitelist data is not stored on a server or communicated through any channel, it is highly impossible for $\mathcal{N}$, $\mathcal{P}$ or Φ to access the LPD-specific data structures and manipulate them. The LPD storage of the whitelist is hence completely safe from an attack from outside.

Apart from these security specific requirements, the LPD also fulfills the outlined efficiency specific requirements to be termed as a lightweight phishing detection solution.

1 α-efficiency: LPD is α efficient. This can be proved as follows: On a sufficiently large dataset, it has been identified that 1000 ms $< T_p <$ 1700 ms. Also, as testing of the LPD suggests, $T_a <$ 1530 ms, eventually making either $T_p - T_a > 0$ or $\alpha = T_a - T_p <$ 1000.

2 β-dependency: LPD has the lowest β-dependency of 1. The LPD is from one of the least dependent solutions as it only depends on a Google web search API to complete its logic for phishing detection. Therefore, LPD is 1-dependent.

3 γ-updatability: LPD has γ-updatability of 0. γ-updatability is calculated based on four parameters described as follows:

U_f = Updates needed in the feature set with time.
U_l = Updates needed in the predefined logic.
T_t = Time needed for training or corrections over a period of time.

For LPD U_f, U_l and T_t are zero, making $\gamma = U_f + U_l + T_t = 0$. Hence, the LPD possesses zero-updatability.

5. Comparing LPD with other proposals

To test and validate LPD over client end, a set of experiments was carried out. The motive of these experiments was to evaluate the performance of LPD in terms of response time, TPR, TNR and Accuracy). A comparison of LPD's accuracy, memory and response time with other popular implementations was also done. The phishing sites tested in the experiments were taken from Phishtank in real time for a transparent evaluation and validation.

5.1. *Experiment 1. Comparison of LPD with existing SE based antiphishing proposals*

This subsection presents a comparison of LPD with the existing search engine based proposals discussed in the related work section. The comparison is based on TNR, TPR, Accuracy (Acc.), Threshold value T chosen (Thresh.), detection time (DT), client side availability (CSA), optimal feature selection (OFS), the capability of stopping malicious script execution (SMSE) and redirection of user to authentic websites corresponding to a phish (Red.).

Work	TNR %	TPR %	Acc. %	Thresh.	DT	UR / Complex.	CSA	OFS	SMSE	Red.
Ramesh et.al.	99.5	99.67	99.62	Random = 10	27272 ms on dedicated server	SE, DNS N(C1+C2)	NA	No	No	No
Huh et.al.	98	98	98	Verified = 10	~500 ms on dedicated server	Multiple SE, KNN Classifier N(3*C1+C3)	NA	No	No	No
Chang et.al.	100	92.5	96.25	Random = 30	NA. Need one Image search and Text search	SE for Image Search N(C4)	NA	No	No	No
Xiang et.al	98.05	90.06	94.05	Random = 5	NA, Authors mentioned time complexity as a concern	SE, TF-IDF N (C1+C3)	NA	No	No	No
Dunlop et.al.	100	98	99	Random = 4	4310 ms	OCR and SE N(C5 + C1)	AV	No	No	No
Xiang et.al.	99.6	90	94.8	Random =5 to 30	722565 ms	ML, Third party services and SE N(C3+C5+C1)	NA	Yes	No	No
Our work	100	99.5	**99.75**	Verified & Optimized **= 6**	**1530 ms** on client browser	**Only one SE / domain name C1**	AV for User	Yes.	Yes	Yes

Fig. 5 – Comparison with existing search engine based proposals.

The TNR considered for comparison in case of LPD is the one obtained using the top 500 ranked Alexa URLs and was 100%. This is because most of the schemes (with the exception of Xiang et al., 2011) have performed a TNR calculation on the top ranked (first 500 or in a similar small range) normal websites from Alexa's dataset. Past research proposals have not calculated TNR for low ranked websites (such as at the 500,000th position in the ranking list) and may not correctly classify most of the websites that are newly created or are low ranked. LPD gives an average TNR of 92.4% for Alexa rank as low as 500,000, which proves that LPD can detect normal websites even when they are low ranked or are newly created. LPD also provides a high TPR of 99.5%. Only the work of Ramesh et al. (2014) gives a TPR that is higher by 0.17%, but its detection time per page is very high and it also utilizes DNS resources. The accuracy of LPD is the highest among all search engine based schemes.

Another important factor of comparison is the threshold *T*. A arbitrary and large value of *T* may result in unnecessary matching of search results, thus increasing the computational needs. This may also lead to a decreased TPR, as a long-running phishing website can appear at a lower place in the search results. A small value of *T* may miss the matching of genuine domain names, thus decreasing TNR and increasing FPR. Most of the search engine based proposals use a arbitrary and large value of *T* for phishing detection. LPD uses an optimized value of *T*, which provides a better response time, and less computation needs per page with a better TNR and TPR. The detection time of LPD is higher than that of Huh and Kim (2012) only, but LPD provides higher accuracy and does not use any dedicated server resources. Other solutions take large amount of time per page for detection and therefore not suitable for use at the client end.

LPD does not use any resource other than a SE. Most of the techniques using search engine paradigm need other resources to reach a decision such as classification and machine learning (ML) algorithms, OCR, DNS and sometimes multiple SEs. LPD, through the use of best features, provides competitive accuracy with minimal resources. This also makes LPD easily portable and independent of the changes to the protocol and services provided by third parties. A comparison of utilized resources (UR) by each scheme for detecting phishing per domain name has been shown in Fig. 5 along with the complexity of the scheme based on UR. Let us consider that, a user visits a webpage *N times*, and the cost of accessing each resource for phishing detection is:

a C1: Complexity in performing a search query over a search engine.
b C2: Complexity of performing DNS search.
c C3: Complexity of using a data mining/machine learning algorithm.
d C4: Complexity in performing an image search on a search engine
e C5: Complexity of using a third party service or device.

It is easy to infer a vague comparison between the above complexities, which is as follows:

$$C1 < C2 < C3 < C4 < C5$$

The complexity (Complex.) of resource utilization in case of LPD is independent of N because LPD performs two smart operations: "*(1) It does phishing detection check on a website's domain name only and the results are used to verify individual URLs and subdomains of the same domain name, and (2) Stores the decision of the phishing check of a website's domain name in its local storage to avoid its rechecking*", thus it only needs only one phishing detection check per domain name. It is unlike other schemes which must parse each different URL/subdomain of the same domain name every time for phishing detection. As a result

complexities of other schemes is N times more in comparison to LPD as shown in Fig. 5.

Most previous works parsed and extracted features from a complete webpage (such as Ramesh et al. 2014 and Chang et al., 2013), which requires loading of the complete webpage. This may provide an attacker sufficient time to execute its malicious script on the host computer. Deep feature extraction also increases response time. LPD uses minimal features which do not require the complete page to load, thus safeguarding the browser from background malicious script executions. None of the existing proposals have focused on this while LPD solves this problem indigenously. LPD is also the first intelligent solution to provide a choice to users for redirection to a corresponding authentic website when a phishing website is detected. Fig. 5 summarizes these findings.

5.2. *Experiment 2. Comparison of LPD with other well-known antiphishing solutions at the browser's end*

In this experiment, LPD was compared with Google Chrome, Mozilla Firefox, Microsoft Internet Explorer and Netcraft antiphishing solutions at the browser's end. These are well-known solutions that detect phishing websites, and work as a part of browsers. Google Chrome and Mozilla Firefox use Google Safe Browsing API to identify a website as "phishing" or "normal". The Google Safe Browsing API is fired by the browsers whenever a user visits a URL using Chrome. The URL is searched in a phishing blacklist on the server and if it is found, API replies back to the browser that the website being visited is a phishing website and access to it should be blocked. Microsoft Internet Explorer uses the smart screen filter utility which uses a phishing white list for detection of phishing attacks. Netcraft toolbar utilizes page ranking and other heuristics to flag phishing websites to its user. We compared different schemes based on TPR, TNR and response time.

TPR was evaluated by performing experiments on a set of 50 phishing sites reported to Phishtank within a period of 30 minutes before the experiments were done in order to consider a real-time detection scenario. Every phishing URL was opened at the same time in three browsers with the security features for phishing detection enabled, and with the Netcraft toolbar installed as a browser extension in the Google Chrome browser. TPR was calculated for each one of them is shown in Fig. 6. TNR was calculated for each one of them over Alexa

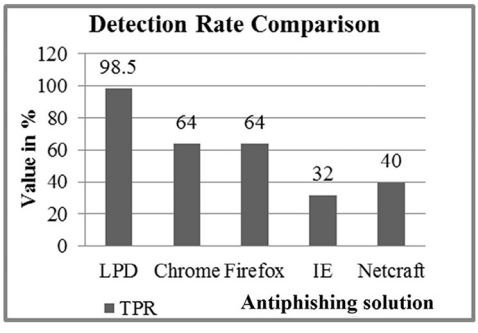


Fig. 6 – Detection rate comparison with browser based antiphishing solutions.

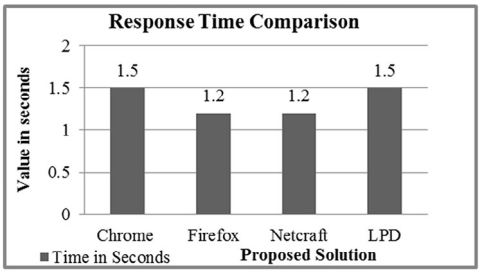


Fig. 7 – Response time needed to alert user of a phishing URL.

dataset top 1000 URLs and were found to be almost similar. Hence TNR is not used for comparison.

Response time taken by different schemes to alert a user is shown in Fig. 7. As the results may depend on the network speed and other parameters, we performed all experiments with the same system over a consistent network and computing environment. It can be seen from Fig. 7 that LPD currently takes slightly more time (300 milliseconds) than Firefox and Netcraft toolbar. LPD takes more time than Firefox and Netcraft toolbar because it performs an extra Google search in the background. While the solution for reducing the time needed for searching and parsing is under evaluation, one should note that the detection rate of LPD is noticeably higher than those of Firefox and of Netcraft toolbar. An implementation of the improved version of LPD is in progress.

It can be easily observed from the results that LPD works in real time, has high detection rate and is able to detect the latest phishing attacks or zero hour attacks with higher overall accuracy than others. One must note that the comparison was done with Chrome, Firefox, IE and Netcraft only, because open source implementations of other discussed solutions were not available.

5.3. *Experiment 3. Comparison of LPD with client cum server side phishing detection solutions based on heuristics (Cascaded Phish Detector, CPD)*

This experiment was carried out to compare the accuracy of LPD, which runs solely on the client end, with the CPD (D. Dream, 2012), which can run at the both client and server sides and uses a heuristic based approach with learning for phishing detection. The CPD was added as an extension to the Google Chrome browser and was tested in parallel with LPD over a set of 50 phishing websites reported to Phishtank within a period of 30 minutes before the experiments were done. The detection rate of both solutions was calculated and was found to be 100% for LPD, and only 50% for CPD. This clearly shows that search engine based techniques give more accurate results than heuristics based techniques, with a lower communication and computational cost and without dependency on any third party or a dedicated server.

Other than the above, two major drawbacks of CPD are: (1) it performs its phishing detection logic after complete loading of the webpage in the browser, and hence cannot stop malicious script execution via already loaded phishing websites; and (2) It does not detect phishing over all webpages; a user must manually check phishing on the page he is interested in.

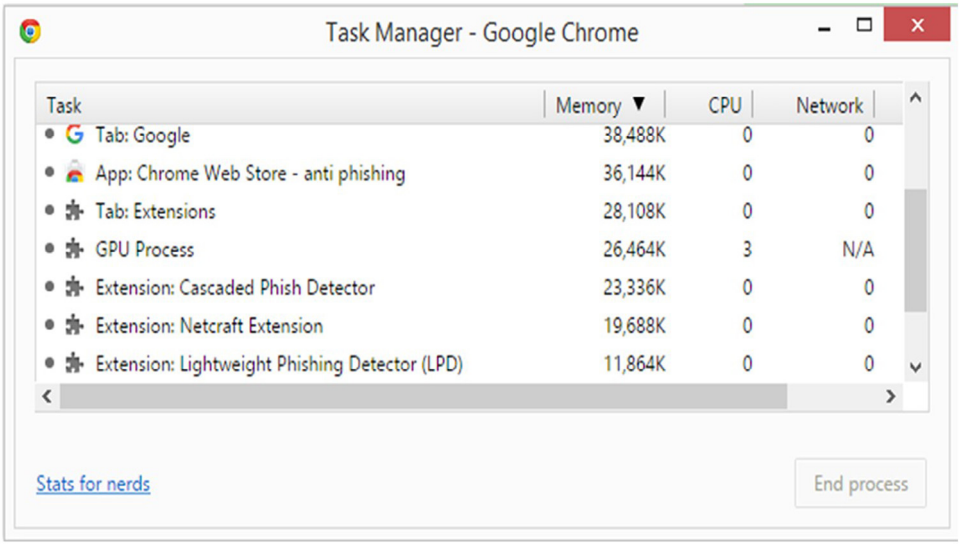


Fig. 8 – Memory utilization of antiphishing extensions over Chrome.

This logic might not be helpful for novice users who do not know when to use CPD, and on which websites. LPD and CPD have nearly same response time of 1620 ms and 1530 ms on an average as obtained during our experimentations. However, CPD takes slightly more time in communication with the dedicated server. It is important to note that LPD performs all of its operations simultaneously while the page is loading. Hence, page load time is included in the response time, whereas in the case of CPD, response time excludes the webpage load time.

5.4. *Experiment 4. Comparing the memory utilization of LPD with other browser extensions for phishing detection over Google Chrome*

This experiment was done to measure the amount of memory utilized by LPD and other known antiphishing extensions (C. M. University, 2014; Netcraft, 2015). The amount of memory utilization was recorded over the Google Chrome browser using its task manager utility. Fig. 8 shows the snapshot during the evaluation. LPD consumes the lowest memory of 11,864 K while CPD consumes 23,336 K and the Netcraft extension consumes 19,688 K. The comparison is done with only Netcraft and CPD as only they were available as a client solution over Google Chrome to verify and test. This comparison demonstrates that LPD is lightweight in its memory usage. The size of the deployables of the three solutions over Chrome is; 154 KB for CPD, 634KB for Netcraft extension and 47 KB for LPD. LPD hence stands out as the most lightly written extension with a size of only 47 KB.

6. Conclusions and future work

6.1. *Conclusions*

A study and analysis of various phishing detection schemes was performed and research gaps were identified. A Lightweight Phish Detector that uses a search engine based antiphishing technique is proposed and tested over Google Chrome. LPD answers the research gaps that are identified in Section 2, as it is:

1. Efficient in terms of computation, communication and storage cost. It requires only single search engine query per domain name. It saves the results acquired locally (which can also be stored globally if needed), thus reducing communication and computation cost per domain name. It also uses minimal and the most effective features and avoids deep content searching for every webpage. LPD does not require learning or interaction with third parties such as DNS.
2. No dedicated server resources are needed as in other proposals. LPD, being lightweight, can run as a client side module with a high level of accuracy. Its dependency on only over search engine logic makes it platform independent and easily portable.
3. LPD needs minimal updates compared to the proposals which need deep feature extraction or rely on third parties. This provides LPD zero hour or real time phishing detection capability.
4. It can be seen from the comparison with existing techniques that LPD is the most efficient and intelligent search engine based antiphishing technique. LPD is available as a client side deployable solution and can be obtained from the Google web store.
5. LPD is compared with Google Chrome, Firefox, Internet Explorer, Netcraft Toolbar and Cascaded Phish Detector antiphishing techniques and performs better in terms of detection rate, FPR, real-time phishing detection, time required to raise an alert, resource utilization, etc. The results of the various experiments show that the performance of the proposed solution is appreciable and gives new directions to lightweight and accurate phishing detection in the future.

6.2. *Future work*

LPD gives a few false positives, most of which occur when webpages are in languages other than English. This may be due to the fact that for languages other than English, Google search might not be giving the expected results and we are currently verifying this assumption.

To reduce the number of false positives (as obtained in rigorous experiments over all websites in Fig. 3), the improved LPD in the future might only parse the websites that ask for personal information through input fields. This can be easily identified by checking for the text input fields in the webpage while parsing the contents of the webpage. If no input field is found, LPD can immediately declare the webpage as normal. This is because an attacker performing phishing can only phish the users if he can obtain any vital information from them via any input fields. In the experimental study, it was identified that many false positive cases did not have any input fields. It is believed that this enhancement would reduce the number of false positives to a great extent and is currently under testing.

Further research is also going on to incorporate few other improvements to LPD in future, to make it a more better antiphishing tool. Some of which includes:

1. Creating a network of LPDs running over individual client machines and communicating a list of recently detected phishing websites to a central server. This improvement will help gather the most recent list of phishing websites in a distributed manner from user's end directly to a common central server. This list will help other techniques (such as phishing blacklist based techniques) and antiphishing organizations to detect and shutdown phishing websites in real time. The study is under progress.
2. LPD intelligent action and response mechanism can be improved in future with the inclusion of user's browsing history. In future LPD might use both: (1) the top URLs returned by the search and (2) the browsing history of the user, as an input to the decision making algorithm while offering a suggestion of the authentic URL. The browsing history might give better clues about which website user might have visited most of the times which can help provide a better recommendation of authentic website to the users in future.

Acknowledgements

We want to acknowledge the Department of CSE, IIT Roorkee and MHRD, Government of India for providing the necessary resources for pursuing this research. We would also like to thank Dr. Ramesh Chandra Joshi, Dr. M.J Nigam, Dr. Durga Toshniwal, Dr. Sandeep Garg for their suggestions and guidance.

Appendix

A working prototype of LPD can be accessed from Google Chrome Webstore at https://chrome.google.com/webstore/detail/lightweight-phishing-dete/jielndbgelplgdkgaeikbhjkippcdjcp?authuser=1. A demonstration video of its working can also be accessed at https://www.youtube.com/watch?v=cSaqnOeMPzU.

REFERENCES

Aburrous M, Hossain MA, Dahal K, Thabtah F. Intelligent phishing detection system for e-banking using fuzzy data mining. Expert Syst Appl 2010;37(12):7913–21.

Aburrous MR, Hossain A, Dahal K, Thabatah F. Modelling intelligent phishing detection system for e-banking using fuzzy data mining, in: Proceedings of the 2009 International Conference on CyberWorlds, CW '09, IEEE Computer Society, Washington, DC, USA, 2009, pp. 265–72. doi:10.1109/CW.2009.43, http://dx.doi.org/10.1109/CW.2009.43.

Alexa, The top 500 sites on the web, http://www.alexa.com/topsites; 2016 [accessed 10.01.16].

APWG, APWG phishing attack trends reports, http://www.antiphishing.org/resources/apwg-reports/; 2016 [accessed 02.01.16].

Barraclough P, Hossain M, Tahir M, Sexton G, Aslam N. Intelligent phishing detection and protection scheme for online transactions. Expert Syst Appl 2013;40(11):4697–706. <http://dx.doi.org/10.1016/j.eswa.2013.02.009>, <http://www.sciencedirect.com/science/article/pii/S0957417413001255>.

Basnet R, Sung A. Mining web to detect phishing URLs, in: Machine Learning and Applications (ICMLA), 2012 11th International Conference on, Vol. 1, 2012, pp. 568–73. doi:10.1109/ICMLA.2012.104.

Bin S, Qiaoyan W, Xiaoying L. A DNS based anti-phishing approach, in: Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on, Vol. 2, 2010, pp. 262–5. doi:10.1109/NSWCTC.2010.196.

C. E. Developer, Add-ons, settings, history locker, https://chrome.google.com/webstore/detail/addons-settings-history/gjnnmmhkhjdokoddghejcfbfnoigpbfp?hl=en-GB; 2015 [accessed 20.11.15].

C. M. University, Our cascaded learning framework for phish detection and an online demo, http://www.cs.cmu.edu/guangx/phishdetector.html; 2014 [accessed 20.12.14].

Chang EH, Chiew KL, Sze SN, Tiong WK. Phishing detection via identification of website identity, in: IT Convergence and Security (ICITCS), 2013 International Conference on, 2013, pp. 1–4. doi:10.1109/ICITCS.2013.6717870.

Chen CS, Su S-A, Hung Y-C. Protecting computer users from online frauds, US Patent 7,958,555, 2011.

Chen T-C, Stepan T, Dick S, Miller J. An anti-phishing system employing diffused information. ACM Trans Inf Syst Secur 2014;16(4):doi:10.1145/2584680. <http://doi.acm.org/10.1145/2584680>.

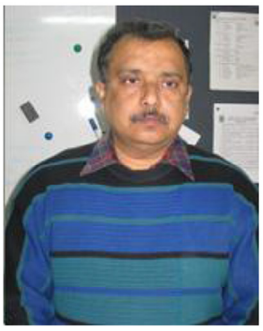
D. Dream, Cascaded phish detector by CHIMPS Lab at Carnegie Mellon, https://chrome.google.com/webstore/detail/cascaded-phish-detector/pfngencjknacaakdekdhfjgoalpjnjni?hl=en-US; 2012 [accessed 20.12.14].

De Ryck P, Nikiforakis N, Desmet L, Joosen W. Tabshots: client-side detection of tabnabbing attacks, in: Proceedings of the 8th ACM SIGSAC Symposium on Information, Computer and Communications Security, ASIA CCS '13, ACM, New York, NY, USA, 2013, pp. 447–56. doi:10.1145/2484313.2484371, http://doi.acm.org/10.1145/2484313.2484371.

Dunlop M, Groat S, Shelly D. Goldphish: using images for content-based phishing analysis, in: Internet Monitoring and

Protection (ICIMP), 2010 Fifth International Conference on, 2010, pp. 123–8. doi:10.1109/ICIMP.2010.24.
Eshete B. Effective analysis, characterization, and detection of malicious web pages, in: Proceedings of the 22Nd International Conference on World Wide Web, WWW '13 Companion, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 2013. pp. 355–60 <http://dl.acm.org/citation.cfm?id=2487788.2487942>.
G. Developers, Safe browsing API-developer guide v3, https://developers.google.com/safebrowsing/developers_guide_v3; 2014a [accessed 12.11.14].
G. Developers, Chrome java script api's, https://developer.chrome.com/extensions/api_index; 2014b [accessed 12.11.14].
Google, Safe browsing API, https://developers.google.com/safe-browsing/; 2014 [accessed 10.11.14].
He M, Horng S-J, Fan P, Khan MK, Run R-S, Lai J-L, et al. An efficient phishing webpage detector. Expert Syst Appl 2011;38(10):12018–27. <http://dx.doi.org/10.1016/j.eswa.2011.01.046>, <http://www.sciencedirect.com/science/article/pii/S0957417411000662>.
Huh JH, Kim H. Phishing detection with popular search engines: simple and effective, in: Proceedings of the 4th Canada-France MITACS Conference on Foundations and Practice of Security, FPS'11, Springer-Verlag, Berlin, Heidelberg, 2012, pp. 194–207. doi:10.1007/978-3-642-27901-0_15. http://dx.doi.org/10.1007/978-3-642-27901-0_15.
Jamey, Download alexa top 1,000,000websites for free, http://www.seobook.com/downloadalexa-top-1-000-000-websites-free; 2015 [accessed 20.09.15].
Lam I-F, Xiao W-C, Wang S-C, Chen K-T. Counteracting phishing page polymorphism: an image layout analysis approach, in: Advances in Information Security and Assurance, Springer, 2009, pp. 270–9.
Li L, Berki E, Helenius M, Ovaska S. Towards a contingency approach with whitelist- and blacklist-based anti-phishing applications: what do usability tests indicate? Behav Inform Technol 2014;33(11):1136–47.
Ma J, Saul LK, Savage S, Voelker GM. Beyond blacklists: learning to detect malicious web sites from suspicious URLs, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '09, ACM, New York, NY, USA, 2009, pp. 1245–54. doi:10.1145/1557019.1557153, http://doi.acm.org/10.1145/1557019.1557153.
Mao J, Li P, Li K, Wei T, Liang Z. Baitalarm: detecting phishing sites using similarity in fundamental visual features, in: Intelligent Networking and Collaborative Systems (INCoS), 2013 5th International Conference on, 2013, pp. 790–5. doi:10.1109/INCoS.2013.151.
Marchal S, François J, State R, Engel T. Proactive discovery of phishing related domain names, in: Research in Attacks, Intrusions, and Defenses, Springer, 2012, pp. 190–209.
Mohammad R, Thabtah F, McCluskey L. Intelligent rule-based phishing websites classification, Information security. IET 2014;8(3):153–60. doi:10.1049/iet-ifs.2013.0202.
Moore T, Clayton R. Examining the impact of website take-down on phishing, n: Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit, ACM, 2007, pp. 1–13.
Netcraft, Netcraft extension, https://chrome.google.com/webstore/detail/netcraft-extension/bmejphbfclcpmpohkggcjeibfilpamia; 2015 [accessed 24.10.15].
Nguyen LAT, To BL, Nguyen HK, Nguyen MH. A novel approach for phishing detection using URL-based heuristic, in: Computing, Management and Telecommunications (ComManTel), 2014 International Conference on, 2014, pp. 298–303. doi:10.1109/ComManTel.2014.6825621.
Phishtank, Developer information, http://www.phishtank.com/developer_info.php; 2015 [accessed 20.10.15].
Ramesh G, Krishnamurthi I, Kumar KSS. An efficacious method for detecting phishing webpages through target domain identification. Decis Support Syst 2014;61:12–22. <http://dx.doi.org/10.1016/j.dss.2014.01.002>, <http://www.sciencedirect.com/science/article/pii/S0167923614000037>.
Shahriar H, Zulkernine M. Trustworthiness testing of phishing websites: a behavior model-based approach. Future Gener Comput Syst 2012;28(8):1258–71. doi:10.1016/j.future.2011.02.001. <http://dx.doi.org/10.1016/j.future.2011.02.001>.
Uptrends, Most search engines have an excellent uptime, https://www.uptrends.com/news/mostsearch-engines-have-an-excellent-uptime; 2015 [accessed 20.12.15].
Varshney G, Joshi R, Sardana A. Personal secret information based authentication towards preventing phishing attacks. In: Meghanathan N, Nagamalai D, Chaki N, editors. Advances in computing and information technology, Vol. 176 of advances in intelligent systems and computing. Springer Berlin Heidelberg; 2012. p. 31–42 doi:10.1007/978-3-642-31513-8_4 <http://dx.doi.org/10.1007/978-3-642-31513-8_4>.
Varshney G, Sardana A, Joshi RC. Secret information display based authentication technique towards preventing phishing attacks, in: Proceedings of the International Conference on Advances in Computing, Communications and Informatics, ICACCI '12, ACM, New York, NY, USA, 2012, pp. 602–8. doi:10.1145/2345396.2345494. http://doi.acm.org/10.1145/2345396.2345494.
Xiang G, Hong JI. A hybrid phish detection approach by identity discovery and keywords retrieval, in: Proceedings of the 18th International Conference on World Wide Web, WWW '09, ACM, New York, NY, USA, 2009, pp. 571–80. doi:10.1145/1526709.1526786. http://doi.acm.org/10.1145/1526709.1526786.
Xiang G, Hong J, Rose CP, Cranor L. Cantina+: a feature-rich machine learning framework for detecting phishing web sites. ACM Trans Inf Syst Secur 2011;14(2):doi:10.1145/2019599.2019606. <http://doi.acm.org/10.1145/2019599.2019606>.
Xu L, Zhan Z, Xu S, Ye K. Cross-layer detection of malicious websites, in: Proceedings of the Third ACM Conference on Data and Application Security and Privacy, CODASPY '13, ACM, New York, NY, USA, 2013, pp. 141–52. doi:10.1145/2435349.2435366. http://doi.acm.org/10.1145/2435349.2435366.
Zhuang W, Jiang Q, Xiong T. An intelligent anti-phishing strategy model for phishing website detection, in: Distributed Computing Systems Workshops (ICDCSW), 2012 32nd International Conference on, 2012, pp. 51–6. doi:10.1109/ICDCSW.2012.66.

Gaurav Varshney is a PhD student at IIT, Roorkee. He is currently a Visiting Scholar (with Prof. Pradeep Atrey) at the Department of Computer Science, SUNY Albany. His research interests include securing users against cyber spying and developing advanced authentication schemes for thwarting phishing attacks. He has done his Master's from IIT, Roorkee in 2012 in the area of phishing prevention schemes. Gaurav has also worked with Qualcomm, India (2012–14) as an Engineer and done internships at TRDDC Labs, Pune India and Pure Testing Private Limited Noida.



Manoj Misra is a Professor and Head of Department at IIT Roorkee. Dr. Misra got his PhD from University of New Castle upon Tyne and has past experience of working as an Engineer at CMC Limited Noida, Assistant Engineer at Hindustan Aeronautic Limited at Kanpur India, Assistant Professor at HBTI Kanpur. His research interests include Distributed Computing, Performance Evaluation, Distributed Computing, and Performance Evaluation, Computer Networks, Network Security and Cyber frauds.

Pradeep K. Atrey is an Associate Professor at the State University of New York, Albany, NY, USA. He is also an Adjunct Professor at University of Ottawa, Canada. Previously he was an Associate Professor at the University of Winnipeg, Canada. He received his Ph.D. in Computer Science from the National University of Singapore. He was a Postdoctoral Researcher at the Multimedia Communications Research Laboratory, University of Ottawa, Canada. His current research interests are in the area of Security and Privacy with a focus on multimedia surveillance and privacy, multimedia security, secure-domain cloud-based large-scale multimedia analytics, and social media.