International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

PhishShield: ML Based-Powered Phishing [Client-Side Protection Against Web Spoofing Attacks]

Veni S[1], Mr. K. Arun Kumar[2]

[1]MSc Computer Science, Rathinam College of Arts & Science, Coimbatore veni150801@gmail.com
[2]Department Of Computer Science, Rathinam College of Arts & Science, Coimbatore

ABSTRACT

The realm of cybersecurity faces a monumental task in safeguarding the confidentiality and integrity of users' sensitive information, such as passwords and PIN codes. Each day, billions of users encounter fraudulent login pages seeking confidential data. Various methods, including phishing emails, enticing advertisements, click-jacking, malware, SQL injection, session hijacking, man-in-the-middle, denial of service, and cross-site scripting attacks, are employed to lure users to deceptive web pages.

Web spoofing, or phishing, involves the creation of a malicious replica of a legitimate web page to illicitly solicit users' private information, including passwords. Despite numerous security strategies proposed by researchers, latency and accuracy remain significant challenges. In response, we present a client-side defense mechanism grounded in machine learning techniques to identify spoofed web pages and shield users from phishing attacks.

As a proof of concept, we introduce PhishShield, a Google Chrome extension developed to implement our machine learning algorithm. This algorithm evaluates a URL's legitimacy, classifying it as either suspicious or trustworthy. By analyzing four distinct types of web features, the algorithm employs a random forest classifier to determine the authenticity of a login web page. To evaluate the accuracy and precision of our extension, we conducted multiple experiments on real web applications. The results of these experiments demonstrate an impressive accuracy rate of 98.5% and a precision rate of 98.5% across 400 classified phishing URLs and 400 legitimate URLs. Additionally, we assessed the latency of PhishShield by conducting experiments on forty phishing URLs, revealing an average response time of just 62.5 milliseconds.

Keywords: Web spoofing, security and privacy, machine learning, web security, browser extension.

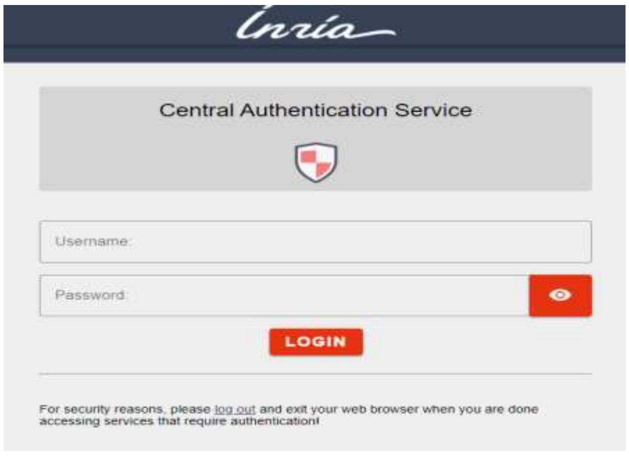
I.INTRODUCTION

In October 2022, the members of the National Institute for Research in Digital Science and Technology (Inria) in France received a French-language email requesting confirmation of their webmail accounts. The email contained a direct link, purportedly leading to the confirmation page: https://www.educationonline.nl/Cliquez.ici.cas.inria.fr.cas.login/login.html.Upon clicking this link, users were directed to a counterfeit, yet convincingly authentic, central authentication login page resembling Inria's legitimate login page (https://cas.inria.fr/cas/login?service=).

Unaware of the deception, users inadvertently entered their Inria usernames and passwords into the fake website, believing it to be genuine. The attacker could then exploit this information by submitting it to the authentic Inria login page, perpetrating a phishing attack against Inria and its registered users/members. A cautionary email alerting Coq-club Inria users about this phishing attempt was received on October 10, 2022.

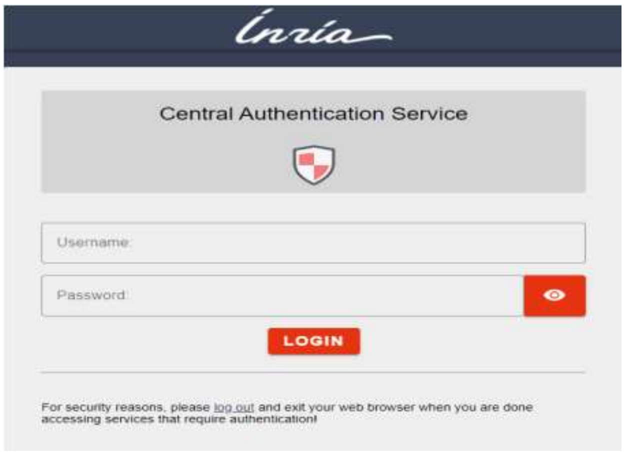
Both the real and fake Inria login pages are visually indistinguishable, making it easy for users to fall prey to such phishing attacks. With the rapid advancement of modern technologies, the online landscape has witnessed a significant expansion, encompassing various domains like e-commerce, online banking, distance learning, e-health, and e-governance. Social networking platforms such as Facebook and Twitter play pivotal roles in the global connectivity of today's world, with billions of users embracing this growing trend. Many websites offer users the option to create personalized accounts to access specialized services. Typically, users encounter login web pages where they must establish an account by providing identification (e.g., username) and a password.

When users need to access remote resources or services in the future, they submit their identification and password via a login form. However, this process exposes users to potential risks of identity theft and unauthorized access to confidential information. A phishing attack scenario, illustrated in Figure 2, unfolds with the reception of an email containing a link to a malicious website [1]. The email often contains convincing language or incentives to entice users to click and follow the link. Upon opening the web page, users may mistake it for a legitimate site where they have an account. After entering their credentials and clicking the submit/login button, the unsuspecting user unwittingly sends their sensitive information to the attacker, who orchestrated the phishing attack.



For security reasons, please log out and exit your web browser when you are done accessing services that require authentication!

Real Login Web Page of Inria



For security reasons, please log out and exit your web browser when you are done accessing services that require authentication!

Fake Login Web Page of Inria

FIGURE 1. Phishing attack on Inria

Identity theft, online fraud, and scams have surged since the emergence of web spoofing or phishing attacks. Web spoofing, a form of cybercrime, involves malicious individuals attempting to steal valuable data from users. Attackers employ various phishing and web spoofing techniques to compromise online systems. Initially utilized for identity theft, web spoofing now targets sensitive information related to national security, intellectual property, and organizational secrets. The evolution of phishing attacks has introduced new tactics such as QR code phishing, mobile application spoofing, and spear phishing, among others.

Despite efforts to bolster security measures like firewalls, digital certificates, encryption software, and two-factor authentication, these advanced scam techniques have rendered many systems vulnerable [2]. Although numerous companies employ two-factor authentication systems to mitigate monetary scams and identity theft, the sophistication of modern scam methods has challenged the effectiveness of these security measures.

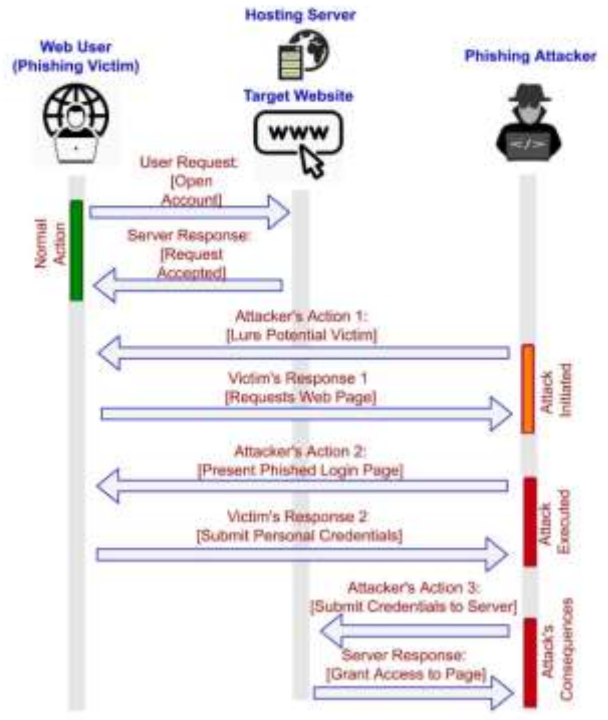

Hosting Server
Web User (Phishing Victim)
Phishing Attacker
Target Website
WWW
User Request: [Open Account]
Normal Action
Server Response: [Request Accepted]
Attacker's Action 1: [Lure Potential Victim]
Victim's Response 1 [Requests Web Page]
Attack Initiated
Attacker's Action 2: [Present Phished Login Page]
Victim's Response 2 [Submit Personal Credentials]
Attack Executed
Attacker's Action 3: [Submit Credentials to Server]
Server Response: [Grant Access to Page]
Attack's Consequences

FIGURE 2. A typical phishing attack.

To ensnare unsuspecting victims, attackers commonly incorporate logos, either by storing copies or embedding links to logos from legitimate sites onto their spoofed websites to mimic their appearance. Additionally, attackers may include HTML from the authentic site and make necessary modifications. Phishing attack vectors utilized by attackers include email, trojan horses, keyloggers, and man-in-the-middle proxies. The preferred targets of attackers are online banking sites, third-party payment systems (the most targeted industry sector), and e-commerce sites [3]. Since phishers target non-cryptographic components, cryptographic security protocols SSL/TLS alone do not offer a comprehensive solution. To defend against spoofing attacks, these protocols must be supplemented with additional protective measures [4]. These mechanisms may be implemented server-side, client-side, or both. While server-side solutions [5], [6] necessitate website modifications, which can be cumbersome and often overlooked by developers [7], client-side solutions offer user protection without server support. While server-side solutions may effectively identify spoofed sites, this paper focuses on client-side solutions. Most anti-spoofing tools rely on third-party certification [8], passwords [9], or URLs [3].

Anti-spoofing tools are sometimes categorized as stateful or stateless and classified based on the automatic phishing detection mechanism used: blacklists and heuristics. Tools relying on black/white lists generate minimal false positives (accuracy) and can identify nearly 90% of phishing sites [10]. However, they may miss zero-day attacks [11]. Moreover, blacklisting methodologies have drawbacks as they cannot adapt to changing domains and new attacks and can be easily fooled by spam URLs [12]. Heuristic-based techniques have been promising in capturing phish sites not included in blacklists. Heuristic (content)-based tools like CANTINA [13] and SpoofCatch [1] can identify 90% of phishing sites with 1% false positives. SpoofCatch's latency is in the order of seconds and increases over time. Stateful anti-phish techniques excel in accuracy but quickly fill local storage, leading to performance degradation over time. In SpoofCatch, visual similarity is initially compared with a few login page images, but as users browse more websites, the number of stored login page images increases, prolonging the time required to compare images. Following this research direction, we design and develop a stateless anti-phish tool based on Machine Learning (ML) techniques.

In the past decade, numerous renowned researchers have proposed machine learning techniques for detecting malicious URLs to prevent future scams. Multiple sets of URLs serve as training data in ML approaches. Based on statistical properties derived from training sets, it is determined whether the requested URL is a scam or not. Training data collection is crucial for URL identification using ML. Once training data is obtained, it is further processed to derive a mathematical model. Feature collection from training data is essential since simple strings may not suffice to predict the status of the URL under test. Ultimately, an actual model is derived from the predicted model obtained from the training data. Machine learning techniques such as Naïve Bayes, Support Vector Machines (SVM), and Logistic Regression (LR) are among the algorithms used for this purpose by many scholars, although several issues make them susceptible [14].

In this paper, we introduce and create a stateless client-side tool named PhishShield to combat web spoofing attacks. PhishShield, designed as a Google Chrome extension, leverages machine learning techniques and implements the random forest algorithm to determine whether a login web page is genuine or spoofed. We conducted thorough evaluations of PhishShield on real web applications, yielding remarkable results. The source code for the PhishShield Google Chrome extension is publicly accessible online at https://github.com/wilstef/PhishShield.

The primary contributions of our research are as follows:

1) Proposal of a client-side anti-phishing mechanism employing machine learning.

2) Development of the PhishShield Google Chrome extension, which embodies the proposed mechanism.

3) Thoughtful selection of web features for the phishing classifier algorithm utilized in the extension.

4) Experimental analysis of PhishShield's performance.

The structure of the remainder of this paper is as follows: We provide a summary of related literature in the subsequent section. Section III delves into the detailed research methodology employed in this study. The design and development process of the Google Chrome extension are outlined in Section IV. Section V presents the testing results of the Chrome extension, while Section VI evaluates its effectiveness. Finally, we conclude our paper in Section VII.

II. RELATED WORK

At present, numerous open-source methodologies exist to shield users against phishing attacks; however, many of them come with certain limitations such as latency, restricted feature sets, and generic databases.

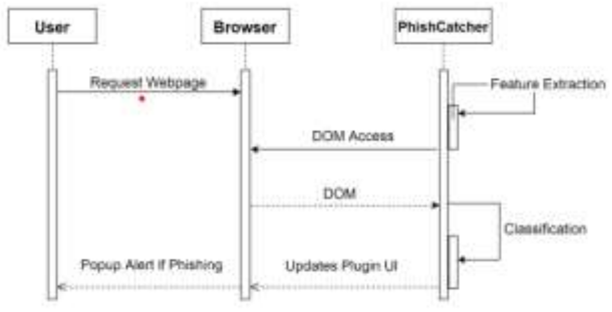


FIGURE 3. Summary of the anti-phishing schemes.

This section offers an overview of the existing anti-phishing tools and frameworks utilized for detecting and thwarting phishing attempts. These anti-phishing tools and techniques are segmented into seven primary schemes, as outlined in Figure 3 and elaborated upon in the subsequent subsections.

A. VISUAL SIMILARITY AND PAGE CONTENT ANALYSIS

One anti-phishing approach grounded in visual similarity hinges on analyzing the visual elements of received web pages. Wilayat et al. [1] engineered a phish identification tool named SpoofCatch, which relies on visual comparison. SpoofCatch captures a screenshot of a website's login page upon the user's initial visit and compares subsequent visits' login page screenshots against locally stored ones. A match indicates legitimacy, while differences signal potential phishing. Notably, [15] proposes a method to visually distinguish suspected phishing sites from legitimate ones using three key web features: text fragments and layout, embedded images, and overall visual presentation. Experimental tests with real-world phishing sites yielded promising results regarding error rates. Meanwhile, [16] suggests a novel phishing prevention approach centered on spatial design attributes of web pages. This methodology involves extracting spatial arrangement attributes and creating an R-tree to facilitate phishing identification based on spatial layout similarities. Zhang et al. [13] adopt a content-focused strategy employing Term Frequency-Inverse Document Frequency (TF-IDF) filtering to accurately detect 95% of phishing URLs. Additionally, [18] introduces a browser extension PWDHASH++ for client-side protection, which identifies visual similarities between web pages using Gestalt philosophy, treating each page as a single, indivisible entity, evaluated via algorithmic complexity analysis.

B. INTEGRATED STRATEGIES FOR PHISHING DETECTION

In [20], the authors present a comprehensive spoofing and phishing detection framework, employing a two-step process primarily driven by deep learning. They introduce the Dynamic Category Decision Algorithm (DCDA) based on deep learning, which efficiently processes over a million malicious URLs. Results indicate that their algorithm significantly reduces the time required to detect web spoofing. Another study [21] proposes a hybrid machine learning approach to counter phishing threats, leveraging five distinct machine learning techniques. The proposed four-layered model outperforms existing ones after training on a substantial dataset comprising numerous URLs, showcasing superior efficiency and effectiveness.

Kaur and Sharma [22] deploy the Repeated Incremental Pruning to Produce Error Reduction (RIPPER) algorithm [23] for detecting malicious emails. Notably, upon identifying a phished URL, their system automatically generates an email to the victim server, furnishing details such as the attacker's IP, location, and contact information. It also blocks all traffic originating from servers with malicious intent. In [24], researchers combine machine learning with the Resource Description Framework (RDF) to minimize false positives and enhance the accuracy of their model.

Furthermore, [25] explores various machine learning techniques, including Linear Model (LM), Decision Tree (DT), Random Forest (RF), and Neural Networks (NNs), to identify phishing and malicious websites in test data. These studies collectively contribute to the advancement of integrated strategies for detecting and mitigating phishing threats in diverse online environments.

C. MACHINE LEARNING TECHNIQUES FOR ANTI-PHISHING

Numerous researchers have developed robust and reliable solutions using machine learning techniques for detecting malicious URLs. Mao et al. [26] identified specific attributes of web pages to recognize phishing URLs and designed a logistic regression classifier for this purpose. Their findings revealed that approximately 777 phishing websites were visited daily, affecting around 8.24% of users.

In [14], authors evaluated nine machine learning methodologies, including LR, RF, AdaBoost, SVM, NN, Naïve Bayes, Bagging, and Bayesian additive regression, based on a dataset of 1500 phishing URLs. They trained the dataset using machine learning techniques. Similarly, researchers in [27] introduced a scalable classifier for phishing detection, trained on noisy datasets, and achieved a detection rate of about 90% for malicious URLs.

In [28], the authors employed a PART-algorithm for spoof detection and enhanced the detection process using MAP-REDUCE [29]. Jain et al. [30] conducted a comprehensive survey on phishing detection techniques worldwide. They described an NLP model based on machine learning for identifying illegitimate social media accounts [31] and used an SVM tool to expedite the process.

Xiang et al. [32] proposed an anti-phishing approach based on the CANTINA+ model, adopting a filtering algorithm to reduce false positives. Lakshmi and Vijaya [33] utilized supervised machine learning techniques, including multi-layer perceptron, Naïve Bayes classifier, and decision tree classifier, to classify and predict malicious websites based on features extracted from a collection of 200 URLs and HTML source codes. Their findings favored the decision tree classifier.

Furthermore, [34] offers a detailed analysis of machine learning approaches for identifying malicious URLs, highlighting different aspects such as feature description and algorithm architecture. Random Forest Tree-based (RFT) algorithm and SVM are evaluated for facial recognition in [35], optimizing kernel parameters to assess efficiency.

Yu et al. [36] proposed a strategic APT detection approach using deep learning in IIoT, employing the BERT model to detect APT attack patterns. Empirical results indicate that the BERT system has high precision and lower error rates compared to existing statistical models for identifying APT attack sequences.

D. STRATEGIES FOR PHISHING PREVENTION THROUGH ONLINE TRAINING

Despite the significant impact of web spoofing and phishing attacks on users, various browser and server-based techniques have been proposed to mitigate such threats [4], [40]. A comprehensive investigation into the security of login pages was conducted in [41], where the authors developed an effective attacker model to assess login security. Their evaluation revealed that approximately 63% of the tested login pages were susceptible to attacks.

In another study [42], researchers conducted a survey to identify fraudulent websites using online learning strategies that incorporate lexical and host-based attributes of URLs. They emphasized the relevance of this program to online algorithms due to the large scale of training data. A real-time method was devised to capture URL attributes and labeled URLs from a web mail provider, demonstrating that newly established online algorithms, such as batch strategies, achieve a classification accuracy of 99% across diverse datasets.

Authors in [43] demonstrated precise identification of phishing emails using a specific filter that considers parameters relevant to phishing attacks, yielding a correct recognition rate of 96% with only a 0.1% classification error. Additionally, [44] proposed a phishing identification method that assesses website security by analyzing the source code for phishing features outlined by the World Wide Web Consortium (W3C) guidelines. The security percentage is determined based on the presence of phishing parameters in the source code.

Kumaraguru et al. [45] introduced and evaluated an embedded email training scheme to educate users about phishing. Laboratory experiments compared the efficacy of conventional phishing safety notifications with two integrated learning models proposed in the study, demonstrating superior performance of integrated training over existing safety notification procedures.

E. AUTOMATED DETECTION OF FAKE AND LEGITIMATE WEBSITES

The Automated Individual White List (AIWL) presents an innovative anti-phishing strategy aimed at maintaining a white-list of known Login User Interfaces (LUIs) of websites for users [46]. It employs a Naive Bayesian classifier to automatically update the white list and warn users of potential threats if sensitive details are submitted to an unlisted LUI.

In [47], the architecture and optimization techniques of a scalable machine learning classifier for detecting suspicious websites are discussed. This classifier dynamically manages Google's blacklist by scrutinizing millions of pages daily, evaluating URL and page content to determine the authenticity of a website.

F. URL INSPECTION FOR PHISHING DETECTION

A lightweight phishing detection method based on URL characteristics was introduced by researchers in [48]. Their dataset comprised 1000 legitimate and 1000 fraudulent URLs, which were evaluated using Support Vector Machines (SVM). This method utilizes only six URL attributes for identification, with the similarity index being a key feature used for the first time.

In another study [49], an approach for automated classification of fake and legitimate URLs using supervised learning over lexical and host-based features was proposed. This approach complements previous techniques like blacklisting, as it can predict the status of previously unvisited URLs, which blacklisting cannot. Additionally, it eliminates the need to visit potentially harmful sites for models evaluating site content and behavior.

Khonji et al. [50] initiated research to test the functional efficacy of website classification through lexical evaluation of URL tokens, along with an innovative tokenization method to enhance prediction efficiency. They conducted experiments using an experimental HTTP proxy server, analyzing over 70,000 valid and phishing URLs gathered from various sources. A predictive classification model was developed to assess the effectiveness of lexical URL analysis.

As phishing emails often contain malicious URLs, enhancing website detection procedures can improve the performance of anti-phishing email classifiers. Khonji et al. [51] expanded their study to enhance the classification accuracy of anti-phishing email filters using lexical URL analysis.

G. NOTABLE ANTI-PHISHING TOOLS

In [52], authors designed and developed the Spoofguard browser extension, which displays a window where photographic passwords reveal user credentials. The extension assigns a separate password to each URL for client-side protection and alerts users of potential scams.

Yue et al. [11] developed the BOGUSBITER anti-phishing client-side tool, which operates on an offensive defense strategy by feeding bogus data to malicious phishing sites, making it challenging for them to distinguish between real and fake datasets.

In another effort [53], authors highlighted the severity of threats posed by large-scale web crawling, leading to the creation of the MadTracer detection tool. MadTracer effectively combats malvertising and captures significantly more harmful domain tracks than Google's Safe Browsing and Microsoft Forefront combined.

Prophiler [56] aims to reduce the number of web pages needing automatic evaluation to identify harmful websites. It acts as a front-end for Wepawet, a public analytics platform for network malware, significantly reducing Wepawet's load with minimal error.

Imran et al. developed DAISY [58], a lightweight identification and prevention system, to defend software-defined networks (SDN) against DoS attacks. Unlike techniques that only restrict hosts or ports, DAISY can reactivate a port or host once it is no longer receiving malicious traffic, resulting in improved SDN performance across various metrics.

III. METHODOLOGY

In our research methodology, we commenced by reviewing pertinent literature to gain insights into the latest developments concerning phishing attacks, web spoofing, machine learning, and various mechanisms employed for detecting suspicious login pages along with their advantages and drawbacks. Subsequently, we delved into the exploration of several machine learning-based frameworks aimed at identifying malicious login pages, as detailed in Section II. The comparison of these anti-phishing tools with our plug-ins is presented in Section VI. Additionally, we conducted Document Object Model (DOM) analysis and utilized JavaScript and Python practices to develop an innovative and sophisticated Google Chrome extension for detecting spoofing attacks. The core concept was to create a Google Chrome add-on that functions as a classifier of fake and authentic login pages and displays phishing warnings on the user's screen.

Prior to selecting a suitable classifier model, it was imperative to identify desirable features. To accomplish this, we primarily focused on the set of features widely implemented in existing frameworks, as discussed in the related work section. Eventually, we curated the most prominent, effective, and easy-to-integrate features for our classifier. Our feature set comprises the following:

- Collection of fake and legitimate login page image pairs (visual similarity-based)

- URL parameters (URL-based)

- Web page content (content-based)

- Blacklist

Traditional classifiers employed techniques such as whitelisting, blacklisting, online learning strategies, and lexical and host-based analysis of URLs, as indicated in Section II. Blacklisting alone proved inefficient as it did not anticipate the status of previously unvisited URLs. Moreover, classifiers based on online strategies exhibited inaccuracies, while whitelisting and lexical-based models demonstrated high latency.

Following web page feature extraction, a random forest classifier model was selected based on performance metrics such as latency, accuracy, and efficiency. Subsequently, the classifier underwent training using supervised machine learning techniques. The extracted features were then inputted into the selected model to complete the learning process. Once the learning process concluded, the model became ready for testing and simulation, enabling predictions regarding the authenticity of login web page responses.

The primary objective was to achieve efficiency in terms of latency, false positives, and false negatives.

A. SELECTION OF MODEL

In the realm of identifying phishing attacks, data mining methods stand out as particularly effective. Subasi et al. [60] utilized various data mining strategies to differentiate between legitimate and phishing web pages. They employed multiple classifiers to construct a robust phishing detection system, with the random forest classifier demonstrating superior performance in detecting phishing attempts. However, these methods typically rely on machine learning libraries written in Python, making real-time execution within most browsers impractical. The primary aim of our research is to develop a client-side tool capable of detecting phishing attacks in real-time.

Traditionally, predictions are made on the server side, and the plugin then communicates with the server to check the status of each web page. While this server-based approach is effective, it is vulnerable to compromise if web developers do not adhere to standard practices, potentially impacting all visiting users [61].

In contrast to the conventional approach, our proposal advocates for running the classification algorithm directly inside the browser rather than on the server. This approach offers several advantages, including enhanced privacy (as the user's browsing data remains on their machine) and independence from network latency. Similar to [60], we have implemented our technique using JavaScript, a scripting language compatible with browser plugins. However, given JavaScript's limited support for machine learning libraries and the constrained processing capabilities of client machines, our implementation prioritizes lightweight design.

PhishShield facilitates the feature extraction process and classification directly within the client's browser, promptly alerting users to potential phishing threats on their screen.

B. DATA PREPARATION

In this phase, the selection of appropriate datasets is crucial for extracting relevant features. Our dataset comprises information from four distinct sources:

- Mohammad et al. [62] have identified highly effective features that have demonstrated their efficacy in detecting phishing attacks. This dataset is accessible through the UCI Machine Learning Repository [63].

- Jalalian et al. have compiled a comprehensive collection of 90 compromised journal websites [64], which we utilized for testing and evaluating our classifier.

- We obtained a set of 310 blacklisted URLs from PhishTank [55].

- Additionally, we collected a set of 310 legitimate URLs from moz.com/top500.

C. FEATURE EXTRACTION

This phase presents the most challenging aspect of our study. We encountered various hurdles, including the absence of suitable datasets. Several researchers [25], [28], [31] have proposed anti-phishing mechanisms based on data mining and machine learning techniques. However, many of these training datasets lack reliability, accessibility, and are often based on generalized rules. There is a lack of consensus in the literature regarding the definitive attributes that distinguish phishing websites, complicating the formulation of comprehensive datasets.

Despite these challenges, we endeavored to curate a set of features best suited for our model through a meticulous analysis of existing strategies outlined in the literature. The dataset suggested by Mohammad et al. [62] emerged as the most prominent among these techniques.

Our feature set is categorized into four groups:

1) Group-1: Address bar-based features

2) Group-2: Abnormality-based features

3) Group-3: HTML and JavaScript-based features

4) Group-4: Domain-based features

D.PHISHING CLASSIFICATION AND CLASSIFIER SELECTION

In our model, the classification process, which serves as the cornerstone of machine learning, adopts a supervised learning approach. Researchers have explored various tools and machine learning techniques to evaluate their effectiveness in detecting phishing attacks [14]. An insightful comparison of commonly used machine learning techniques for network intrusion detection is presented in [65], where standard machine learning classifiers are evaluated using openly available datasets, namely KDD99 and UNSW-NB15 [66]. The development time for each classifier is also assessed to gauge its efficiency.

The findings indicate that Decision Tree (DT), Random Forest (RF), Hoeffding Tree (HT), and K-Nearest Neighbors (KNN) classifiers outperform others in the 10-fold cross-validation test mode. Upon careful analysis of existing strategies for phishing attack identification, the random forest algorithm emerges as superior. By creating and consolidating multiple decision trees, the random forest algorithm provides a robust and reliable forecast. It stands out as a versatile, user-friendly, and widely used supervised machine learning algorithm, often yielding optimal results even without hyper-parameter optimization.

The random forest approach is highly flexible and applicable to both regression and classification problems, making up a significant portion of modern ML systems. Utilizing the bagging technique, the algorithm forms an ensemble of decision trees to enhance overall performance. This strategy improves predictive accuracy, prevents data overfitting, and facilitates rapid training with the dataset. Moreover, it accommodates a broad range of results in high-dimensional spaces, further enhancing accuracy.

Our proposed model for lightweight phishing identification using the random forest classifier is illustrated in Figure 4. Initially, a suitable dataset is selected, as discussed in sub-section III-B. Next, relevant features are extracted based on their performance and compatibility, categorized into four groups, as outlined in sub-section III-C, where each group serves as a decision tree.

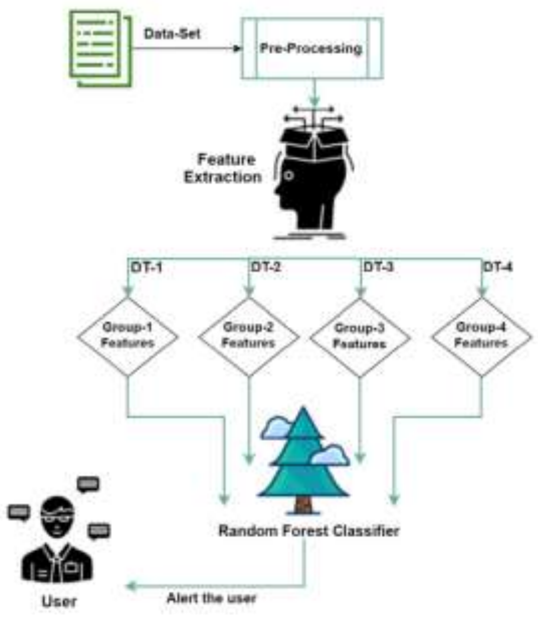


FIGURE 4. Random forest classifier for phishing detection.

These feature groups are then inputted into the random forest classifier for identifying and classifying phishing URLs. In essence, the classifier alerts users to potential phishing attacks, a functionality implemented in the PhishShield browser extension through alert notifications.

IV. EXTENSION ARCHITECTURE

Browser extensions, also known as add-ons, are small software packages designed to customize and enhance the browsing experience based on user preferences. They are developed using web-based programming languages such as HTML, CSS, and JavaScript. This section presents an overview of the design and development process of PhishShield, a Google Chrome extension aimed at identifying and safeguarding against phishing attacks. The primary objective is to perform classification directly within the client's browser, providing real-time results while enhancing user privacy and reducing latency.

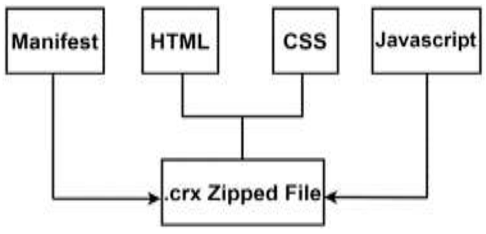


FIGURE 5. Layout of the Google Chrome extension.

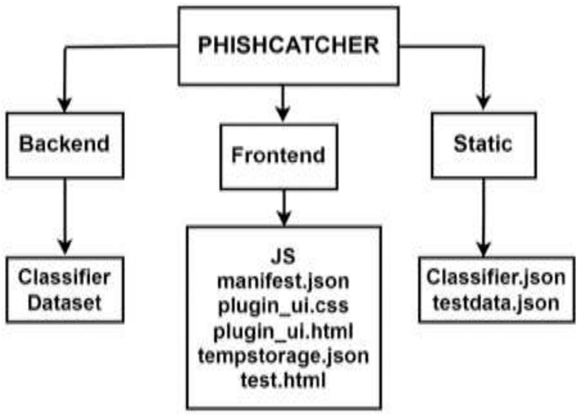


FIGURE 6. PhishShield architecture.

The architecture of a Google Chrome extension typically comprises backend, frontend, and static modules. These components serve distinct functions and are constructed using different code and content types, as illustrated in Figure 6. The backend module hosts the classifier, which operates on the dataset, while the static module contains essential files such as testdata.json and classifier.json. The frontend encompasses executable code written in JavaScript, CSS, and HTML.

Fundamentally, an extension is characterized by a manifest file, manifest.json, in JSON format, which stores essential information about the extension, including its name, version, description, browser action, and required permissions. The extension directory must contain this manifest file, which serves as a blueprint for the extension's functionality.

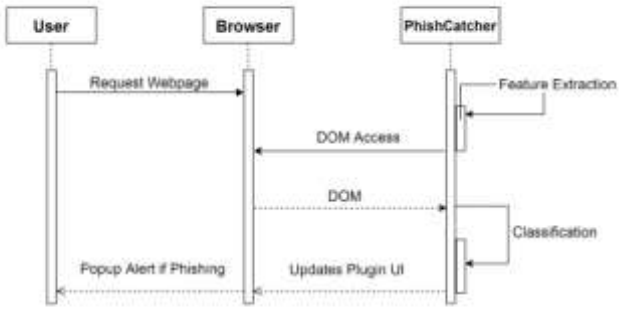


FIGURE 7. Use case diagram.

A use case diagram (Figure 7) illustrates the interaction between the user and the system, depicting various user interactions with the browser across different scenarios. It visualizes the operational flow between the user and the system, ensuring a seamless user experience. PhishShield is designed to be intuitive and user-friendly, providing a phishing alert when a suspicious URL is accessed. The graphical user interface highlights the features responsible for classifying a URL as phishing. Additionally, PhishShield includes a whitelist feature, allowing users to mark certain pages as genuine and exempt them from blocking. Users can simply click "OK" to add a webpage to the whitelist, as demonstrated in the test cases outlined in Section V.

V. TESTING

To assess the performance of the PhishShield, we tested and evaluated it against the real web application scenarios.

This study mainly focuses on the aggregated analysis of all the features under consideration for the classification of

legitimate and bogus URLs, rather than applying unit testing method for each feature. Nevertheless, screen shots of a few tested URLs taken by PhishShield are also presented here.

The data-set considered in these tests contains:

• set of legitimate and corresponding fake URLs of

90 hijacked journals [64],

• set of 310 blacklisted URLs from PhishTank [55] and

• set of 310 legitimate URLs from the website

https://moz.com/top500.

After several experiments, we extracted seventeen prominent features, categorically listed in the Table 2. Sub-sets of these features have been previously used in different tools and

analysis [62], [68]. The significance of each feature varies

according to the nature of the URL being tested by our

plug-in.

A. PERFORMANCE ANALYSIS

In this section, we present an overview of the results obtained through the implementation of PhishShield, focusing on the testing of fraudulent and legitimate URLs from our dataset. We introduce select test cases to offer insights into the performance and efficacy of our approach.

No.	Feature Name	Category
1.	IP address	Address bar based features
2.	URL length	Address bar based features
3.	Tiny URL	Address bar based features
4.	"@" symbol	Address bar based features
5.	Redirecting using //	Address bar based features
6.	Prefix/suffix in the domain	Address bar based features
7.	No. of sub-domains	Address bar based features
8.	HTTPS	Address bar based features
9.	Favicon	Address bar based features
10.	Using non-standard port	Address bar based features
11.	HTTPS in URL's domain part	Address bar based features
12.	Request URL	Abnormal based features
13.	URL of anchor	Abnormal based features
14.	Links in meta, script and link	Abnormal based features
15.	Server form handler	Abnormal based features
16.	Submitting to mail	Abnormal based features
17.	Using iFrame	HTML/JavaScript features

A. TEST CASES

This section includes a concise view of the results generated by the PhishCatcher in terms of testing fraudulent and reliable URLs from our data-set. A few test cases are introduced here to provide a better understanding and performance analysis of our approach.

Test Case 1

URL:https://www.education-online.nl/Cliquez.ici.cas.inria.fr.cas.login/login.html

Result: Phishing

Our initial test case evaluates PhishShield's response to a recent phishing attack targeting Inria, as discussed in Section I. Despite the sophisticated nature of the attack, wherein the perpetrators meticulously replicated Inria's login webpage, our tool successfully identifies it as phishing.

Both the authentic URL (https://cas.inria.fr/cas/login?service=) and the deceptive URL (https://www.education online.nl/Cliquez.ici.cas.inria.fr.cas.login/login.html) underwent scrutiny, with PhishShield accurately distinguishing between genuine and counterfeit login pages. Among the seventeen features analyzed for phishing URLs (refer to Table 2), six specific attributes—including URL length, domain prefix/suffix length, favicon presence, request URL, anchor, and script link—contributed significantly to the detection of this phishing attempt.

Test case 2

URL: http://www.ijiq.com

Result: Identified as Phishing

In the second test case, we simulate a scenario involving a spam URL masquerading as a legitimate journal website. Users receive promotional emails containing a link to the bogus site www.ijiq.com. Upon clicking the link, the user's browser loads the attacker's website, which appears trustworthy, prompting the user to enter their credentials. Our plugin successfully detects this exploit, triggering a phishing alert .

Test case 3

URL: http://www.revistas-academicas.com

Result: Identified as Phishing

VI. EVALUATION

The proposed model underwent a series of trials to evaluate the accuracy and latency of our tool. Latency experiment results are presented and discussed in subsection VI-B. Other performance metrics, including precision, recall, and accuracy, were recorded in the form of a confusion matrix for further analysis.

A. PERFORMANCE METRICS

A confusion matrix is a structured table used to evaluate the performance of a classifier. It assesses the effectiveness of a supervised learning algorithm across a set of test data with known values. Each row of the matrix represents the instances in a predicted class, while each column represents the cases in an actual class. Precision, recall, and accuracy, key performance metrics of our plug-in, are calculated using Equations 1, 2, and 3, respectively.

$$\frac{TP}{TP + FP} \quad (1)$$

$$\frac{TP}{TP + FN} \quad (2)$$

$$\frac{TP + TN}{TP + FP + TN + FN} \quad (3)$$

The values of variables have been assigned The variables in these equations are derived from the confusion matrix, where TP represents True Positive, FP denotes False Positive, TN indicates True Negative, and FN represents False Negative. Here, P and N signify Positive and Negative, respectively. True positive occurs when a phished URL is correctly identified, while false positive happens when a legitimate URL is mistakenly classified as phished. Conversely, true negative denotes the correct identification of a legitimate URL, while false negative refers to a phished URL being incorrectly classified as legitimate.

Our experiments were conducted on a dataset of 800 URLs, comprising 400 phished and 400 legitimate URLs, to classify fake and authentic URLs. The scores were recorded after multiple iterations and thorough analysis of the extension. Consequently, PhishShield demonstrated outstanding accuracy of 98.5%, precision of 98.5%, and recall of 98.5%.

B. LATENCY

Latency refers to the speed or the time taken by an anti-phishing tool to detect a phishing attempt. It is influenced by various factors including the algorithm used, computing resources, network speed, and the nature of the tool (stateless or stateful). In the case of a stateful tool, the decision-making process involves analyzing both the current web page and the previously stored data, either locally or on a remote server. The latency of such tools depends on factors like the algorithm complexity, network speed, and the size of the stored data.

In contrast, a stateless tool like PhishShield does not require previous data and makes decisions solely based on the implemented algorithm. To measure the latency of PhishShield, we conducted experiments by testing it on forty phished URLs. Before launching the extension in the browser, we updated the code to record the start time when the computation begins and the decision time when the result is announced.

The start time is recorded just before the computation starts to extract features and run the classifier for identifying phishing attacks. Once the computation concludes and the phishing attack is identified, the decision time is captured. The difference between the decision and start time represents the time taken to determine whether a URL is phished or not.

In our experiments, conducted on a Windows 10 system with a 64-bit Intel® Core i7 CPU @ 3.40GHz and 8GB RAM, the average latency of PhishShield was found to be 62.5 milliseconds across the forty URLs tested.

Clearly, stateless tools like PhishShield exhibit faster performance compared to stateful tools. To empirically assess the latency difference between PhishShield and a stateful tool, we conducted experiments using SpoofCatch [1] on the same machine. It's important to note that the experimental requirements for SpoofCatch and PhishShield differ.

SpoofCatch necessitates that a legitimate URL be opened at least once in the browser before accessing the phished URL. Similar to PhishShield, we modified the source code of SpoofCatch to record the start and decision times. The average latency observed for SpoofCatch was 512 milliseconds, and this latency increased as the number of experiments expanded.

The latency degradation in SpoofCatch can be attributed to its method of operation. Each time SpoofCatch captures a login web page, it stores it in the local storage. As the number of web pages stored locally increases over time, the tool must compare each current web page with all previously visited pages stored in the local storage, resulting in increased processing time.

VII. CONCLUSION AND FUTURE WORK

In today's digital landscape, online applications play a vital role in various domains, including online banking, e-commerce, social networking, digital libraries, healthcare services, online education, digital marketing, and gaming platforms. Users rely on authentication procedures to create accounts and access private web content. However, the security and privacy of users are constantly threatened by sophisticated web spoofing attacks. While numerous research and commercial tools have been developed to combat these attacks, many still have limitations.

We have introduced an optimized and user-friendly browser plug-in called PhishShield, designed to intelligently detect phishing attacks using supervised machine learning. Unlike traditional approaches, our solution performs classification directly within the browser, addressing latency issues and improving tool efficiency. The user interface is designed to be intuitive, providing clear phishing alerts and highlighting relevant phishing features in a drop-down menu when a user encounters a suspicious URL.

Our feature set comprises thirty categorized features organized into decision trees. A random forest classifier leverages the collective decision of these trees to distinguish between bogus and genuine login pages. We evaluated our plug-in using a dataset of 400 malicious and 400 legitimate URLs, employing a confusion matrix to measure true positives, true negatives, false positives, and false negatives. PhishShield demonstrated remarkable classification results with precision and recall both at 98.5%, and an accuracy of 98.5%.

Looking ahead, enhancing our feature set with more automated features could further improve performance. Additionally, implementing other discriminative classifiers like Support Vector Machines (SVM) and training them on larger datasets can enhance the prediction of fake or real URLs. We plan to evolve evaluation metrics using different tools for more comprehensive performance analysis. These future enhancements aim to strengthen PhishShield's capabilities and provide users with robust protection against evolving web spoofing threats.

REFERENCES

[1] W. Khan, A. Ahmad, A. Qamar, M. Kamran, and M. Altaf, ‘‘SpoofCatch: A client-side protection tool against phishing attacks,’’ IT Prof., vol. 23, no. 2, pp. 65–74, Mar. 2021.

[2] B. Schneier, ‘‘Two-factor authentication: Too little, too late,’’ Commun. ACM, vol. 48, no. 4, p. 136, Apr. 2005.

[3] S. Garera, N. Provos, M. Chew, and A. D. Rubin, ‘‘A framework for detection and measurement of phishing attacks,’’ in Proc. ACM Workshop Recurring malcode, Nov. 2007, pp. 1–8.

[4] R. Oppliger and S. Gajek, ‘‘Effective protection against phishing and web spoofing,’’ in Proc. IFIP Int. Conf. Commun. Multimedia Secur. Cham, Switzerland: Springer, 2005, pp. 32–41.

[5] T. Pietraszek and C. V. Berghe, ‘‘Defending against injection attacks through context-sensitive string evaluation,’’ in Proc. Int. Workshop Recent Adv. Intrusion Detection. Cham, Switzerland: Springer, 2005, pp. 124–145.

[6] M. Johns, B. Braun, M. Schrank, and J. Posegga, ‘‘Reliable protection against session fixation attacks,’’ in Proc. ACM Symp. Appl. Comput., 2011, pp. 1531–1537. [7] M. Bugliesi, S. Calzavara, R. Focardi, and W. Khan, ‘‘Automatic and robust client-side protection for cookie-based sessions,’’ in Proc. Int. Symp. Eng. Secure Softw. Syst. Cham, Switzerland: Springer, 2014, pp. 161–178.

[8] A. Herzberg and A. Gbara, ‘‘Protecting (even naıve) web users from spoofing and phishing attacks,’’ Cryptol. ePrint Arch., Dept. Comput. Sci. Eng., Univ. Connecticut, Storrs, CT, USA, Tech. Rep. 2004/155, 2004.

[9] N. Chou, R. Ledesma, Y. Teraguchi, and J. Mitchell, ‘‘Client-side defense against web-based identity theft,’’ in Proc. NDSS, 2004, 1–16.

[10] B. Hämmerli and R. Sommer, Detection of Intrusions and Malware, and Vulnerability Assessment: 4th International Conference, DIMVA 2007 Lucerne, Switzerland, July 12-13, 2007 Proceedings, vol. 4579. Cham, Switzerland: Springer, 2007.

[11] C. Yue and H. Wang, ‘‘BogusBiter: A transparent protection against phishing attacks,’’ ACM Trans. Internet Technol., vol. 10, no. 2, pp. 1–31, May 2010.

[12] W. Chu, B. B. Zhu, F. Xue, X. Guan, and Z. Cai, ‘‘Protect sensitive sites from phishing attacks using features extractable from inaccessible phishing URLs,’’ in Proc. IEEE Int. Conf. Commun. (ICC), Jun. 2013, pp. 1990–1994.

[13] Y. Zhang, J. I. Hong, and L. F. Cranor, ‘‘Cantina: A content-based approach to detecting phishing web sites,’’ in Proc. 16th Int. Conf. World Wide Web, May 2007, pp. 639–648.

[14] D. Miyamoto, H. Hazeyama, and Y. Kadobayashi, ‘‘An evaluation of machine learning-based methods for detection of phishing sites,’’ in Proc. Int. Conf. Neural Inf. Process. Cham, Switzerland: Springer, 2008, pp. 539–546.

[15] E. Medvet, E. Kirda, and C. Kruegel, ‘‘Visual-similarity-based phishing detection,’’ in Proc. 4th Int. Conf. Secur. privacy Commun. Netowrks, Sep. 2008, pp. 1–6.

[16] W. Zhang, H. Lu, B. Xu, and H. Yang, ‘‘Web phishing detection based on page spatial layout similarity,’’ Informatica, vol. 37, no. 3, pp. 1–14, 2013.

[17] J. Ni, Y. Cai, G. Tang, and Y. Xie, ‘‘Collaborative filtering recommendation algorithm based on TF-IDF and user characteristics,’’ Appl. Sci., vol. 11, no. 20, p. 9554, Oct. 2021.

[18] W. Liu, X. Deng, G. Huang, and A. Y. Fu, ‘‘An antiphishing strategy based on visual similarity assessment,’’ IEEE Internet Comput., vol. 10, no. 2, pp. 58–65, Mar. 2006.

[19] A. Rusu and V. Govindaraju, ‘‘Visual CAPTCHA with handwritten image analysis,’’ in Proc. Int. Workshop Human Interact. Proofs. Berlin, Germany: Springer, 2005, pp. 42–52.

[20] P. Yang, G. Zhao, and P. Zeng, ‘‘Phishing website detection based on multidimensional features driven by deep learning,’’ IEEE Access, vol. 7, pp. 15196–15209, 2019.

[21] P. Sornsuwit and S. Jaiyen, ‘‘A new hybrid machine learning for cybersecurity threat detection based on adaptive boosting,’’ Appl. Artif. Intell., vol. 33, no. 5, pp. 462–482, Apr. 2019.

[22] S. Kaur and S. Sharma, ‘‘Detection of phishing websites using the hybrid approach,’’ Int. J. Advance Res. Eng. Technol., vol. 3, no. 8, pp. 54–57, 2015.

[23] W. W. Cohen, ‘‘Fast effective rule induction,’’ in Machine Learning Proceedings. Amsterdam, The Netherlands: Elsevier, 1995, pp. 115–123.

[24] V. Muppavarapu, A. Rajendran, and S. K. Vasudevan, ‘‘Phishing detection using RDF and random forests,’’ Int. Arab J. Inf. Technol., vol. 15, no. 5, pp. 817–824, 2018. [25] V. K. Nadar, B. Patel, V. Devmane, and U. Bhave, ‘‘Detection of phishing websites using machine learning approach,’’ in Proc. 2nd Global Conf. Advancement Technol. (GCAT). Rajasthan, Jaipur, India: Amity University, Oct. 2021, pp. 1–8.

[26] J. Mao, W. Tian, P. Li, T. Wei, and Z. Liang, ‘‘Phishing-alarm: Robust and efficient phishing detection via page component similarity,’’ IEEE Access, vol. 5, pp. 17020–17030, 2017.

[27] N. C. R. L. Y. Teraguchi and J. C. Mitchell, ‘‘Client-side defense against web-based identity theft,’’ Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, 2004. [Online]. Available: http://crypto.stanford. edu/SpoofGuard/webspoof.pdf

[28] W. Ali, ‘‘Phishing website detection based on supervised machine learning with wrapper features selection,’’ Int. J. Adv. Comput. Sci. Appl., vol. 8, no. 9, pp. 72–78, 2017.

[29] A. Sharma and D. Upadhyay, ‘‘VDBSCAN clustering with map-reduce technique,’’ in Recent Findings in Intelligent Computing Techniques. Singapore: Springer, 2018, pp. 305–314.

[30] A. K. Jain and B. B. Gupta, ‘‘Comparative analysis of features based machine learning approaches for phishing detection,’’ in Proc. 3rd Int. Conf. Comput. Sustain. Global Develop. (INDIACom), Mar. 2016, pp. 2125–2130.

[31] P. Rao, J. Gyani, and G. Narsimha, ‘‘Fake profiles identification in online social networks using machine learning and NLP,’’ Int. J. Appl. Eng. Res., vol. 13, no. 6, pp. 973–4562, 2018.

[32] G. Xiang, J. Hong, C. P. Rose, and L. Cranor, ‘‘CANTINA+: A featurerich machine learning framework for detecting phishing web sites,’’ ACM Trans. Inf. Syst. Secur., vol. 14, no. 2, pp. 1–28, Sep. 2011.

[33] V. S. Lakshmi and M. S. Vijaya, ‘‘Efficient prediction of phishing websites using supervised learning algorithms,’’ Proc. Eng., vol. 30, pp. 798–805, 2012.

[34] D. Sahoo, C. Liu, and S. C. H. Hoi, ‘‘Malicious URL detection using machine learning: A survey,’’ 2017, arXiv:1701.07179.

[35] E. Kremic and A. Subasi, ‘‘Performance of random forest and SVM in face recognition,’’ Int. Arab J. Inf. Technol., vol. 13, no. 2, pp. 287–293, 2016.

[36] K. Yu, L. Tan, S. Mumtaz, S. Al-Rubaye, A. Al-Dulaimi, A. K. Bashir, and F. A. Khan, ‘‘Securing critical infrastructures: Deep-learning-based threat detection in IIoT,’’ IEEE Commun. Mag., vol. 59, no. 10, pp. 76–82, Oct. 2021.

[37] P. Chen, L. Desmet, and C. Huygens, ‘‘A study on advanced persistent threats,’’ in Communications and Multimedia Security. Aveiro, Portugal: Springer, Sep. 2014, pp. 63–72.

[38] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, ‘‘Industrial Internet of Things: Challenges, opportunities, and directions,’’ IEEE Trans. Ind. Informat., vol. 14, no. 11, pp. 4724–4734, Nov. 2018.

[39] S. Alaparthi and M. Mishra, ‘‘Bidirectional encoder representations from transformers (BERT): A sentiment analysis Odyssey,’’ 2020, arXiv:2007.01127.

[40] P. A. Barraclough, M. A. Hossain, M. A. Tahir, G. Sexton, and N. Aslam, ‘‘Intelligent phishing detection and protection scheme for online transactions,’’ Exp. Syst. Appl., vol. 40, no. 11, pp. 4697–4706, Sep. 2013.

[41] S. Van Acker, D. Hausknecht, and A. Sabelfeld, ‘‘Measuring login webpage security,’’ in Proc. Symp. Appl. Comput., Apr. 2017, pp. 1753–1760.

[42] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, ‘‘Identifying suspicious URLs: An application of large-scale online learning,’’ in Proc. 26th Annu. Int. Conf. Mach. Learn., Jun. 2009, pp. 681–688.

[43] I. Fette, N. Sadeh, and A. Tomasic, ‘‘Learning to detect phishing emails,’’ in Proc. 16th Int. Conf. World Wide Web, May 2007, pp. 649–656.

[44] M. G. Alkhozae and O. A. Batarfi, ‘‘Phishing websites detection based on phishing characteristics in the webpage source code,’’ Int. J. Inf. Commun. Technol. Res., vol. 1, no. 6, pp. 1–9, 2011.

[45] P. Kumaraguru, Y. Rhee, A. Acquisti, L. F. Cranor, J. Hong, and E. Nunge, ‘‘Protecting people from phishing: The design and evaluation of an embedded training email system,’’ in Proc. SIGCHI Conf. Human Factors Comput. Syst., Apr. 2007, pp. 905–914.

[46] Y. Cao, W. Han, and Y. Le, ‘‘Anti-phishing based on automated individual white-list,’’ in Proc. 4th ACM workshop Digit. identity Manage., Oct. 2008, pp. 51–60.

[47] C. Whittaker, B. Ryner, and M. Nazif, ‘‘Large-scale automatic classification of phishing pages,’’ in Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS), San Diego, CA, USA, Feb./Mar. 2010.

[48] M. Zouina and B. Outtaj, ‘‘A novel lightweight URL phishing detection system using SVM and similarity index,’’ Human-Centric Comput. Inf. Sci., vol. 7, no. 1, p. 17, Dec. 2017.

[49] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, ‘‘Beyond blacklists: Learning to detect malicious web sites from suspicious URLs,’’ in Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining, Jun. 2009, pp. 1245–1254.

[50] M. Khonji, Y. Iraqi, and A. Jones, ‘‘Lexical URL analysis for discriminating phishing and legitimate websites,’’ in Proc. 8th Annu. Collaboration, Electron. Messaging, Anti-Abuse Spam Conf., Sep. 2011, pp. 109–115.

[51] M. Khonji and Y. Iraqi, ‘‘Enhancing phishing e-mail classifiers: A lexical URL analysis approach,’’ Int. J. Inf. Secur. Res., vol. 2, nos. 1–2, p. 40, 2012.

[52] V. P. Reddy, V. Radha, and M. Jindal, ‘‘Client side protection from phishing attack,’’ Int. J. Adv. Eng. Sci. Technol., vol. 3, no. 1, pp. 39–45, 2011.

[53] Z. Li, K. Zhang, Y. Xie, F. Yu, and X. Wang, ‘‘Knowing your enemy: Understanding and detecting malicious web advertising,’’ in Proc. ACM Conf. Comput. Commun. Secur., Oct. 2012, pp. 674–686.

[54] Y. Mansour, S. Muthukrishnan, and N. Nisan, ‘‘Doubleclick AD exchange auction,’’ 2012, arXiv:1204.0535.

[55] S. Bell and P. Komisarczuk, ‘‘An analysis of phishing blacklists: Google safe browsing, OpenPhish, and PhishTank,’’ in Proc. Australas. Comput. Sci. Week Multiconference, Feb. 2020, pp. 1–11.

[56] D. Canali, M. Cova, G. Vigna, and C. Kruegel, ‘‘Prophiler: A fast filter for the large-scale detection of malicious web pages,’’ in Proc. 20th Int. Conf. World wide web, Mar. 2011, pp. 197–206.

[57] S. Ford. Wepawet. (2009). [Online]. Available: http://wepawet.cs. ucsb.edu/index.php

[58] M. Imran, M. H. Durad, F. A. Khan, and H. Abbas, ‘‘DAISY: A detection and mitigation system against denial-of-service attacks in software-defined networks,’’ IEEE Syst. J., vol. 14, no. 2, pp. 1933–1944, Jun. 2020.

[59] Q. Yan, F. R. Yu, Q. Gong, and J. Li, ‘‘Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges,’’ IEEE Commun. Surveys Tuts., vol. 18, no. 1, pp. 602–622, 1st Quart., 2016.

[60] A. Subasi, E. Molah, F. Almkallawi, and T. J. Chaudhery, ‘‘Intelligent phishing website detection using random forest classifier,’’ in Proc. Int. Conf. Electr. Comput. Technol. Appl. (ICECTA), Nov. 2017, pp. 1–5.

[61] M. Bugliesi, S. Calzavara, R. Focardi, and W. Khan, ‘‘CookiExt: Patching the browser against session hijacking attacks,’’ J. Comput. Secur., vol. 23, no. 4, pp. 509–537, Sep. 2015.

[62] R. M. Mohammad, F. Thabtah, and L. McCluskey, ‘‘Phishing websites features,’’ School Comput. Eng., Univ. Huddersfield, West Yorkshire, U.K., Tech. Rep., 2015. [Online]. Available: http://eprints.hud.ac.uk/id/ eprint/24330/6/MohammadPhishing14July2015.pdf

[63] D. Dua and C. Graff. (2017). UCI Machine Learning Repository. [Online]. Available: http://archive.ics.uci.edu/ml

[64] M. Jalalian and M. Dadkhah, ‘‘The full story of 90 hijacked journals from August 2011 to June 2015,’’ Geographica Pannonica, vol. 19, no. 2, pp. 73–87, 2015.

[65] F. A. Khan and A. Gumaei, ‘‘A comparative study of machine learning classifiers for network intrusion detection,’’ in Artificial Intelligence and Security. New York, NY, USA: Springer, Jun. 2019, pp. 75–86.

[66] N. Moustafa and J. Slay, ‘‘The significant features of the UNSW-NB15 and the KDD99 data sets for network intrusion detection systems,’’ in Proc. 4th Int. Workshop Building Anal. Datasets Gathering Exper. Returns Secur. (BADGERS), Nov. 2015, pp. 25–31.

[67] Y. T. Ho, C. Wu, M. Yang, T. Chen, and Y. Chang, ‘‘Replanting your forest: NVM-friendly bagging strategy for random forest,’’ in Proc. IEEE NonVolatile Memory Syst. Appl. Symp. (NVMSA), Aug. 2019, pp. 1–6.

[68] G. Sonowal and K. S. Kuppusamy, ''PhiDMA—A phishing detection model with multi-filter approach,'' J. King Saud Univ. Comput. Inf. Sci., vol. 32, no. 1, pp. 99–112, Jan. 2020.

[69] Y. Zhang, S. Egelman, L. Cranor, and J. Hong, ''Phinding phish: Evaluating anti-phishing tools,'' Carnegie Mellon Univ., 2018, doi: 10.1184/R1/6470321.v1.

[70] A. K. Jain and B. B. Gupta, ''A machine learning based approach for phishing detection using hyperlinks information,'' J. Ambient Intell. Humanized Comput., vol. 10, no. 5, pp. 2015–2028, May 2019.