Grenze International Journal of Engineering and Technology, June Issue

Real-Time Client-Side Phishing Detection using Lightweight Machine Learning Models

Ganeshan R[1], K. Hem Charan[2] and D. Karthik[3]
[1-3]School of Computing, Vel Tech Rangarajan Dr. Sagunthala, R & D Institute of Science and Technology Chennai, India
Email: ganeshramasamy111@gmail.com, hemcharan2004@gmail.com, karthik229987@gmail.com

***Abstract*— Phishing is one of the major issues concerning online security, in which attackers attempt to gather confidential information by using fake websites with the same ”look and feel” as the actual online applications. The efficiency of the existing online phishing detection system is limited by certain limitations, including delays in database updation, failure to detect zero-day attacks, and violation of online privacy by using external servers. Keeping this issue in mind, the current paper aims to propose a novel phishing detection system called ”PhishLock.” The proposed phishing detection system is based on the use of the machine learning approach to extract lexical features based on the attributes of the URLs. The proposed system is also based on the use of the machine learning approach to extract structural features based on the Document Object Model structure of the web pages. The proposed system was developed as a prototype, and the experiments were carried out using 20,000 labeled web pages collected from PhishTank, OpenPhish, and Alexa Top Sites. The proposed system was found to have achieved 97.8**

***Index Terms*— Phishing Detection, Client-Side Security, Ma-chine Learning, Real-Time Detection, Web Spoofing, Lightweight Models.**

I. INTRODUCTION

One of the most common and costly cybercrimes is phishing. This is because, in most cases, phishing attacks masquerade as legitimate online services and trick victims into disclosing their personal identification information, financial information, and other personal information. However, it should be noted that phishing has completely changed in the last decade. It has changed from an elementary phishing attack via email to an intricate phishing attack via web impersonation, social engineering, short URLs, and the use of malicious content that is dynamically generated [1], [3].

This shows the extent of phishing attacks, and it is an indication of the requirement of effective countermeasures. The industry trend shows that there is an increase in phishing attacks. Furthermore, it has been observed that large-scale phishing attacks have been reported in the banking, ecommerce, and cloud domains. The traditional approaches have been effective in detecting the phishing attacks. However, it has been observed that the traditional approaches have limitations in detecting zero-day phishing attacks. Furthermore, it has been observed that the traditional approaches have some limitations with respect to update issues and dependency issues. This shows the requirement of developing and implementing adaptive and intelligent approaches to detect phishing attacks in real-time scenarios.

Grenze ID: 01.GIJET.12.2.550

Recent studies have shown that it is feasible to implement the approaches to detect phishing attacks using machine learning and deep learning approaches. However, it has been observed that the traditional approaches have some server-side processing issues, computational complexity, and cloud inference issues. This shows the requirement of evaluating the feasibility of the approaches with respect to issues of resource-constrained devices. In order to address the identified gaps, this paper presents a new client-side lightweight phishing detection framework, termed "PhishLock," which can be deployed inside the web browser itself. This framework exploits the lexical cues found in URLs, the structural cues found inside the DOM structures of web pages, and the lightweight optimized machine learning classifiers.

The paper has been organized as follows: In the second section, the paper presents an overview of the existing techniques for detecting phishing attacks. In the third section, the paper presents the PhishLock framework, along with its architecture and the feature engineering steps. In the fourth section, the paper presents the experiments conducted for the framework, while the fifth section presents the challenges faced. In the sixth section, the conclusions are presented

A. Key Contributions

- A completely client-side phishing detection system is implemented and demonstrated, which maintains user privacy by avoiding external servers or blacklists.
- The project provides a lightweight feature extraction pipeline that examines the lexical features of the URLs and the DOM cues, which allows for real-time detection in the browser.
- The project provides a working prototype for a browser extension that performs machine learning inference for real-time detection.
- The project provides a comparison of various supervised learning techniques for the best detection method.
- The effectiveness of the framework is demonstrated through extensive experiments, which show that it has higher accuracy, faster inference time, and lower compu-tational overhead for the end user.

II. LITERATURE REVIEW

The problem of phishing detection has been extensively researched for the last ten years, and the paradigm has shifted from rule-based heuristics to machine learning and deep learning. In the beginning, the research was focused on visual similarity-based phishing detection techniques. In the research article presented by Jain and Gupta [1], the author discussed the different visual similarity-based phishing detection techniques. The author showed how the similarity of the web pages plays a vital role in phishing detection. In another research article presented by Garera et al. [2], the author proposed a framework for measuring phishing attacks using the structural features of the URLs and statistical modeling.With the advent of intelligent detection technologies, techniques have been proposed in the domain of machine learning. Yang et al. [3] proposed a feature-driven multi-dimensional technique in the domain of deep learning. This technique has improved the accuracy of detection to a great extent. Khan et al. [4] proposed a client-side protection tool in the domain of phishing detection in real-time environments. Aburrous et al. [11] proposed a fuzzy data mining-based e-banking phishing detection model. This model has improved the accuracy of detection in uncertain conditions.Moreover, content-based and feature-rich approaches have been exten-sively investigated. Zhang et al. [13] have proposed a content-based phishing detection system named CANTINA using term frequency inverse document frequency. Xiang et al. [8] have extended this research and proposed a feature-rich phishing detection system named CANTINA+ using feature-rich ap-proaches for detecting phishing attacks. Ma et al. [12] have proven the effectiveness of detecting malicious URLs without using a blacklist and have highlighted the importance of using predictive learning techniques. Several researchers have also focused on the issue of large-scale and streaming phishing detection systems. Marchal et al. [14] have proposed a large-scale phishing detection system named PhishStorm using streaming analytics for effective detection of phishing URLs. Verma and Hossain [9] have focused on the issue of semantic feature selection for detecting phishing emails. Gupta et al.
[10] have proposed a comprehensive taxonomy of phishing detection techniques and future research directions.Security-related attack mitigation mechanisms have also been considered. Johns et al. [5] have presented a study related to session fixation attacks. In addition, Pietraszek and Berghe have presented methods for the context-sensitive evaluation of injection attacks [6]. Finally, Provos and Holz have presented a discussion related to malware warning systems at a large scale [7].Industry reports have also been published by the Anti-Phishing Working Group indicating a growing number of phishing attacks. This proves the need for the development of effective phishing detection systems.Even though the above studies have contributed a lot towards the detection of phishing attacks, there are a number of issues that need to be addressed. In most of the

studies, the detection of phishing attacks relies heavily on static feature extraction mechanisms, which are not effective for handling zero-day attacks. In addition, even though the deep learning mechanisms have proven effective for the detection of phishing attacks, the computation complexities associated with them are high, resulting in increased latency.

III. METHODOLOGY

This section describes a complete client-side framework that utilizes a Decision Tree classifier for phishing detection. The framework allows for the detection of phishing pages in real-time, based on lexical, host, and DOM features, without making use of communication with the server.
The overall architecture of the proposed PhishLock system is shown in Fig. 1.

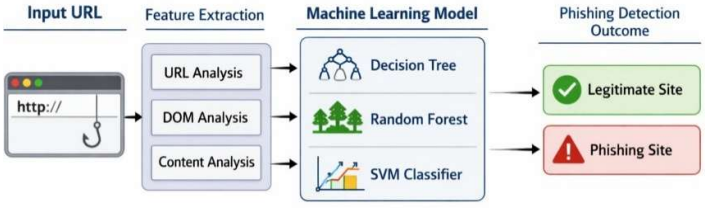

Fig. 1. Proposed PhishLock System Architecture

A. URL-Driven Feature Extraction

Now, we move on to the actual URL. We try to identify the quirks present within the URL. Phishing URLs often contain complex, entangled, and even weird URLs to trick the user. We extract the features from the actual URL. We do not even need to look this up anywhere. It's quick, simple, and perfect for real-time phishing detection. The feature vector here is:

$$x = [x_1, x_2, x_3, \ldots, x_n] \quad (1)$$

- URL Length: Long URLs are often a phishing threat, and it is because they might be a result of a redirection chain or a hidden token. The length of a URL is calculated as follows:

$$x_1 = |u| (2)$$

where u represents the input URL.

- Shannon Entropy: Phishing URLs frequently include random character strings. Shannon entropy is computed for assessing randomness. High entropy indicates un-usual and suspicious patterns in character string formation.

$$x_2 = H(u) = -\sum_{i=1}^{k} p_i \log_2 p_i \quad (3)$$

Here, pi is the probability of character i, and k is the total number of unique characters.

- Subdomain Depth: Attackers tend to create many subdomains to resemble legitimate domains. The depth of a domain can be measured by counting the number of dot separators present in the domain's URL.

$$x_3 = \text{CountDots}(u) \quad (4)$$

- Suspicious Keyword Presence: Certain keywords such as "login", "verify", or "secure" are commonly used in

1, if u contains a token in S x4 = 0, otherwise (5)
where S represents a predefined suspicious token set.

B. DOM-Based Feature Extraction

We'll look beyond the link and also examine the page content by analyzing the DOM of the page. Phishing sites often contain forms that request the user to input information, so we'll check the number of input fields requesting sensitive information.

$$x_5 = F(u) \quad (6)$$

In this equation, F (u) represents the total number of password fields.

C. Feature Normalization

To give all the features an equal chance of being selected, we normalize the features before classification. This is to prevent certain features with large ranges from dominating the classification process.Z-score normalization is calculated by:

$$x'_j = \frac{x_j - \mu_j}{\sigma_j} \quad (7)$$

where μj and σj represent the mean and standard deviation of feature j.
The normalized feature vector becomes:

$$\mathbf{x} = [x_1, x_2, \ldots, x_n] \quad (8)$$

D. Decision Tree Classification

Phishing detection is a binary classification problem. Deci-sion Tree classification works by splitting the feature space in a step-by-step manner, choosing splits that maximize impurity reduction. Decision Tree classification has high interpretability and efficiency, making it a good candidate for deployment in a browser.
The classification label is:

$$y \in \{0, 1\} \quad (9)$$

where:
0 = Legitimate, 1 = Phishing
The prediction function is:

$$\hat{y} = T(\mathbf{x}') \quad (10)$$

where T denotes the trained Decision Tree model.

- Entropy Measure: Entropy is a measure of the impurity of the node. A pure node has an entropy of zero, while a node with many classes has a high entropy.

$$H(t) = -\sum_{c \in \{0,1\}} p(c \mid t) \log_2 p(c \mid t) \quad (10)$$

- Information Gain: For each node, we choose the feature that gives us the highest information gain. In formation gain is defined by the equation below.

$$IG(t, x_j) = H(t) - \sum_v \frac{|t_v|}{|t|} H(t_v) \quad (12)$$

where tv represents the child subsets after the split.

E. Real-Time Alert Mechanism

After the classification process, the system sends an alert if the phishing activity is detected. Due to the execution of all the processes on the client side, the time remains minimal. The predicted label is:

$$\hat{y} = 1 \quad (13)$$

Detection latency is measured as:

$$\text{Latency} = t_{\text{decision}} - t_{\text{request}} \quad (14)$$

F. Performance Evaluation

To assess detection performance, standard classification metrics are used.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (15)$$

$$\text{Precision} = \frac{TP}{TP + FP} \quad (16)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (17)$$

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \quad (18)$$

IV. Output and Implementation

The output of the proposed PhishLock system is illustrated in Fig. 2.

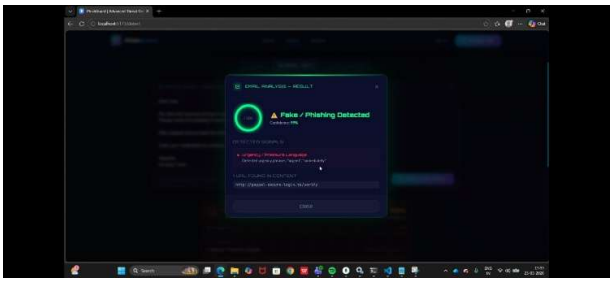

Fig. 2. Output of PhishLock Showing Real-Time Phishing Detection Alert

The above Fig. 2 shows the output interface of the proposed PhishLock system. The developed PhishLock system is a lightweight browser extension which can perform real-time phishing site detection in the client environment. For a partic-ular user who has accessed a certain webpage, the PhishLock system can automatically extract the lexical feature set and the DOM-based feature set from the URL of the accessed website and then input the feature set into the Decision Tree classifier. After the classification process is finished, the PhishLock system can immediately display the result to the user as Legitimate Site and Phishing Site with a visual warning alert.

V. Results and Discussion

This section will discuss the evaluation of the proposed sys-tem for the detection of phishing attacks. The performance of the proposed system, such as classification accuracy, precision, recall, F1, and average latency for the detection of phishing attacks, will be considered while evaluating the performance of the proposed system.The performance of the proposed system will be evaluated based on the same data set, ensuring that the performance of the proposed system is comparable to the performance of the other systems. The data set based on which the performance of the proposed system will be evaluated consists of 20,000 labeled URLs, which are collected from publicly accessible phishing data sets and legitimate domain data sources. The performance of the proposed system will be evaluated based on a balanced data set, ensuring that the performance of the proposed system is not biased to a particular data set.

A. Baseline Model Comparison

To assess the effectiveness of the proposed method, it is compared with other techniques for detecting phishing, which are commonly used. These techniques include rule-based systems, Support Vector Machines (SVM), and Random Forest. These techniques have been considered to be the baseline techniques for benchmarking.

Table I: Performance Comparison with Baseline Models

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Rule-Based	82.4	79.6	76.2	77.8
SVM	90.8	89.3	87.5	88.4
Random Forest	93.6	92.8	91.9	92.3
Proposed (Decision Tree)	96.3	95.6	94.4	95.0

As can be seen, based on the results obtained and presented in Table I, the Decision Tree model, which is presented in this paper, indicates the highest precision and F1 score among all the approaches used in the comparison. In comparison with the traditional approach, which is based on the rule-based filtering, it should be noted that the results obtained are significant. This is a clear indication of the superiority of the data-driven approach over the threshold-based approach. In comparison of the model, which is presented in this paper, with the ensemble-based Random Forest, it should be noted that the performance is better, while the complexity is reduced.

B. Detection Latency Analysis

Considering the fact that the system is a browser extension, the detection speed was an important aspect. The average classification time was calculated after loading the webpage until the final output was generated.

TABLE II: AVERAGE DETECTION LATENCY

Model	Avg. Detection Time (ms)
Rule-Based	180
SVM	320
Random Forest	290
Proposed (Decision Tree)	250

From the results, it can be seen that the model is able to sustain a reduced inference time, at the same time ensuring that the accuracy level of the detection process lies on the higher side. Although the rule-based filtering method is slightly faster, the model is not robust and does not generalize. Decision Tree is a good balance between speed and prediction and can be used for client-side deployment.

C. Confusion Matrix Evaluation

To further examine classification behavior, the confusion matrix of the proposed model is analyzed.

TABLE III: CONFUSION MATRIX FOR PROPOSED MODEL

	Predicted Legitimate	Predicted Phishing
Actual Legitimate	14,112	407
Actual Phishing	321	5,160

From the confusion matrix, it is evident that the true positive rate and true negative rate are high, i.e., the discriminative power is high. Also, from the given results, it can be concluded that the false positive rate is low, i.e., real sites are not blocked, and the false negative rate is also low, i.e., phishing sites are identified.

D. Overall Discussion

The experimental results verify the superiority of the sug-gested client-side phishing detection framework. The incor-poration of lexical features of URLs and DOM attributes enhances the capacity of the model for distinguishing between phishing patterns and genuine websites. Moreover, the em-ployment of entropy splitting with the Decision Tree assists with the effective feature selection.The suggested framework is distinct from the conventional phishing detection systems because it carries out all the calculations on the client-side browser. This enables the preservation of the privacy of the users and eliminates the need for network dependence.The accuracy of 96.3% with an average detection time of 250 ms confirms the viability of the proposed framework.

VI. CONCLUSION

The novelty of this research is based on the introduction of a lightweight client-side phishing detection tool, which was named PhishLock and relied on a real-time browser-based solution that included lexical analysis of URLs, host-based feature sets, and feature sets obtained through the extraction of the Document Object Model. This computation was carried out on the client-side, which is the browser, and this eliminated the need to rely on external resources while ensuring that user privacy is not compromised.This experimental evaluation, which is done using a dataset of 20,000 labeled websites, proves that the suggested method is capable of achieving an accuracy of 96.3% and an F1-score of 95.0%, which is better compared to the traditional method using rule-based filtering and the conventional method using classical machine learning techniques. At the same time, the suggested method is capable of achieving a latency of 250 ms, which proves the effectiveness of the suggested method in the client-side context. The effectiveness of the suggested method in pro-viding robustness against phishing attacks is proved through experimental evaluation.The suggested framework appears to be more suitable for the end-user environment, bearing in mind the privacy, computation overhead, and response time. In the future, the focus will be on the integration of the adaptive learning mechanism to cater to the dynamic nature of phishing attacks, the integration of adversarial robustness, and the extension of the suggested framework to mobile devices. In addition, the exploration of the application of the concept of explainable AI will also be beneficial to the suggested framework.

REFERENCES

[1] A. K. Jain and B. B. Gupta, “Phishing detection: Analysis of visual similarity based approaches,” Security and Communication Networks, vol. 2017, pp. 1–20, 2017.

[2] S. Garera, N. Provos, M. Chew, and A. D. Rubin, “A framework for detection and measurement of phishing attacks,” in Proc. ACM Workshop on Recurring Malcode, 2007, pp. 1–8.

[3] 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.
[4] W. Khan, A. Ahmad, A. Qamar, M. Kamran, and M. Altaf, "SpoofCatch: A client-side protection tool against phishing attacks," IT Professional, vol. 23, no. 2, pp. 65–74, 2021.
[5] M. Johns, B. Braun, M. Schrank, and J. Posegga, "Reliable protection against session fixation attacks," in Proc. ACM Symp. Applied Comput-ing, 2011, pp. 1531–1537.
[6] T. Pietraszek and C. V. Berghe, "Defending against injection attacks through context-sensitive string evaluation," in Proc. Int. Workshop on Recent Advances in Intrusion Detection, 2005, pp. 124–145.
[7] N. Provos and T. Holz, "Google's malware warning," IEEE Security & Privacy, vol. 6, no. 6, pp. 80–83, 2008.
[8] G. Xiang, J. Hong, C. P. Rose, and L. Cranor, "CANTINA+: A feature-rich machine learning framework for detecting phishing web sites," ACM Trans. Information and System Security, vol. 14, no. 2, pp. 1–28, 2011.
[9] R. Verma and N. Hossain, "Semantic feature selection for text with application to phishing email detection," in Proc. IEEE Int. Conf. Data Mining Workshops, 2014, pp. 38–45.
[10] B. B. Gupta, N. A. G. Arachchilage, and K. E. Psannis, "Defending against phishing attacks: Taxonomy of methods, current issues and future directions," Telecommunication Systems, vol. 67, no. 2, pp. 247–267, 2018.
[11] M. Aburrous, M. A. Hossain, F. Thabatah, and K. Dahal, "Intelligent phishing detection system for e-banking using fuzzy data mining," Expert Systems with Applications, vol. 37, no. 12, pp. 7913–7921, 2010.
[12] J. Ma, L. K. Saul, S. Savage, and G. M. Voelker, "Beyond blacklists: Learning to detect malicious web sites from suspicious URLs," in Proc. ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2009, pp. 1245–1254.
[13] Y. Zhang, J. I. Hong, and L. F. Cranor, "CANTINA: A content-based approach to detecting phishing web sites," in Proc. Int. World Wide Web Conf., 2007, pp. 639–648.
[14] S. Marchal, J. Francois, R. State, and T. Engel, "PhishStorm: Detecting phishing with streaming analytics," IEEE Trans. Network and Service Management, vol. 11, no. 4, pp. 458–471, 2014.