Neural Computing and Applications
https://doi.org/10.1007/s00521-020-05354-z

ORIGINAL ARTICLE

A heuristic technique to detect phishing websites using TWSVM classifier

Routhu Srinivasa Rao[1] · Alwyn Roshan Pais[2] · Pritam Anand[3]

Received: 22 January 2019 / Accepted: 8 September 2020

Abstract
Phishing websites are on the rise and are hosted on compromised domains such that legitimate behavior is embedded into the designed phishing site to overcome the detection. The traditional heuristic techniques using HTTPS, search engine, Page Ranking and WHOIS information may fail in detecting phishing sites hosted on the compromised domain. Moreover, list-based techniques fail to detect phishing sites when the target website is not in the whitelisted data. In this paper, we propose a novel heuristic technique using TWSVM to detect malicious registered phishing sites and also sites which are hosted on compromised servers, to overcome the aforementioned limitations. Our technique detects the phishing websites hosted on compromised domains by comparing the log-in page and home page of the visiting website. The hyperlink and URL-based features are used to detect phishing sites which are maliciously registered. We have used different versions of support vector machines (SVMs) for the classification of phishing websites. We found that twin support vector machine classifier (TWSVM) outperformed the other versions with a significant accuracy of 98.05% and recall of 98.33%.

Keywords Phishing · Anti-phishing · Compromised server · Heuristic · SVM · TWSVM

1 Introduction

Phishing is a kind of attack which deceives the online users with a fake website imitating the trusted website. Once the fake site is visited, then the script embedded in the fake website may steal the sensitive information such as user name, password, credit card or bank account number, etc. Nowadays, payment services, financial transactions, web-mail, etc. are mostly conducted online; therefore, they are mostly targeted by the phishers.

Phishing is on the rise in the recent years and is very hard to defend against it as it explores the weakness of human mind. According to the APWG 2016 fourth-quarter report [3], 1,220,523 number of phishing attacks were recorded in 2016 which have been confirmed to be the highest than in any year since it began monitoring in 2004.

RSA 2013 online fraud report [43] estimates a loss of over USD $5.9 billion with 450,000 phishing attacks. Another report from Kaspersky Lab revealed that their anti-phishing system was triggered 154,957,897 times on the systems of Kaspersky Lab users in 2016. Interestingly, Kaspersky reported that in 2015 there were 148,395,446 instances triggering the anti-phishing system. Note that over this period of one year there was an increase of 6,562,451 more instances than in 2015. These figures reveal that phishing attacks increased year after year accounting for billions of dollars in loss.

✉ Routhu Srinivasa Rao
routhsrinivas.cs15fv13@nitk.edu.in

Alwyn Roshan Pais
alwyn@nitk.ac.in

Pritam Anand
ltpritamanand@gmail.com

[1] Department of CSE, GMR Institute of Technology, Rajam, Andhra Pradesh 532127, India

[2] Information Security Research Lab, Department of Computer Science and Engineering, National Institute of Technology, Surathkal, Karnataka 575025, India

[3] Faculty of Mathematics and Computer Science, South Asian University, New Delhi 110021, India

Published online: 24 September 2020

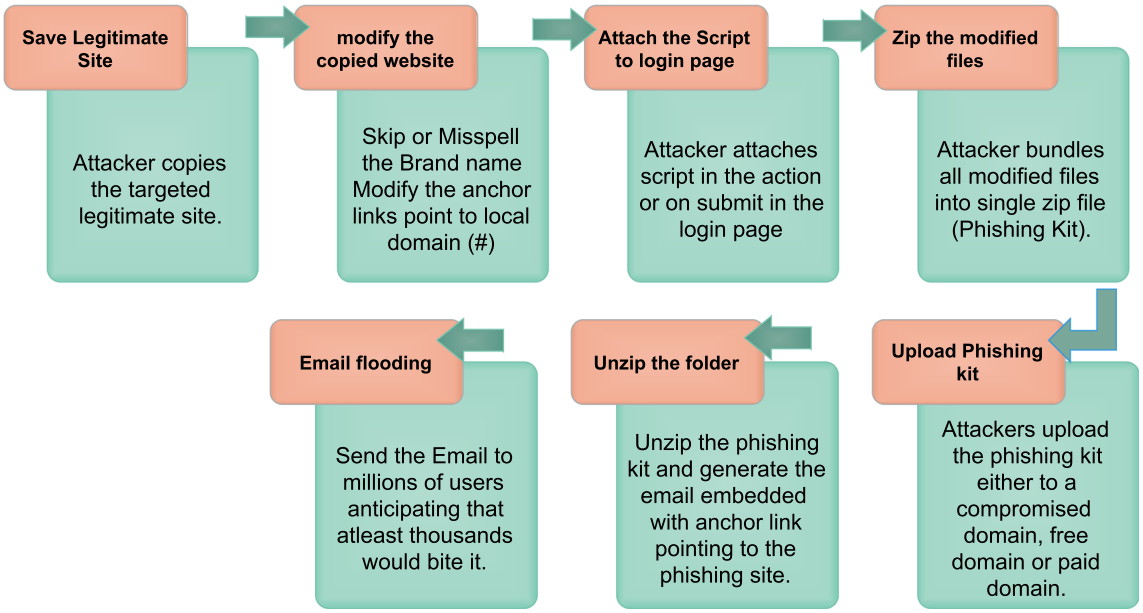


Fig. 1 Distribution of phishing kit

1.1 Phishing distribution

Attackers distribute phishing sites with the use of phishing toolkits [6, 31, 39] using the technique given in Fig. 1. The phishing toolkits are zip files or tar files which consist of copied legitimate log-in page (HTML, PHP etc) and resource files (images, CSS, logos, favicons, JavaScripts, etc). The zip file is copied to a web server for hosting his phishing site. The attacker may choose different web servers such as compromised web server, free hosting providers or paid web server. The designer of the phishing toolkit takes good care of the phishing files such that traditional anti-phishing techniques are easily bypassed. One of such bypassing tricks is to skip or misspell the brand names in the copied legitimate site. Mostly, the brand names are found in the title, URL, copyright, headers, description, etc. The other tricks include removal of anchor links that point to target legitimate URLs with null anchor links [30, 38, 40, 46]. Note that the attacker can also design the phishing site with his own code starting from scratch but it is highly expensive with respect to time and design.

In the recent APWG 2017 first-half report, APWG claimed that there is an increase in hosting phishing sites in the compromised domains [4]. According to Tyler Moore et al.'s study [32], 76% of designed phishing sites were hosted on compromised servers. If we consider the statistics of various APWG reports collected during the period from 2009 to 2014[1], it is observed that at least 70% of the phishing sites were hosted on the compromised servers for each year. From these statistics, we believed that there is a need of technique which detects the phishing sites that are hosted on compromised servers.

The advantage of hosting phishing site in the compromised server over the malicious registration is that the designed phishing site also carries the same reputation as of the compromised domain. The reputation may include page ranking, visibility of domain in search engine results, an age of domain with WHOIS database, etc. In addition to the above, there is no requirement of buying the domain or resources for maintaining their phishing page. Note that due to its drawn reputation score from the legitimate site, the phishing site will have more chance to be alive for a longer span of time than malicious registered phishing domain. While storing the designed phishing kit in the compromised server, attackers leave the compromised domain pages intact such that no suspicious behavior is observed by the owner of the legitimate site. The drawn reputation of legitimate site enables the phishing page to bypass the blacklist techniques [11, 34, 39] and third-party service-based detection techniques [9, 29, 40, 51, 54].

1.2 Limitations of existing works

There exist many techniques which compare visiting phishing site with a preapproved list of URLs or images or DOMs of target legitimate websites. These preapproved list-based techniques are called as whitelist-based mechanisms. The main limitations of these works are as follows.

Firstly, whitelist-based techniques may fail when encountered with different versions of phishing site targeting the same whitelisted legitimate site. This is due to the fact that most of the anti-phishing techniques maintain

[1] https://www.antiphishing.org/resources/apwg-reports/.

a single version of target legitimate site for the detection of phishing sites based on the similarity between the visited site and whitelisted data. The phishing site which mimics the different version of target site which is not whitelisted is left undetected by the whitelisted techniques [8, 16, 25, 27, 37, 41, 49].

Secondly, attackers misspell or skip the brand names in the designed phishing sites to bypass the whitelist techniques [20, 26, 39], which rely on the brand name extraction from the website.

Thirdly, attackers manipulate the DOM of designed phishing page such that the dissimilarity score between their phishing page and target whitelisted DOM is very high. This makes the phishing site undetectable by the anti-phishing techniques [42] depending on the layouts of websites.

1.3 Contributions

In our proposed work, we considered hybrid features including basic URL-based features, hyperlink-based features and similarity-based features for the detection of malicious registered phishing sites and also sites which are hosted on compromised servers. The features of different categories may provide information complementary to each other for the classification. Hence, our work focused on proposing hybrid features for the classification of phishing sites with superior accuracy.

Like [9, 15, 17, 29, 33], we have also used the standard SVM classifier for the classification of phishing sites. But, apart from this, we have also tested PSVM and TWSVM for the classification of the phishing websites using the same dataset. We have found that TWSVM model outperforms among other versions of SVM by achieving 98.05% accuracy and 98.33% recall with the dataset of 5500 phishing sites and 5500 legitimate sites.

The basic idea of our work is that any webpage in a website has a certain similarity between webpages with any other webpage of the website. If a webpage is hosted on compromised server, then there exists a very large dissimilarity between the webpages in the website. This abnormal behavior is used as basis to detect the phishing sites hosted on compromised server. Additionally, hyperlink-based features are used to detect the remaining phishing sites which are registered maliciously. Our model emphasizes the phishing detection without relying on preapproved or unapproved database, prior knowledge of the user such as web history, third-party services such as search engine, page ranking, WHOIS, etc.

The contributions of our proposed work are summarized as follows.

- Proposed efficient similarity-based features for detection of phishing sites hosted on compromised servers.
- Proposed TWSVM-based phishing detection (this is the first attempt using TWSVM) for the websites hosted on compromised servers and other malicious registered sites.
- Proposed hyperlink-based features for detecting single/common page phishing sites and image-based phishing sites
- Proposed technique is robust to different URL masquerade techniques and can also detect zero-day phishing attacks.

The paper is organized as follows. The review of existing works related to identifying phishing websites is discussed in Sect. 2. The proposed work is explained in Sect. 3. A brief overview of machine learning classifiers is discussed in Sect. 4. Experimentation and results are given in Sect. 5. The discussion on effectiveness and limitations of the proposed work is discussed in Sect. 6. Finally, we conclude the paper in Sect. 7.

2 Related work

There exist many anti-phishing techniques which rely on URL, source code, third-party services (page rank, search engine) and machine learning to counter the phishing websites. In this section, we discuss some of the latest and most popular techniques that are used in detecting the phishing websites. Also, we group the existing works into three categories.

- **Third-party-based techniques:** These techniques use third-party services like search engine, page rank and WHOIS for the detection of phishing websites. There exists the decent number of works which rely on third-party services, and some of them are given as follows: Cantina [54] and Cantina+ [51] techniques use search engine results when queried with related text of target brand for the detection of phishing sites. Cantina uses famous term frequency–inverse document frequency algorithm (TF–IDF) for extracting key descriptors of the visited website to form search query. The query is fed to the search engine to output the search results. If the suspicious URL is found in the search results, it is classified as legitimate else phishing. Similarly, Cantina+ uses title and domain of the visited website as search query which is fed to the search engine. The URL of visited website is searched in the results to check legitimacy of the given website. The authors also included some additional heuristic features which are extracted from URL and hyperlinks of the website. Recently, a similar work [48] to Cantina+ has been published, and it claims only title and domain name as search query is sufficient to detect phishing sites with

significant accuracy of above 99%.

Jain and Gupta [19] proposed a two-level search engine-based technique to detect phishing sites. The authors used search engine mechanism similar to [48] at level 1 to detect phishing sites and hyperlink-based features at the other level to detect newly registered and unpopular legitimate domains. The hyperlink features include the ratio of local (L) to null (#) hyperlinks (N) out of total number of hyperlinks (T) in the website. If the ratio of L to T is high and the ratio of N to T is low, then the technique classifies the suspicious website as legitimate site. But, the technique fails when the designed phishing site contains all the hyperlinks assigned with a single common local page, leading to L/T as 1. These types of hyperlinks are also termed as common hyperlinks [40].

Chiew et al. [9] proposed a hybrid method which extracts logo from a visiting website using machine learning algorithm. The extracted logo is fed to Google image search to check the legitimacy of the website. The visited URL is checked with search engine results, and if it is found, it is classified as legitimate site else phishing. An efficient algorithm is required for logo extraction and also fails to detect when a phishing site with no logo is encountered.

- **Visual similarity-based techniques:** These techniques compare the visual resemblance between the visited site and prestored database of legitimate site resources. If the similarity between the both exceeds a certain threshold, then the visited site is classified as phishing else legitimate. The visual resemblance resources include screenshots, pixels, mages, font styles, page layouts, text content, font colors , logos, etc. Some of these techniques are given as follows: Rosiello et al [41] proposed document object model (DOM) similarity-based approach named as DOMAntiPhish for the detection of phishing sites. Authors maintained whitelist of DOMs of target website which are used for similarity calculation between the DOM of visited website and stored DOMs. This technique fails when DOM obfuscation techniques like insertion of empty tags or deletion of some unimportant tags are applied to the designed phishing site.

  Hara et al [16] proposed a visual similarity-based technique which uses whitelist of screenshots of targeted websites for the detection of phishing sites. The screenshot of visited website is compared with whitelisted screenshots for calculating the similarity score between the both. If the similarity score is above a threshold and both the URLs are unique, then the visited site is classified as phishing else legitimate. The limitation of this approach is it has very high false positive rate of 17.5%.

  PhishZoo [2] is an anti-phishing technique which uses trusted profiles for the detection of phishing websites. It detects the phishing sites which look like legitimate sites by comparing the visited website content against stored trusted profiles. It fails when the phishing site is significantly different from the target website. The trusted profiles include URL, SSL, images and scripts.

  Mao et al [25] proposed a technique called phishing alarm which uses whitelisted cascaded style sheets (CSS) between the visited page and target pages for determining the phishing status of the website. Authors claimed that CSS features are very hard to be evaded by the attackers because of their property that defines the visual structure to the HTML elements in the webpage. The limitation of this technique is it fails when the entire HTML content is replaced with an image in the website. Also, phishing sites may go undetected when the target website is not whitelisted.

  AuntieTuna [5] is a technique which stores cryptographic hashes of small chunks of data as whitelisted data for every target website. The small chunks of data are chosen based on the text between $<p>$ and $<div>$ tags. For any visiting website, the number of matchings of hashes between the visited and target website is calculated and if it is greater than a threshold, then it is classified as phishing site. As the technique is relying on cryptographic hash, a small change in the chunk of data is sufficient to bypass the technique.

  Zhang et al. [53] proposed a Bayesian approach which detects phishing sites by comparing the visual and textual contents between the target website and visited website. The textual contents include plaintext extracted from DOM after removing HTML tags and stemming process. The visual contents include extraction of images from HTML and are forwarded to normalization for the generation of visual signatures with color and coordinate features. Earth mover's distance is used to calculate the similarity score between the protected website and visited website. If the score is above a threshold, then it is classified as phishing else legitimate.

- **Heuristic and machine learning techniques:** These techniques extract features from URL, source code of the visited website which provides distinction between the phishing and legitimate behavior. The URL features include length of URL, number of dots, presence of special symbols$(@, , -, *)$ in the URL. Some of the techniques extract source code features from hyperlinks, textual content, copyright, title, DOM, images. These features are fed to the machine learning algorithms for the classification of phishing and legitimate sites. Some of the heuristic and machine learning

techniques are as follows: PhishNet [34] is a blacklist technique which predicts the variants of phishing URLs by applying heuristics of combinations of known phishing URLs. The heuristics include interchange of top-level domain (TLD), directory structure, IP address, query, etc. A unique phishing URL with randomized string in path fails in detection. With this technique, original legitimate pages in compromised domain are also classified as phishing pages.

Moghimi and Varjani [29] proposed a rule-based technique to detect banking websites. The technique extracts features from hyperlinks and URL of the visited website. The rules are extracted from the dataset of extracted features by applying C4.5 rule induction algorithm which are further used for detecting phishing sites. The authors concentrate on detecting phishing sites targeting banking sites only and also fail when # or same page anchor links are used.

Rao and Pais [40] proposed machine learning framework with rich set of features including URL-, hyperlink- and third-party-based features for the detection of phishing sites. The authors used oblique random forest as classifier for the detection process and achieved an accuracy of 99.5%.

Our work falls under this category as it extracts the features from the source code and uses machine learning algorithms for the classification. In the next section, we provide the review of limitations of existing solutions compared to our work.

2.1 Review

The existing visual similarity techniques use a database of resources of target websites for calculating similarity with visited websites. Unfortunately, this process of comparison with all target websites results in high time and space complexity, making visual similarity-based techniques not adaptable at client side. Our work targets only home page and visiting page to calculate the similarity score, which is further used for classification of phishing sites.

The existing third-party-based techniques like search engine- and page ranking-based techniques result in processing delay due to the dependence on third-party services. Moreover, these techniques fail to detect phishing sites which are hosted on compromised web servers because some of the compromised domains are well indexed by the search engines with good page rank and greater age of domain. Our technique is neither dependent on third-party services nor uses database of target website's resources, which makes our technique more robust to phishing sites which use compromised web servers for hosting purpose. Note that we have also included additional hyperlink-based features to counter phishing sites which are registered maliciously. The summary of existing techniques with respect to various attributes such as search engine independence (SEI), page ranking independence (PRI), database independence (DI), brief description and limitation of the work are given in Table 1.

The table also includes our work with aforementioned attributes too so that comparison of other works can be observed.

3 Proposed work

Attackers copy the phishing page into a compromised server so that the purchase of domain and computation resources is avoided. It also gives an advantage of carrying the same reputation of compromised domain and is able to bypass search engine- and URL-based techniques. Due to this behavior, the DOM of the phishing page looks significantly different from that of compromised domain pages.

Based on the above observation, the basic idea of our work is to find the degree of dissimilarity between the log-in page of phishing site and home page of the compromised domain. If the website is not compromised, then the similarity between any two pages in the website will have a certain similarity.

We divide our work into two phases. Firstly, we extract the hybrid features consisting of basic URL based, similarity based and hyperlink based for the generation of feature vectors. Secondly, we apply the machine learning algorithm on the generated feature vectors for performing the classification on the extracted data. The reason to choose the machine learning classifier is the greater accuracy achievement in the real-time applications.

The flowchart and working algorithm of our work are given in Fig. 2 and Table 2, respectively. Our model takes input as suspicious URL and alerts the user with the status of URL. The model extracts the document object model (DOM) from the suspicious URL using Selenium WebDriver. If the URL is found to be a non-log-in page, then the detection process is skipped; otherwise, the DOM is fed to feature extraction module. The home page of the given URL is visited to extract the similarity-based features for the identification of phishing sites hosted on compromised servers. For the calculation of similarity scores between the log-in page and home page of suspicious URL, Jaccard similarity coefficients are calculated. The extracted features are combined to form a feature vector which is fed to the machine learning algorithm to get the status of the URL as either legitimate or phishing. To ease our discussion, we define the following terms.

Table 1 Summary and comparison of existing works with our work

Technique	Year	Brief description	SEI	PRI	DI	Limitations
Rosiello et al. [41]	2007	It uses DOM of suspicious website with target websites DOMs	Yes	Yes	No	It fails when empty tags are inserted or unimportant tags deleted from the suspicious site
Hara et al. [16]	2009	It uses screenshot of suspicious site for comparison with list of screenshots of target websites for the detection process	Yes	Yes	No	Very high false positive rate (18%)
PhishNet [34]	2010	It predicts phishing URLs based on the simple combinations of known phishing URLs	Yes	Yes	No	It fails when a unique phishing URL is encountered and also might result in high false positives when compromised domains are encountered
Cantina [54] and Cantina+ [51]	2007, 2011	A content-based technique which uses TF–IDF algorithm, heuristic features and Google search engine results for the reduction in false positive rate	No	No	Yes	It results in search engine delay and fails to detect websites which replace text with images
PhishZoo [2]	2011	Content similarity-based technique which uses stored profiles of target sites for phishing detection	Yes	Yes	No	It detects the phishing sites which exactly looks like legitimate sites and fails when it is significantly different from target site
Chiew et al. [9]	2015	It uses logo as a query to Google image search for identifying the phishing status	No	Yes	Yes	An efficient algorithm is needed for extraction of logo from the website
AuntieTuna [5]	2016	It uses cryptographic hash of chunks of data for the comparison with chunks of data related to target websites	Yes	Yes	No	It fails even if there exists small change in the chunks of data and also it works only when attacker clones the legitimate content for designing the phishing page
Moghimi and Varjani [29]	2016	A rule-based phishing detection which relies on hyperlinks and web address-based features	Yes	Yes	Yes	It fails when the phishing website is not cloned from the target website
Varshney et al. [48]	2016	A lightweight phishing detection which uses search engine results with title and web address as a search query	No	Yes	Yes	It may fail to detect phishing sites which are newly registered or sites hosted on compromised web servers
Phishing-Alarm [25]	2017	It detects phishing sites based on similarity of visual appearances (CSS) between the webpages	Yes	Yes	No	It fails when a phishing site is not cloned and redesigned
Jain & Gupta [19]	2017	A two-level search engine-based technique which uses search engine at level 1 with the search query as domain + title and hyperlink-based features at the other level to increase the accuracy and TNR	No	Yes	Yes	The hyperlinks matching filter reduces the effect of search engine filter. This technique might improve TNR but decreases TPR when it encounters phishing site with common hyperlinks
Rao and Pais [40]	2018	A machine learning framework with rich set of hyperlink-, URL- and third-party-based features	No	No	Yes	Delay due to third-party-based and broken links features
Our work	2018	It is based on content and hyperlink similarity between log-in and home page of website	Yes	Yes	Yes	It gives false alarm for the legitimate sites which are hosted on free web hosting services

- **S** —suspicious URL to be checked.
- **H** —home page of S.
- **TW** —the target website which is imitated by the attacker to perform phishing attack.
- **POCS** —phishing sites hosted on compromised server.

Our proposed model is adaptable to any URL for phishing detection but to ease the computation cost on non log-in pages, we excluded non-log-in pages for experimentation. In this work, our model is applied on log-in pages of given URL since majority of the phishing websites use log-in pages for stealing the sensitive information. Normally, the log-in page is identified at three locations: 1) visiting page of URL, 2) modal window and 3) IFrames. Attackers use IFrames for loading the imitated target website's content such that the anti-phishing techniques which rely on source code of websites are easily bypassed. This is due to the fact that content in IFrame is hidden from the source code of the website. Hence, we used Selenium WebDriver to identify

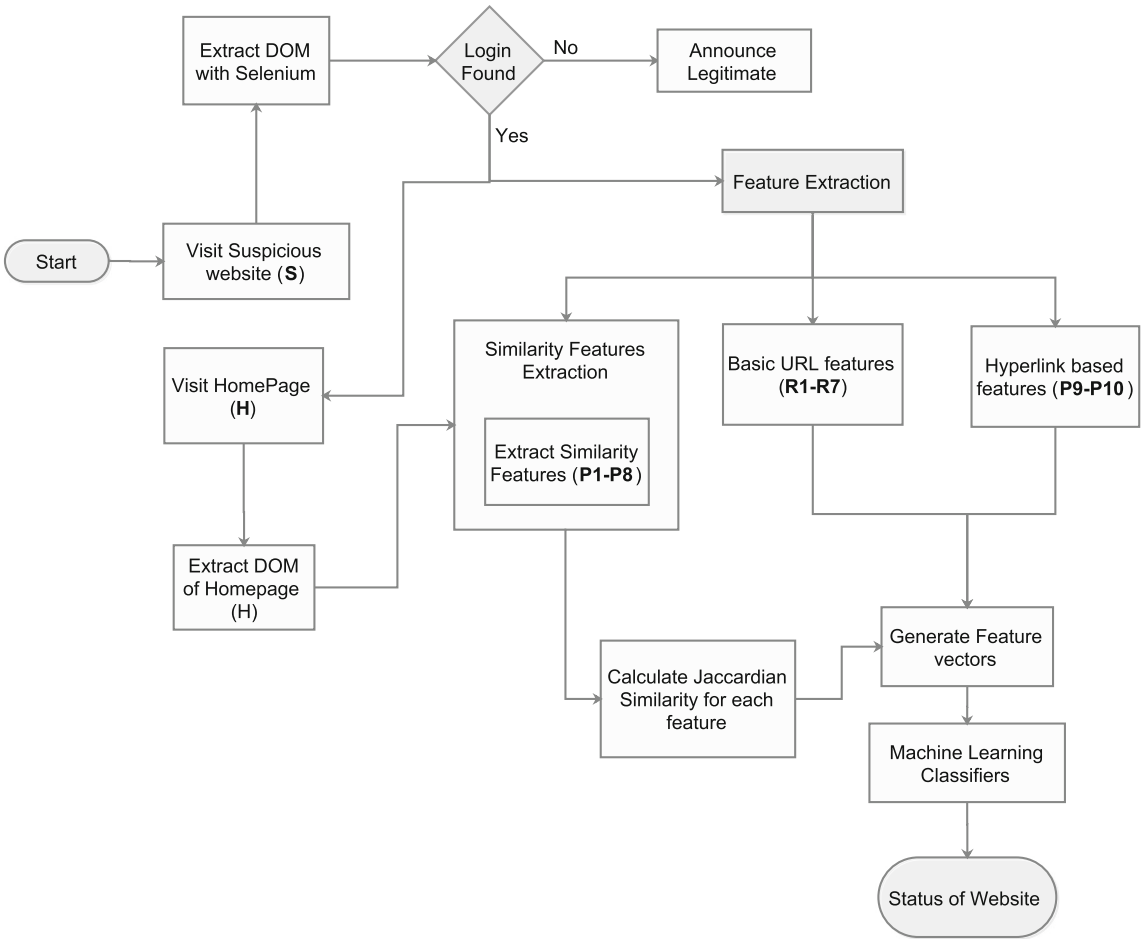


Fig. 2 Flowchart of the proposed work

Table 2 Working algorithm of our proposed model

Step 1 : Visit the given suspicious URL (S)

Step 2: Extract the document object model (DOM) using Selenium WebDriver

Step 3: Feed the DOM to Jsoup API for parsing the source code.

Step 4: Check for the presence of log-in form; if present, go to the next step, else announce the page as legitimate.

Step 5: Extract the basic URL features (R1–R7) from suspicious URL (S), hyperlink-based features (P9, P10) and similarity-based features (P1–P8) from DOM of S.

Step 7: With the same Webdriver, visit the home page (H) and extract the similarity-based features (P1–P8) from DOM of H

Step 8: Calculate the Jaccard similarity coefficient for each of the similarity features (P1–P8) of *H* and *S*.

Given two sets A and B where A, B $\in$ single feature in H and S, respectively, then

$$J(A,B) = \frac{|A \cap B|}{|A|+|B|-|A \cap B|} \qquad (1)$$

where J(A,B)$\in$ [0,1] and J(A,B)=1 when both A and B are empty.

Step 9: Generate the feature vectors based on R1–R5 and P1–P10.

Step 10: Feed these feature vectors of all legitimate and phishing sites to machine learning classifiers such as SVM, TWSVM, PSVM, etc.

Step 11: Do the classification with tenfold cross-validation and tuned parameters to get the status of website as phishing or legitimate

the log-in page at all above locations. The page is categorized as a log-in page when it contains at least one input field with `type=password`, otherwise categorized as non-log-in page. All the log-in pages undergo feature extraction phase, and for the non-log-in pages, the

detection process is skipped as the user does not have any way to give his/her sensitive information.

3.1 Feature extraction

In this module, we extract the features from the source code of log-in page and its respective home page. These features are structured into two categories.

- Basic URL features (R1–R7)
- Proposed features
  - Similarity-based features (P1–P8)
  - Hyperlink-based features (P9–P10)

3.1.1 Basic URL features

These features are proposed in the existing literature [17, 30, 51], which are extracted from the URL of the log-in page. They are as follows.

(1) *R1: number of dots in hostname:* This feature counts the number of dots in the hostname of the given URL. The rationale behind this feature is that attackers append the brand name of the target website to his phishing URL by the use of dots in hostname or pathname. For example, http://www.iamphishurl.com/pathname/index.php is a phishing URL targeting ebay, then the attacker adds the ebay brand name in the hostname such that legitimate behavior is observed by the online user, i.e., http://www.signin.ebay.iamphishurl.com/pathname/index.php The existing work [17, 51] used the count of dots in the entire URL but we observed that brand name importance is greater in the hostname rather than file path.

(2) *R2: Presence of @ in URL :* This feature checks the presence of special symbol @ in the visited URL. If so, then it is set to true (1), else set to false (0). Attacker inserts special symbol like @ in the phishing URL prefixed with legitimate address and postfixed with the actual phishing URL. The reason for using this is that when the browser encounters @ symbol in the URL, it ignores the text prefixed before the @ and redirects to the URL specified after the @. Hence, attackers use this trick to deceive the user that he is visiting the legitimate URL. For example, when user clicks the URL http://www.signin.ebay.com@www.iamphish.com/path-structure/index.php, he is redirected to iamphish domain but not legitimate ebay website.

(3) *R3: Use of HTTPS :* This feature checks the presence of HTTPS in the protocol of the visited URL. If HTTPS is present, then it is set to true (1), else set to false (0). Use of HTTPS claims the legitimacy behavior in the websites, whereas in phishing sites use of HTTPS is rare unless the phishing site is hosted on compromised HTTPS domain.

(4) *R4: Host length :* This feature calculates the number of characters in the hostname of the visited URL. It is observed that phishing URLs use lengthy URL such that actual brand or primary domain is hidden from the address bar.

(5) *R5: Age of the domain :* This feature calculates the age of the domain of the given URL using the information extracted from WHOIS database. Usually, the life spans of most phishing domains are less than 12 months as they are taken down at a faster rate [30].

(6) *R6: Presence of hyphens and underscores in hostname :* This feature checks the presence of hyphens (−) and underscores (_) in the hostname. If any of the symbols is present, the feature is set to true (1), else false (0). Some attackers try to misguide the online user by substituting one or more letters of trusted website URL with the above symbols in creating phishing URL. These small changes may not be noticed by users, leading to visiting of phishing site. For example, an attacker might design a phishing site URL as http://www.signin_ebay.com for the legitimate site http://www.signin_ebay.com.

(7) *R7: Base URL length:* This feature is similar to *R4* feature but it calculates the number of characters in the base URL of the visited URL. Base URL is calculated as protocol + hostname + pathname of given URL. It is observed that attackers insert target website URL in either the path or hostname of designed phishing URL such that online user would get deceived and directed to phishing URL. This behavior also attempts to hide the actual domain name of the URL from the address bar.

3.1.2 Proposed features

These features are further divided into similarity-based features and hyperlink-based features. The similarity-based features include P1 to P8, whereas hyperlink-based features include P9 and P10. The similarity-based features are extracted from the source code of both log-in page and home page of the given URL. Each similarity-based feature is fed to Jaccard similarity coefficient module to calculate the similarity between log-in and home page. The hyperlink-based features are extracted from log-in page of given URL.

(1) *P1: Filename similarity :* This feature extracts the filenames from the log-in page and home page of

the given URL. For the legitimate website, there exists at least some degree of reuse of filenames in both home page and log-in page. The filenames are extracted from various locations such as CSS, JavaScript, Images, favicons and logos. For identifying the resource filenames, we extract all the links from `img[src]`, `a[href]`, `link[href]`, `script[src]` and then save the details. The same procedure is followed for the home page too.

(2) *P2: Copyright Similarity :* This feature identifies similarity degree between the copyright text of respective log-in page and home page. The legitimate behavior is maintenance of same copyright text across all the pages in the entire website unless some page is hosted by the attacker on the compromised domain.

(3) *P3: Title similarity :* We extract the titles from both log-in and home pages anticipating that brand name exists in both the documents. But it is not mandatory to have brand information in the titles of all the pages in the website.

(4) *P4: Description similarity :* The websites use meta-information to describe the brand information and the action involved in that respective page. Hence, we try to find out whether there is any similarity between home and log-in page with descriptions of each as sets of words. These sets of words are given as input Jaccard similarity module to identify the similarity score.

(5) *P5: Maximum frequency-domain similarity :* Most of the websites have greater number of local domains than foreign domains. The local domains are the pages which are containing same domain as the hostname of the URL, and the reverse is called foreign domains. The maximum anchor links pointing to a domain are considered as maximum frequency domain. Usually, for the legitimate sites, the maximum frequency domain is set as local domain. Phishers have anchor links pointing to target legitimate sites, which results in maximum frequency domain set to foreign domain. This differentiates the behavior of legitimate sites with phishing sites. Hence, we extract maximum frequency domain for both log-in and home pages for calculating the similarity score.

(6) *P6: TF–IDF terms similarity :* Term frequency and inverse document frequency (TF–IDF) gives the most important words related to the document. Term frequency defines the number of times the word is repeated in a document. Inverse document frequency defines the number of documents that contain the above term. The frequency of brand names or identifiers of the websites is very high in the original website, and it is very less in other websites of corpus.

Earlier works [17, 35, 51, 54] used TF–IDF for identifying the brand names of the document with corpus of huge database of all documents. This consumes more time and storage while calculating the TF and IDF for each document. Hence, we came up with modified TF–IDF by skipping the database of all documents with either six or seven documents as corpus. We considered brand locations such as plaintext, title, description, copyright, URL terms, header text, maximum frequency domain (if not local domain) as each document for identifying the brand names of the given document as shown in Algorithm 1. The brand name of a website is usually present in title of the page, header texts with h1 and h2 tags, copyright section and URL (because it makes the user familiar with brand and URL map). The reason for choosing maximum frequency domain is that some phishing attacks include anchor links pointing to target legitimate site; in that case, we can get the target website brand name too. In the existing works, top TF–IDF words are considered due to the fact that TF value of brand name in a website results in high count, whereas IDF value of same brand term results in the low count (because of nonrelevance with other documents in the corpus). Combining both high TF value and low IDF values for the calculation of TF–IDF ratio results in high value. Hence, terms with top TF–IDF values are more relevant to the specified document. But, in our case, we considered the documents of corpus as brand locations of the given document, which results in very high IDF value for the brand name. Note that the brand name with high TF value and high IDF value results in low TF–IDF value. Hence, we considered the terms with low TF–IDF value as most relevant terms for the specified document. The same procedure is followed to identify the most relevant terms from the home page. The resulted two sets of words are checked for the similarity degree by using the Jaccard index.

Algorithm 1: FindLeastTFIDFWords

Input : A HTML document *doc*, URL of website *url*
Output: List of 5 least TF-IDF words (w_i)

```
Initialize a String array brandLocations[ ]
host=getHost(url)
title=getTitle(doc)
plaintext=getPlaintext(doc)
desc=getDescription(doc)
cr=getCopyright(doc)
h1h2=getHeadersText(doc)
maxfreqdomain=getMaxFreqDomain(doc)
if maxfreqdomain==getDomain(url) then
    Add(host, title, plaintext, desc,
      cr,h1h2,maxfreqdomain) to brandLocations [ ]
else
    Add (host, title, plaintext, desc, cr,h1h2) to
      brandLocations [ ]
end
words=getLeastTFIDFwords(brandLocations [ ])
return words
```

(7) *P7: NER brand similarity:* Sometimes, brand names of websites may exist in locations other than brand locations (title, copyright, header text, description, etc). Hence, we apply named entity recognition (NER) method to identify the brand names located anywhere in the website. NER is a field of natural language processing that uses sentence structure to identify proper nouns and classify them into a given set of categories. We used Stanford's coreNLP [1, 13, 24], Java library for implementing the named entity recognizer module. The module takes input as extracted plaintext from website and labels the sequence of words as person, organization or location. From the resulting output, we extract the words which are labeled as organization and are considered as brand names related to specified website. This NER module is run with both log-in page and home page with plaintext as input to extract set of brand names present in each page. The members of two sets are compared with Jaccard index to calculate the similarity score between both pages.

(8) *P8: Bond status* Usually, all the pages in a website contain the link back to the home page. If this behavior is present, then the bond status is set to true (1), else it is set to false (0). Phishing sites normally do not have anchor link pointing to home page. Note that this absence of bond behavior can also be present in legitimate sites too.

The above all similarity features are sufficient for identifying the phishing sites which are hosted on compromised domains but fail when the same page is used for both log-in page and home page. Attackers maintain a single log-in page or in addition log-in successor page for phishing campaign such that their effort is less. They might include all anchor links pointing to the same log-in page or single common page such that foreign domains are avoided. This is countered in the hyperlink-based features (P9–P10).

(9) *P9: Pattern-matching ratio :* This feature is adapted from our previous work [40, 46]. It identifies the ratio of frequency of common anchor links to the total number of anchor links. Many of the times phishing sites are designed with only single log-in page so they define all the hyperlinks of the log-in page direct to the same log-in page with # links such that online user is not diverted until sensitive information is given. Sometimes, we also observed attackers redirect all the hyperlinks to a common page other than current log-in page. Hence, phishing sites contain maximum of common page ratio, which includes either use of # links or single common page. On the other side, the legitimate sites usually maintain different pages for different anchor links which include less pattern-matching ratio.

(10) *P10: Zero links presence :* This feature checks for the presence of hyperlinks in the body of the HTML of suspicious site. If present, it is set to 1, else set to 0. The rationale behind this feature is that attackers design a phishing site by replacing the text with screenshot of target website such that text-based techniques [17, 51, 54] would fail to detect as shown in Fig. 3. These kinds of phishing sites are termed as image-based phishing sites. In the figure, it is shown that image A is inserted as background image in the body of the HTML as shown in C. On top of the image A, input fields and log-in buttons (i.e., B) are embedded to result in final desired phishing page C.

4 Machine learning classifiers used

The features which are extracted from URL, similarity and hyperlinks modules are integrated to generate a feature vector required for training the model. The URL-based features generate seven-dimensional vector, whereas proposed features contribute to ten-dimensional vector. These features are combined and fed to machine learning algorithm for the classification of phishing websites using ten cross-validation. We have also applied the exhaustive search method for selecting the parameters for each algorithm.

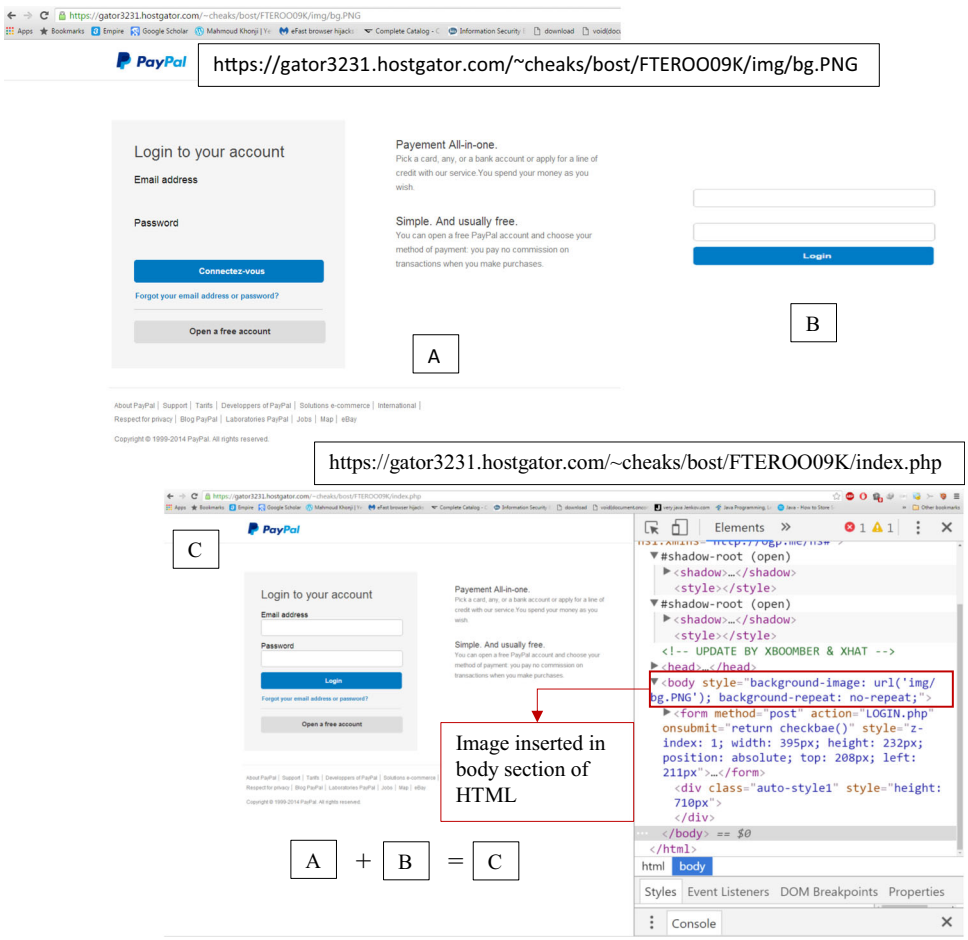


Fig. 3 Phishing site with design of text replacement with screenshot

This section provides a brief overview of the machine learning models used in the paper. We have used support vector machine (SVM) [7, 10], proximal support vector machine (PSVM) [14] and twin support vector machine [21, 22] for the classification of the phishing websites.

4.1 Support vector machine

Support vector machine (SVM) is one of the trending kernel-based machine learning algorithms for classification and regression. SVM has emerged from the research in statistical learning theory [47] on how to regulate the trade-off between the structural complexity and empirical risk.

Given a training set $T = \{(x_i, y_i) : x_i \in \mathbb{R}^n$ and $y \in \{-1, 1\},\ i = 1, 2, ..., l\}$, SVM finds the optimal separating hyperplanes $w^T\phi(x) + b = 0$ in feature space where $\phi : \mathbb{R}^n \to \mathcal{H}$ maps data points in the higher dimensional feature space $\mathcal{H}$. These separating hyperplanes have been obtained by solving following quadratic prime problem (QPP):

$$\begin{aligned} &\min_{(w,b,\xi)} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i \\ &\quad \text{subject to,} \\ &y_i(w^T\phi(x_i) + b) \geq 1 - \xi_i \ i = 1, 2, .., l, \\ &\xi_i \geq 0,\ i = 1, 2, .., l. \end{aligned} \tag{2}$$

where $C \geq 0$ controls the trade-off between the margin width $\frac{1}{2}w^T w$ and the minimization of the training error. Using the Karush–Kuhn–Tucker (KKT) conditions, the dual problem of the QPP (2) has been obtained as follows

$$\begin{aligned} &\min_{\alpha} \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \phi(x_i)^T \phi(x_j) \alpha_i \alpha_j - \sum_{i=1}^{l} \alpha_i \\ &\quad \text{subject to,} \\ &\sum_{i=1}^{l} \alpha_i y_i = 0, \\ &0 \leq \alpha_i \leq C,\ i = 1, 2, ..., l. \end{aligned} \tag{3}$$

Using the Mercer condition [28, 47], it is possible to use a kernel function $K(x_i, x_j)$ to represent the inner product $\phi(x_i)^T\phi(x_j)$ in the feature space $\mathcal{H}$.

After obtaining the optimal values of w and b from (3), a test point $x \in \mathbb{R}^n$ has been classified using the decision function

$$f(x) = sign(w^T x + b). \tag{4}$$

4.2 Proximal support vector machine

Proximal support vector machine (PSVM) [14] solves the following optimization problem for finding the separating hyperplane $w^T\phi(x) + b = 0$

$$\begin{aligned} &\min_{(w,b,\xi)} \frac{1}{2}(w^T w + b^2) + \frac{C}{2}\sum_{i=1}^{l} \xi_i^2 \\ &\quad \text{subject to,} \\ &y_i(w^T\phi(x_i) + b) = 1 - \xi_i,\ i = 1, 2, .., l. \end{aligned} \tag{5}$$

As compared to SVM, the PSVM uses the equality constraints instead of inequality constraints for finding the decision function (4). It minimizes the square of the error variable ξ_i in its objective function which enables its optimization problem (5) to be solved by only solving a system of equations.

4.3 Twin support vector machine

Twin support vector machine (TWSVM) [21, 22] finds two nonparallel hyperplanes

$$w_1^T x + b_1 = 0 \quad \text{and} \quad w_2^T x + b_2 = 0 \tag{6}$$

which are proximal to their corresponding classes and have at least unit distance far from the points belonging to the opposite classes. A given test point $x \in R^n$ is classified into class depending upon which of the two hyperplanes it lies closer to.

For the nonlinear classifier, TWSVM considers the following kernel-generated surfaces

$$K(x^T, C^T)u_1 + b_1 = 0 \ \text{ and } \ K(x^T, C^T)u_2 + b_2 = 0 \tag{7}$$

where $C^T = [A\ B]^T$. Here, A and B are $m_1 \times n$ and $m_2 \times n$ data matrices containing the data points belonging from the class 1 and class -1, respectively. It should be noted that the hyperplanes (6) can be obtained as the special case of (7) by using the linear kernel $K(x^T, C^T) = x^T A$ and thereafter defining the $w_1 = A^T u_1$ and $w_2 = A^T u_2$.

For finding hyperplanes (7), the TWSVM [21, 22, 44] solves the following pair of optimization problems, i.e., kernel TWSVM-1 (KTWSVM-1) and kernel TWSVM-2 (KTWSVM-2)

(KTWSVM-1)

$$\begin{aligned} &\min_{(u_1,b_1,\xi_1)} \frac{c_3}{2}(||u_1||^2 + b_1^2) + \frac{1}{2}||(K(A, C^T)u_1 + e_1 b_1)||^2 \\ &\quad + c_1 e_2^T \xi_1 \\ &\quad \text{subject to,} \\ &\quad -(K(B, C^T)u_1 + e_2 b_1) + \xi_1 \geq e_2,\ \ \xi_1 \geq 0, \end{aligned} \tag{8}$$

(KTWSVM-2)

$$\begin{aligned} &\min_{(u_2,b_2,\xi_2)} \frac{c_4}{2}(||u_2||^2 + b_2^2) + \frac{1}{2}||(K(B, C^T)u_2 + e_2 b_2)||_2 \\ &\quad + c_2 e_1^T \xi_2 \\ &\quad \text{subject to,} \\ &(K(A, C^T)u_2 + e_1 b_2) + \xi_2 \geq e_1,\ \ \xi_2 \geq 0. \end{aligned} \tag{9}$$

As compared to SVM, which solves a single QPP which has $m_1 + m_2$ constraints, TWSVM solves two small QPPs with m_1 and m_2 constraints. It makes the TWSVM much faster than SVM.

The KKT optimality conditions for problem (8) are given by

$$K(A, C^T)^T(K(A, C^T)u_1 + e_1 b_1) + K(B, C^T)^T \alpha = 0, \tag{10}$$

$$e_1^T(K(A, C^T)u_1 + e_1 b_1) + e_2^T \alpha = 0, \tag{11}$$

$$c_1 e_2 - \alpha - \beta = 0, \tag{12}$$

$$-(K(B^T, C^T)u_1 + e_2 b_1) + \xi_1 \geq e_2,\ \ \xi_1 \geq 0, \tag{13}$$

$$\alpha^T(-(K(B, C^T)u_1 + e_2 b_1) + \xi_1 - e_2) = 0, \beta^T \xi_1 = 0, \tag{14}$$

$$\alpha \geq 0, \beta \geq 0. \tag{15}$$

Combining (10) and (11), we get

$$\begin{aligned} &[K(A, C^T)^T\ e_1^T][K(A, C^T)\ e_1][u_1, b_1]^T \\ &\quad + [K(B, C^T)^T\ e_2^T]\alpha = 0 \end{aligned} \tag{16}$$

Let $H = [K(A, C^T)\ e_1]$, $G = [K(B, C^T)\ e_2]$ and the augmented vector $z = [u_1, b_1]^T$, then (16) can be written as

$$H^T H z + G^T \alpha = 0 \quad \text{i.e.,} \ \ z = -(H^T H)^{-1} G^T \alpha \tag{17}$$

Using the KKT optimality condition, Wolfe dual of the (KKTWSVM-1) is given by

(KDTWSVM-1)

$$\begin{aligned} &\max_{\alpha} \quad e_2^T \alpha - \frac{1}{2}\alpha^T G(H^T H)^{-1} G^T \alpha \\ &\quad \text{subject to,} \\ &0 \leq \alpha \leq c_1. \end{aligned} \tag{18}$$

In the similar way, the Wolfe dual of the (KKTWSVM-2) is obtained as

(KDTWSVM-2)

$$\max_{\gamma} \quad e_1^T\gamma - \frac{1}{2}\gamma^T H(G^TG)^{-1}H^T\gamma \quad \text{subject to,} \quad 0 \le \gamma \le c_2. \tag{19}$$

After solving the (KDTWSVM-1) and (KDTWSVM-2), the hyperplanes (7) have been obtained. A new data sample $x \in R^n$ is assigned to class $r = (r = 1, 2)$, depending on which of the two hyperplanes it lies closer to, i.e.,

$$class\ r = arg\ \min_{l=1,2} |K(x^T, C^T)u_l + b_l|$$

5 Experimentation results

5.1 Experimental setup and tools used

Our method takes as input suspicious URL and extracts the source code from both suspicious URL and home page of the URL using Selenium WebDriver[2]. Selenium is a web browser automation tool which opens Firefox browser externally to perform browser like action as an automated process. Note that home page of any website is considered as hostname of given input URL. Sometimes, visiting of hostname leads to not-found pages; in that case, domain of the input URL is considered as home page. All the features from the source code are extracted using the Jsoup library. Jsoup[3] is a Java library which is used to parse and manipulate the HTML code of a given website. The reason behind the choosing Jsoup library is its availability of API which provides functionality to select elements of document object model (DOM) and also that it handles nonwell-formed HTML very effectively. Once the features and similarity scores between page S and page H are calculated, then the scores are used to construct feature vectors. These vectors are given as input to the machine learning classifiers for classifying the suspicious sites as phishing or legitimate.

All of the machine learning methods SVM, PSVM, and TWSVM have been simulated in MATLAB 12.0 environment on Dell Precision T1700 CPU with 16 GB of RAM. The MATLAB codes of SVM and PSVM are downloaded from URLs http://www.svms.org/software.html, http://svms.org/tutorials/Gunn1998.pdf and http://research.cs.wisc.edu/dmi/svm/psvm, respectively. We have used the RBF kernel in the SVM, PSVM and TWSVM for the prediction of the phishing websites.

[2] http://docs.seleniumhq.org/download/.

[3] https://jsoup.org/.

Table 3 Tuning parameters for the experiments

Learning algorithms	Parameters
TWSVM	P = 0.5 and C = 0.4
PSVM	P = 0.4 and v = self-tuned
SVM	P = 2 and C = 2

5.2 Parameter tuning

We have tuned the parameters of all machine learning methods using exhaustive search method [36]. The value of the RBF kernel parameter P and C of SVM and PSVM has been searched in the set $\{2^i : i = -10, -9, ..., 9, 10\}$. For the TWSVM, we have taken $c_1 = c_2$ for the sake of convenience of experiments and their optimal values have been searched in the set $\{0.1, 0.2, ..., 1\}$.

The final selected parameters are given in Table 3.

5.3 Dataset

We collected dataset from two sources as given in Table 4. The phishing sites are collected from PhishTank website, and legitimate sites are collected from list of top websites maintained by Alexa. Note that we have collected the 5500 legitimate websites randomly from the Alexa database such that bias is prevented toward high-ranked websites unlike existing works [12, 18, 40, 50, 51]. One of the limitations in existing works is that they considered top-ranked legitimate sites and page rank as one of the attributes for the classification of legitimate and phishing. This results in bias toward high-ranked benign sites, which obviously increases the accuracy. But this is not a fair dataset for identifying the performance of the proposed models. Hence, we have randomly chosen legitimate sites from the list of top 1 lakh websites from Alexa.

5.4 Evaluation

In this section, we evaluated our proposed model by conducting various experiments with different feature sets and machine learning algorithms. We have included SVM and other versions of SVMs [21, 22] for the classification of suspicious sites.

We divided our features into four sets, namely

- *FS1*—list of basic URL-based features (R1–R7).
- *FS2*—list of proposed features including both similarity and hyperlink-based features (P1–P10).
- *FS2'*—list of proposed features including only similarity-based features (P1–P8).

Table 4 Source of websites

Source	Total instances	Link
Phishing sites	5500	https://www.phishtank.com/developerinfo.php
Legitimate sites	5500	http://www.alexa.com/topsites

– *FS3*—list of all features combining both FS1 and FS2 (R1–R7, P1–P10).

We conducted various experiments with different feature sets and different machine learning algorithms for assessing the importance of the proposed features. All the experiments are conducted with the same dataset of 11000 instances (5500 legitimate, 5500 phishing), and tenfold cross-validation is used for each. In Experiment 1, we have evaluated our proposed model with all the features (FS3) using various classifiers. This experiment also results in identifying the best classifier suitable for our dataset. In Experiment 2, we evaluated different feature sets FS1, FS2, FS3 for identifying the importance of each set in detecting the phishing sites.

In Experiment 3, we evaluated similarity-based features FS2' for identifying the effectiveness of detecting phishing sites hosted on compromised servers (POCS). This experiment reveals the richness of FS2' in detecting POCS. Note that we investigated the dataset of all phishing sites (5500) manually to identify the POCS. We observed 3970 POCS out of 5500, which contributes to 72% of total phishing sites. Finally, In Experiment 4, we evaluated our model with each individual feature to identify the importance of each in detecting phishing websites.

We used eight traditional metrics to identify the performance of our system. Note that we considered condition positive as phishing (P) and negative as legitimate (L). Correctly predicted phishing sites are termed as hit or true positive (TP), correctly predicted legitimate sites as true negative (TN), incorrectly predicted legitimate sites as false positive (FP), incorrectly predicted phishing sites as false negative (FN).

– *Recall* : It is defined as a percentage of correctly predicted phishing sites (TP) out of a total number of phishing sites. It is also termed as sensitivity or true positive rate (TPR)

$$Recall = \frac{TP}{TP + FN} * 100 \tag{20}$$

– *Specificity*: It is defined as a percentage of correctly predicted legitimate sites (TN) out of a total number of legitimate sites. It is also termed as true negative rate (TNR).

$$Specificity = \frac{TN}{TN + FP} * 100 \tag{21}$$

– *False positive rate (FPR)*: It is defined as a percentage of incorrectly predicted legitimate sites (FP) out of a total number of legitimate sites.

$$FPR = \frac{FP}{FP + TN} * 100 = 1 - Specificity \tag{22}$$

– *False negative rate (FNR)*: It is defined as a percentage of incorrectly predicted phishing sites (FN) out of a total number of phishing sites.

$$FNR = \frac{FN}{FN + TP} * 100 = 1 - Recall \tag{23}$$

– *Accuracy (Acc)*: It is defined as a percentage of correctly predicted legitimate and phishing sites out of a total number of websites.

$$Acc = \frac{TP + TN}{TP + FP + TN + FN} * 100 = \frac{TP + TN}{P + L} * 100 \tag{24}$$

– *Error rate (ERR):* It is defined as a percentage of phishing and legitimate sites that are incorrectly classified out of a total number of websites.

$$ERR = \frac{FP + FN}{P + L} * 100 \tag{25}$$

– *Precision (Pre):* It is defined as a percentage of correctly predicted phishing sites out of a total number of predicted phishing sites.

$$Pre = \frac{TP}{TP + FP} * 100 \tag{26}$$

– *F-measure:* It is calculated as a harmonic mean of precision and recall. The F-measure will always be nearer to the smaller value of precision or recall. It is a combined measure that assesses the precision and recall trade-off:

$$F = 2 * \frac{Pre * Recall}{Pre + Recall} \tag{27}$$

5.4.1 Experiment 1: evaluation of FS3 with various classifiers

In this first experiment, we evaluated our feature set FS3 with SVM and observed an accuracy of 98.05% with 98.33% recall and 97.77% specificity. We also tested our features with other versions of SVM such as TWSVM and PSVM to identify the best classifier for the detection of

Table 5 Results of Experiment 1

Metrics	TWSVM	SVM	PSVM
Recall	98.33	97.86	95.49
Specificity	97.77	97.61	94.57
Precision	97.74	97.60	94.67
F1 score	98.03	97.73	95.08
Accuracy	98.05	97.73	95.03

phishing sites in our dataset. From the results shown in Table 5, it is observed that all the applied machine learning algorithms achieved an accuracy of more than 95%, which indicates the richness of our proposed features in classifying the phishing websites.

5.4.2 Experiment 2: evaluation of FS1, FS2, FS3 with twin support vector machine

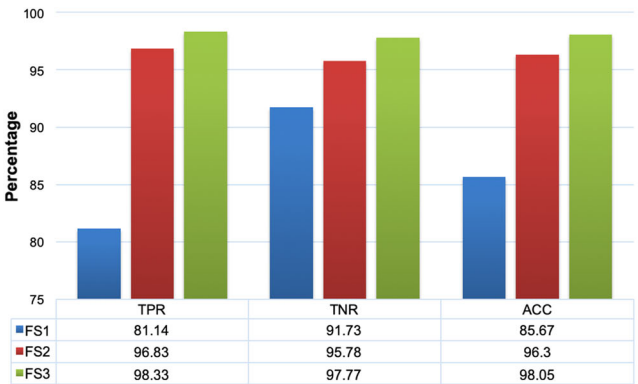
In this experiment, we evaluate the basic URL features FS1, proposed features FS2 and combined features FS3 by calculating the detection rate and error rate with above-mentioned metrics as shown in Fig. 4. Note that the experiments with FS1, FS2 and FS3 are carried out with TWSVM classifier as it performed the best in detecting the phishing sites as discussed in Experiment 1. Firstly, we tested our model with FS1; it resulted in TNR value of 91.73% and TPR value of 81.14%, which shows the richness of selected existing features but the accuracy of 85.57% indicates that FS1 alone is not sufficient to counter the phishing sites. Later, we tested our model with feature set FS2, to evaluate the performance of proposed features. The results of the experiment demonstrate an increase in TPR to 96.83%, TNR to 95.78% and an accuracy to 96.3%. The significant metric values with FS2 dataset indicates the goodness of our proposed features. Finally, we tested our model with FS3, which resulted in increase in TPR, TNR and accuracy with 98.33%, 97.77% and 98.05%, respectively. The significant change in metric values indicates that addition of FS1 to FS2 complements the accuracy in detecting phishing sites. All the experimental results with feature sets can be seen in Fig. 4 where 4 a) gives detection accuracy and 4 b) gives error detection of the system.

Table 6 Evaluation of similarity-based features

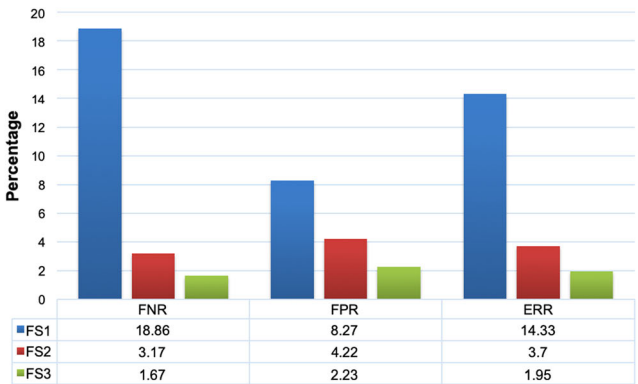
Classification output	Value
# of legitimate sites	5500
# of POCS	3970
Recall	99.52%
False positive rate	6.36%
Precision	91.90%
F-measure	95.56%
Accuracy	96.11%

5.4.3 Experiment 3: evaluation of FS2′ with twin support vector machine

In this experiment, we evaluate only similarity-based features which are used for detecting phishing sites hosted on compromised servers. For testing the performance of our model in detecting phishing sites hosted on compromised servers, we considered a dataset of 5500 legitimate sites and 3970 phishing sites which are hosted on compromised servers. Note that we identified 3970 out of 5500 phishing sites really hosted on compromised servers. The experiment results in Table 6 reveal that percentage of correctly detected phishing sites reached 99.52%, which shows the importance of similarity-based features in detecting the POCS. Our model could achieve an accuracy of 96.11% , precision of 91.90% and F-measure of 95.56%, which illustrate that addition of these features to other category of features complements the information and thereby improves the accuracy of the overall system.



(a) Detection Accuracy



(b) Detection Error

Fig. 4 Performance of the proposed system with different feature sets

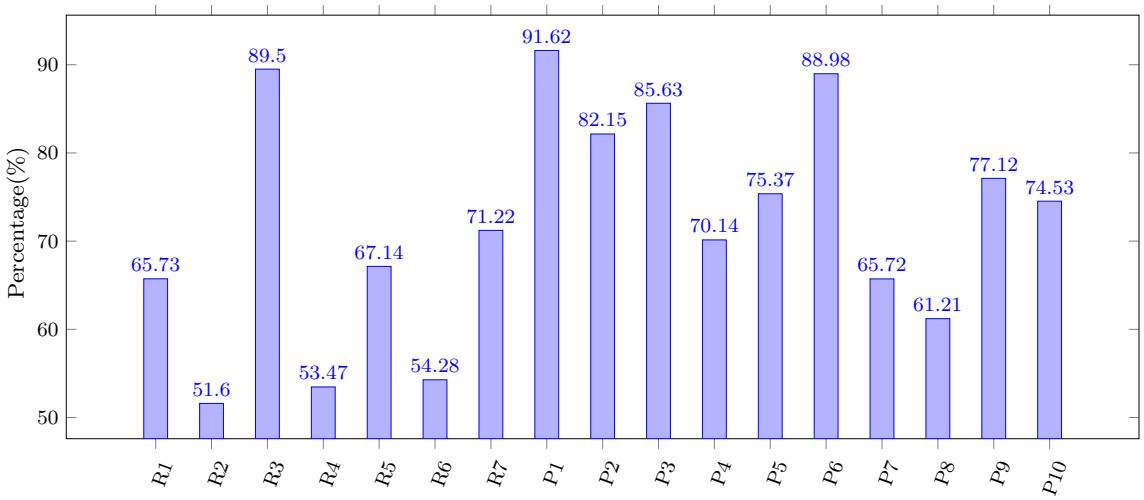


Fig. 5 Classification accuracy of individual features

5.4.4 Experiment 4: performance of system with individual feature

The compromised behavior-based features are very hard to be evaded by attackers because they cannot manipulate the content of home page of compromised domain; if so, the website owner will get a suspicion on it and might get to know that domain is compromised. In the other way, if an attacker tries to modify the content of his phishing page such that similarity between home and log-in page would be increased, then the online user who is exposed to phishing site might get a suspicion on the differences in fake site and may leave the phishing page. In this section, we also discuss the importance of individual feature in detecting phishing sites. The accuracies of the proposed model evaluated with each feature are given in Fig. 5. The feature P1 (filename similarity) has achieved an accuracy of 91.62%, which indicates that the reuse of external resources like images, scripts, styles is most common in legitimate sites and less appealing in phishing sites in compromised domains. The remaining percentage of error is due to the websites which are registered maliciously or due to the poor designing of legitimate sites.

The features like P2 (Copyright), P3 (Title) and P6 (TF–IDF) have crossed an accuracy of above 80%. If we observe these three features, all of them are text-based features which consider textual content of the website for extracting unique descriptors of the website. Note that feature P4 (description) is also textual-based feature, which has achieved an acceptable accuracy of 70.14%. The significant accuracy of these features concludes that textual-based features play a very vital role in detecting phishing sites in POCS. From the figure, it is observed that most of the proposed features outperformed basic URL features with over 70% accuracy including P1–P6 and P9–P10. Out of all URL features, only R3 (HTTPS) could achieve an accuracy of above 85%, which indicates the low usage of HTTPS phishing sites and also low number of POCS with HTTPS protocol. Hyperlink-based features P9–P10 also achieved an accuracy of more than 70%, which indicates the strength of features in detecting the malicious registered phishing sites. It should be noted that the statistics are dependent on the quality and quantity of the training data used for the classification of phishing websites.

5.4.5 Experiment 5: comparison of our work with existing works

In this experiment, we implemented existing benchmarked anti-phishing approaches such as Shirazi et al. [45], Yang et al. [52] and Li et al. [23] for the comparison with our work based on the performance metrics of accuracy, sensitivity and specificity. We used the same dataset used in Experiment 1 for the comparison of existing works. The main reason for choosing these techniques over other works is the similarity in extracted features from URL and source code. To ease the presentation, we term the Shirazi et al.'s work as W1, Yang et al.'s work as W2, Li et al.'s work as W3 and our proposed work as W4. The comparison results are given in Table 7. The results demonstrate

Table 7 Comparison of our work with existing works

Metrics (%)	W1	W2	W3	W4
Recall	95.98	96.77	96.57	98.33
FPR	5.14	2.34	2.40	2.23
Accuracy	95.43	97.21	97.08	98.05
Precision	95.02	97.68	97.62	97.74
F-measure	95.50	97.23	97.09	98.03

that our work outperforms the existing works W1, W2 and W3 with an accuracy of 98.05% and a precision of 97.74%.

6 Discussion and limitations

6.1 Design of chrome extension

Our goal in this work is to provide real-time protection from website phishing attacks. Hence, we built a chrome extension which predicts the visited URL as a phishing or legitimate. Single click on the extension reveals the status of the website. The chrome extension is written in JavaScript, which extracts URL, title and DOM of visited website from the browser and makes a connection to REST API running at the remote server where the actual execution of technique takes place. The REST API is hosted on an Intel Xeon 16 core Ubuntu server with 2.67GHz processor and 16GB RAM.

The REST API feeds the above values to our application running at the remote server and thus proceeds with extraction of features (Sect. 3.1) for the phishing detection. Note that the REST API is implemented with Spring framework and POST method is used for transferring the DOM, whereas GET method is used for transferring the URL and Title to the server. Once the server receives these values, it classifies the status of website as phishing or legitimate based on the outputs of modules discussed in Sect. 3.1.

Since our goal is to provide real-time phishing detection, the time for feature extraction and classification has to be very less. Hence, we parallelized the process of feature extraction for the proposed features with the help of multiple cores present in the system. Once the classification is done, the application returns the status to REST API which is further sent to Extension as a response.

This whole process of execution took an average time of 3.19 seconds. However, this time is dependent on various factors such as website hosted on the server, speed of the Internet and speed of the system. In our case, we used a bandwidth of 100mbps, as mentioned earlier, system with Intel Xeon 16 core Ubuntu server with 2.67GHz processor and 16GB RAM. The output of the extension is shown as popup window containing the status of the website as shown in Fig. 6. In image (a) of Fig. 6, our extension detects the Apple website as legitimate with a response time of 2.64 sec, whereas image (b) shows the detection of phishing site with a response time of 2.77 sec in a popup window containing the status of website and textual content recommending the user to close the webpage immediately.

6.2 Effectiveness of the proposed model

Unlike list-based or visual similarity-based approaches, our technique is independent of the earlier storage of images, DOMS, Styles and URLs. Hence, when we extract home page from the visited URL, it sometimes results in false positives.

In our experiment 1, we observed a false positive rate of 2.23% due to the nonavailability of home page for the visited URL. This can, however, be reduced by identifying the home page with search results (SR) queried with search query=primary domain. The process of identifying the home page with SR includes additional computation cost but increases the accuracy. We intend to attempt this procedure of identifying the home page in the future work.

Similarly, our model also included FNR of 1.67% in Experiment 1. This is due to the fact that phishing sites which are maliciously registered have their own home page and login page or both represent the same page. This made our technique fail to detect such phishing pages. However, if we observe the Experiment 3 results, our model could detect 99.52% of phishing POCS. Hence, our model is more suitable for the detection of POCS.

If the attacker copies the contents of compromised site (code obfuscation) to his/her designed phishing page, then the technique may be bypassed. But such a scenario is very rare because of the following cases.

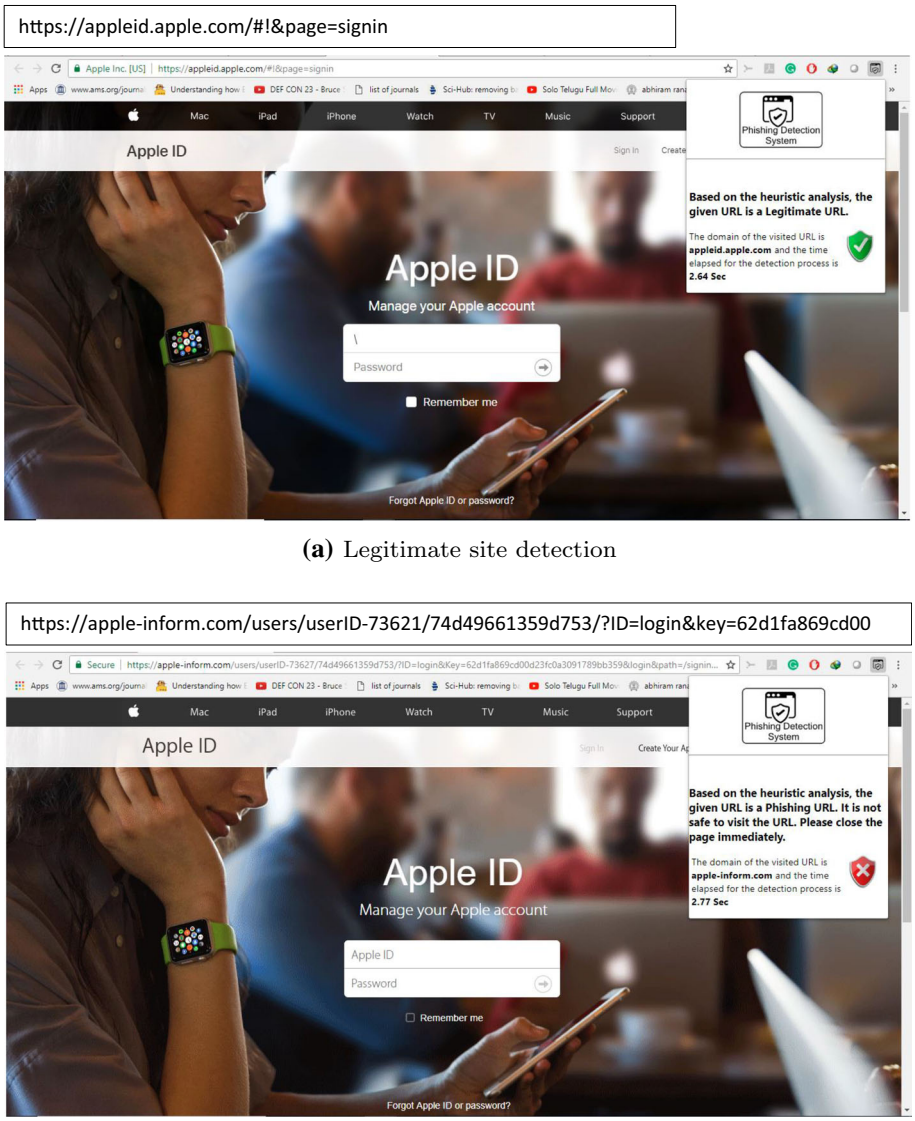
- Attacker has to make an extra effort in making changes to the downloaded phishing kit.
- If he adds the content of compromised site into designed phishing site, then the phishing site might show some suspicious behavior to the online user.
- From the experimental analysis, we did not observe any single phishing instance copying the contents of compromised server to the designed phishing page.

The main focus of the work is to detect phishing sites which are hosted on compromised servers as they contribute to more than 70% of phishing websites [32]. Hence, our proposed work is designed to detect such phishing sites along with malicious registered phishing sites without the dependency on an initial database of resources (CSS, Images, DOM, URLs) or third-party services. With the use of TWSVM classifier, our proposed model is able to achieve a significant accuracy of 98.05%.

6.3 Limitations

We observed some limitations in our study. Firstly, the legitimate sites which are hosted on free websites are also classified as phishing due to the high dissimilarity score between the free hosting domain and legitimate site.

Fig. 6 Output of our extension



(a) Legitimate site detection

(b) Phishing site detection

However, free hosting servers are never used by the genuine corporates for their business transactions.

Secondly, our model fails to detect low-content webpages such as single sign-on. It may be noted that attackers design the phishing pages, which mimic the target website to the maximum degree. We rely mostly on webpage DOM for the extraction of features. Hence, if an attacker attempts to steal sensitive information with low targeted content, our model may not correctly classify the website. But, the designed phishing page with low target content has low success rate as it might create suspicious behavior to the online user.

7 Conclusion and future work

In this paper, we proposed a technique which detects phishing sites hosted on compromised servers based on the dissimilarity between the visiting page and the home page of the given URL. This is achieved through similarity-based features. Additionally, the remaining malicious registered phishing sites are also countered with basic URL and hyperlink-based features. A nonparallel plane classifier, twin support vector machine, is used for the classification of phishing and legitimate sites.

Our technique detects phishing sites without relying on preapproved or unapproved database, prior knowledge of the user such as web history, third-party services such as

search engine, page ranking, WHOIS, etc. This makes our technique adaptable at client side.

With the help of aforementioned heuristic features, we achieved a TPR of 98.33%, TNR of 97.77% and an accuracy of 98.05%. We also achieved TPR of 99.52% in an experiment on POCS, which shows the richness of similarity-based features in detecting phishing sites on compromised sites. But, the overall accuracy of an experiment on POCS resulted in 96.11% due to the FPR of 6.36%. In the future, we intend to include lightweight search engine-based features to reduce the FPR and thereby improve the accuracy of the model.

Acknowledgements The authors would like to thank Ministry of Electronics and Information Technology (MeitY), Government of India, for their support in part of the research.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval This article does not contain any studies with human participants or animals performed by any of the authors.

References

1. (2005) Stanford CoreNLP-Natural language software. https://stanfordnlp.github.io/CoreNLP/#download
2. Afroz S, Greenstadt R (2011) Phishzoo: Detecting phishing websites by looking at them. In: Semantic Computing (ICSC), 2011 Fifth IEEE International Conference on, IEEE, pp 368–375
3. APWG (2016) Phishing attack trends reports, fourth quarter 2016. http://docs.apwg.org/reports/apwg_trends_report_q4_2016.pdf, Accessed: 2017-03-03
4. APWG (2017) Phishing attack trends reports, first half 2017. http://docs.apwg.org/reports/apwg_trends_report_h1_2017.pdf, Accessed: 2018-01-01
5. Ardi C, Heidemann J (2016) Auntietuna: Personalized content-based phishing detection. In: NDSS Usable Security Workshop (USEC)
6. Britt J, Wardman B, Sprague A, Warner G (2012) Clustering potential phishing websites using deepmd5. In: LEET
7. Burges CJ (1998) A tutorial on support vector machines for pattern recognition. Data Min Knowl Discov 2(2):121–167
8. Chen KT, Chen JY, Huang CR, Chen CS (2009) Fighting phishing with discriminative keypoint features. IEEE Internet Comput 13(3)
9. Chiew KL, Chang EH, Tiong WK et al (2015) Utilisation of website logo for phishing detection. Comput Secur 54:16–26. https://doi.org/10.1016/j.cose.2015.07.006
10. Cortes C, Vapnik V (1995) Support-vector networks. Mach Learn 20(3):273–297
11. Drew J, Moore T (2014) Automatic identification of replicated criminal websites using combined clustering. Security and privacy workshops (SPW). IEEE, IEEE, pp 116–123
12. Dunlop M, Groat S, Shelly D (2010) Goldphish: Using images for content-based phishing analysis. In: Internet Monitoring and Protection (ICIMP), 2010 Fifth International Conference on, IEEE, pp 123–128
13. Finkel JR, Grenager T, Manning C (2005) Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd annual meeting on association for computational linguistics, Association for Computational Linguistics, pp 363–370
14. Fung GM, Mangasarian OL (2005) Multicategory proximal support vector machine classifiers. Mach Learn 59(1–2):77–97
15. Gowtham R, Krishnamurthi I (2014) A comprehensive and efficacious architecture for detecting phishing webpages. Comput Secur 40:23–37. https://doi.org/10.1016/j.cose.2013.10.004
16. Hara M, Yamada A, Miyake Y (2009) Visual similarity-based phishing detection without victim site information. In: Computational Intelligence in Cyber Security, 2009. CICS'09. IEEE Symposium on, IEEE, pp 30–36, https://doi.org/10.1109/CICYBS.2009.4925087
17. He M, Horng SJ, Fan P, Khan MK, Run RS, Lai JL, Chen RJ, Sutanto A (2011) An efficient phishing webpage detector. Expert Syst Appl 38(10):12018–12027. https://doi.org/10.1016/j.eswa.2011.01.046
18. Huh JH, Kim H (2011) Phishing detection with popular search engines: Simple and effective. In: International Symposium on Foundations and Practice of Security, Springer, pp 194–207. https://doi.org/10.1007/978-3-642-27901-0_15
19. Jain AK, Gupta BB (2017) Two-level authentication approach to protect from phishing attacks in real time. J Ambient Intell Hum Comput. https://doi.org/10.1007/s12652-017-0616-z
20. Jang-Jaccard J, Nepal S (2014) A survey of emerging threats in cybersecurity. J Comput Syst Sci 80(5):973–993
21. Jayadeva KR, Chandra S (2007) Twin support vector machines for pattern classification. IEEE Trans Pattern Anal Mach Intell 29(5):905–910. https://doi.org/10.1109/TPAMI.2007.1068
22. Jayadeva KR, Chandra S (2017) Twin support vector machines. Springer, Berlin
23. Li Y, Yang Z, Chen X, Yuan H, Liu W (2019) A stacking model using url and html features for phishing webpage detection. Fut Gen Comput Syst 94:27–39. https://doi.org/10.1016/j.future.2018.11.004
24. Manning CD, Surdeanu M, Bauer J, Finkel J, Bethard SJ, McClosky D (2014) The Stanford CoreNLP natural language processing toolkit. In: Association for Computational Linguistics (ACL) System Demonstrations, pp 55–60, http://www.aclweb.org/anthology/P/P14/P14-5010
25. Mao J, Tian W, Li P, Wei T, Liang Z (2017) Phishing-alarm: robust and efficient phishing detection via page component similarity. IEEE Access 5:17020–17030
26. Marchal S, Saari K, Singh N, Asokan N (2016) Know your phish: novel techniques for detecting phishing sites and their targets. In: Distributed Computing Systems (ICDCS), 2016 IEEE 36th International Conference on, IEEE, pp 323–333
27. Medvet E, Kirda E, Kruegel C (2008) Visual-similarity-based phishing detection. In: Proceedings of the 4th international conference on Security and privacy in communication netowrks, ACM, p 22
28. Mercer J (1909) Functions of positive and negative type, and their connection with the theory of integral equations. Philos Trans R Soc Lond Ser A Contain Pap Math Phys Char 209:415–446
29. Moghimi M, Varjani AY (2016) New rule-based phishing detection method. Expert Syst Appl 53:231–242. https://doi.org/10.1016/j.eswa.2016.01.028
30. Mohammad RM, Thabtah F, McCluskey L (2012) An assessment of features related to phishing websites using an automated technique. In: Internet Technology And Secured Transactions, 2012 International Conference for, IEEE, pp 492–497
31. Mohammad RM, Thabtah F, McCluskey L (2015) Tutorial and critical analysis of phishing websites methods. Comput Sci Rev 17:1–24

32. Moore T, Clayton R (2007) Examining the impact of website take-down on phishing. In: Proceedings of the anti-phishing working groups 2nd annual eCrime researchers summit, ACM, pp 1–13
33. Pan Y, Ding X (2006) Anomaly based web phishing page detection. Proc Annu Comput Secur Appl Conf ACSAC 6:381–392. https://doi.org/10.1109/ACSAC.2006.13
34. Prakash P, Kumar M, Kompella RR, Gupta M (2010) Phishnet: predictive blacklisting to detect phishing attacks. In: INFOCOM, 2010 Proceedings IEEE, IEEE, pp 1–5, https://doi.org/10.1109/INFCOM.2010.5462216
35. Ramesh G, Krishnamurthi I, Kumar KSS (2014) An efficacious method for detecting phishing webpages through target domain identification. Decis Support Syst 61:12–22. https://doi.org/10.1016/j.dss.2014.01.002
36. Rao CR, Mitra SK (1971) Generalized inverse of matrices and its applications
37. Rao RS, Ali ST (2015) A computer vision technique to detect phishing attacks. In: Communication Systems and Network Technologies (CSNT), 2015 Fifth International Conference on, IEEE, pp 596–601, https://doi.org/10.1109/CSNT.2015.68
38. Rao RS, Ali ST (2015) Phishshield: a desktop application to detect phishing webpages through heuristic approach. Proc Comput Sci 54:147–156. https://doi.org/10.1016/j.procs.2015.06.017
39. Rao RS, Pais AR (2017) An enhanced blacklist method to detect phishing websites. In: International Conference on Information Systems Security, Springer, pp 323–333
40. Rao RS, Pais AR (2018) Detection of phishing websites using an efficient feature-based machine learning framework. Neural Comput Appl 1:1. https://doi.org/10.1007/s00521-017-3305-0
41. Rosiello AP, Kirda E, Ferrandi F, et al (2007) A layout-similarity-based approach for detecting phishing pages. In: Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on, IEEE, pp 454–463
42. Rosiello AP, Kirda E, Ferrandi F, et al (2007) A layout-similarity-based approach for detecting phishing pages. In: Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on, IEEE, pp 454–463
43. RSA (2013) Rsa fraud report. https://www.emc.com/collateral/fraud-report/rsa-online-fraud-report-012014.pdf, Accessed: 2016-07-15
44. Shao YH, Zhang CH, Wang XB, Deng NY (2011) Improvements on twin support vector machines. IEEE Trans Neural Netw 22(6):962–968
45. Shirazi H, Bezawada B, Ray I (2018) "kn0w thy doma1n name": Unbiased phishing detection using domain name based features. In: Proceedings of the 23Nd ACM on Symposium on Access Control Models and Technologies, ACM, SACMAT '18, pp 69–75, https://doi.org/10.1145/3205977.3205992
46. Srinivasa Rao R, Pais AR (2017) Detecting phishing websites using automation of human behavior. In: Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security, ACM, New York, NY, USA, CPSS '17, pp 33–42, https://doi.org/10.1145/3055186.3055188,
47. Vapnik VN, Vapnik V (1998) Statistical learning theory, vol 1. Wiley, New York
48. Varshney G, Misra M, Atrey PK (2016) A phish detector using lightweight search features. Comput Secur 62:213–228. https://doi.org/10.1016/j.cose.2016.08.003
49. Wenyin L, Huang G, Xiaoyue L, Min Z, Deng X (2005) Detection of phishing webpages based on visual similarity. In: Special interest tracks and posters of the 14th international conference on World Wide Web, ACM, pp 1060–1061
50. Xiang G, Hong JI (2009) A hybrid phish detection approach by identity discovery and keywords retrieval. In: Proceedings of the 18th international conference on World wide web, ACM, pp 571–580
51. Xiang G, Hong J, Rose CP, Cranor L (2011) Cantina+: a feature-rich machine learning framework for detecting phishing web sites. ACM Trans Inf Syst Secur TISSEC 14(2):21. https://doi.org/10.1145/2019599.2019606
52. Yang P, Zhao G, Zeng P (2019) Phishing website detection based on multidimensional features driven by deep learning. IEEE Access 7:15196–15209. https://doi.org/10.1109/ACCESS.2019.2892066
53. Zhang H, Liu G, Chow TW, Liu W (2011) Textual and visual content-based anti-phishing: a bayesian approach. IEEE Trans Neural Netw 22(10):1532–1546
54. Zhang Y, Hong JI, Cranor LF (2007) Cantina: a content-based approach to detecting phishing web sites. In: Proceedings of the 16th international conference on World Wide Web, ACM, pp 639–648, https://doi.org/10.1145/1242572.1242659, http://dl.acm.org/citation.cfm?id=1242659