WWW.IJRPR.COM

International Journal of Research Publication and Reviews

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

Real Time Phishing URL Detection Using Machine Learning Algorithms

Tamizhselvan K

Scholar, Department of MCA, tamizhselvan22042003@gmail.com Dr. M.G.R Educational and Research Institute

A B S T R A C T:

In the current online landscape, phishing attacks present a major risk to internet users by deceiving them into revealing confidential information like usernames, passwords, and credit card details. These attacks usually include deceptive websites that look authentic, complicating users' ability to differentiate between genuine and phony sites. Conventional detection techniques, like blacklists and manual inspections, frequently fail because of the swift appearance of new phishing URLs. Consequently, there is an increasing demand for automated and smart solutions to promptly identify phishing websites. This project introduces a machine learning method for detecting phishing URLs by examining their structural and lexical features. A collection of labeled URLs (comprising both legitimate and phishing sites) was utilized for training and evaluation. Attributes like the existence of IP addresses, unusual URL structures, questionable keywords, the count of special characters, and the utilization of shortening services were gathered from the URLs. These characteristics function as signals to instruct classification algorithms on how to differentiate between phishing and authentic URLs. Three machine learning classifiers—Random Forest, LightGBM, and XGBoost—were developed and assessed according to their accuracy and performance indicators. The Random Forest model yielded impressive classification outcomes, whereas LightGBM and XGBoost demonstrated similar effectiveness with quicker processing times. To enhance user experience, the created models were incorporated into a Flask web application, allowing users to enter a URL and swiftly obtain a prediction regarding its authenticity. The suggested system shows excellent accuracy and dependability in identifying phishing URLs and serves as an important resource for both individuals and organizations in combating phishing risks. It emphasizes the capabilities of machine learning in cybersecurity and offers a scalable approach for rapid threat identification.

Keywords: Phishing Detection, Machine Learning, URL Classification, Cybersecurity, Real-Time Detection, Ensemble Learning, Flask Web Application

Introduction

As the internet rapidly evolves, digital services have changed the way we communicate, conduct business, and exchange information. Nonetheless, this expansion has been accompanied by an increase in cybersecurity risks—especially phishing attacks. Phishing tricks users into disclosing sensitive information via fraudulent websites and harmful URLs, rendering conventional detection methods, like blacklists, less effective [1]. This study introduces a machine learning-driven method for detecting phishing URLs in real time, emphasizing the structural and lexical examination of URLs without depending on external data or content evaluation.

The system identifies important attributes from URLs, such as IP address occurrence, suspicious keywords, URL length, frequency of special characters, and domain patterns, to train classification models such as Random Forest, LightGBM, and XGBoost. These models are assessed with a carefully selected dataset that includes both phishing and valid URLs, which have been refined for quality and consistency.

By employing methods such as cross-validation and hyperparameter optimization, the suggested system attains elevated accuracy while reducing incorrect predictions. The ultimate solution is implemented through a Flask-powered web interface, allowing users to obtain immediate feedback on the authenticity of URLs. This study illustrates how ensemble learning improves phishing detection, providing a scalable and adaptable resource for proactive cybersecurity protection [2][3].

Literature Review

The identification of phishing attacks has been a central topic in cybersecurity research for a significant time, resulting in the creation of multiple automated solutions. Conventional techniques depended significantly on blacklists and rule-based frameworks, which housed collections of identified harmful URLs. Although successful for attacks that have been documented previously, these approaches are fundamentally static and insufficient to address new threats or zero-day phishing websites [4].

To overcome these challenges, machine learning methods started to be utilized. Initial attempts employed lexical and host-related characteristics of URLs, using models like Decision Trees and Support Vector Machines (SVMs) to identify phishing websites [5]. Despite showing promise, these models frequently faced limitations due to skewed data distributions and changing phishing techniques.

Ensemble learning techniques, like Random Forest and Gradient Boosting, represented a notable advancement. By integrating the advantages of various classifiers, these models proved to be more robust and attained greater accuracy in detecting phishing [6]. Recent research has utilized high-efficiency gradient boosting frameworks such as XGBoost and LightGBM because of their rapidity, scalability, and proficiency in effectively managing structured data [7][8].

Another progress has been the shift towards real-time URL detection systems based on URLs. These systems rely entirely on URL architecture—analyzing aspects such as length, special symbols, subdomain numbers, and the occurrence of dubious keywords—to provide instant outcomes without depending on outside databases or content retrieval [9].

Methodology

The suggested phishing URL detection framework is established on an organized pipeline that includes data gathering, feature extraction, training of machine learning models, and real-time prediction. The approach is intended to be strong, expandable, and flexible for detecting phishing URLs with great precision.

Data Gathering: The first step includes compiling an extensive dataset that contains both authentic and phishing URLs. These samples are collected from trustworthy and publicly accessible sources like PhishTank, OpenPhish, and the top sites on Alexa [10]. To guarantee the model's effectiveness, the dataset features a balanced combination of benign and malicious URLs, minimizing bias and enhancing generalization.

Feature Extraction: After the dataset is ready, the subsequent step involves deriving a collection of distinguishing features from each URL. These characteristics are generated solely from the URL's structure and content, necessitating no engagement with the site or outside databases. Instances comprise the use of an IP address in place of a domain name, the length of URLs, the occurrence of special characters (e.g., “@”, “-”, “=”), implementation of HTTPS, count of subdomains, and the presence of potentially dubious terms such as “login”, “secure”, or “bank” [11].

Data Preprocessing: The gathered features go through preprocessing to address missing values, eliminate outliers, and standardize the data for peak performance. Categorical variables are transformed into numerical format, while continuous features are normalized to ensure uniformity among input values. This preprocessing stage guarantees that the data provided to the models is tidy, organized, and suitable for training [12].

Model Training: Three classifiers for machine learning—Random Forest, LightGBM, and XGBoost—are chosen because of their demonstrated effectiveness in classification tasks. Every model is trained with the preprocessed dataset, utilizing methods like k-fold cross-validation and grid search to enhance hyperparameters and minimize overfitting [7][8][13].

Model Assessment: The developed models are assessed using metrics such as accuracy, precision, recall, and F1-score. These measurements offer a comprehensive

Real-Time Implementation: For practical application, the finished model ensemble is incorporated into a web application built on Flask. This application offers a user interface that allows the submission of URLs for evaluation. When submitted, the system retrieves features from the entered URL, sends them to the trained models, and combines the outcomes to provide an immediate decision—genuine or phishing. This real-time functionality guarantees effectiveness in changing settings like email applications, web browsers, and corporate security frameworks [14].

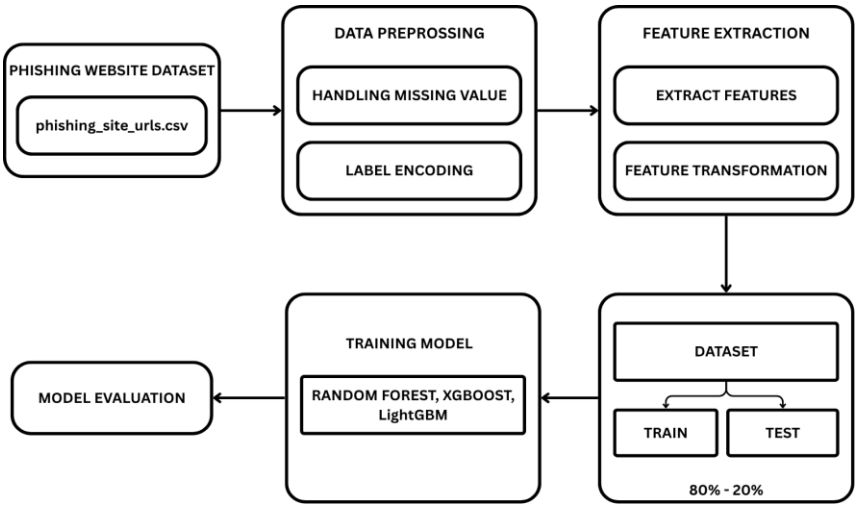

Fig. 1 – System Architecture.

Implementation

The execution of the phishing URL detection system comprises several interconnected elements, starting with feature extraction, training the model, and deploying on the web. At first, a dataset comprising both phishing and genuine URLs is imported into a Python environment through libraries like Pandas and NumPy. A thorough collection of custom features is derived from these URLs. This encompasses syntactic traits including the existence of IP addresses within the domain, the overall count of dots and hyphens, the URL and hostname lengths, the inclusion of dubious terms (e.g., "verify", "secure"), and tallies of special symbols such as '@', '?', and '%'. These characteristics are selected for their importance in distinguishing between authentic and misleading links.

After extracting features, the dataset is divided into training and testing subsets. Well-known machine learning models like Random Forest, LightGBM, and XGBoost are developed with the training data. Hyperparameter tuning utilizes grid search or randomized search to enhance the performance of each model. The models that have been trained are subsequently assessed on the test set through accuracy, precision, recall, and F1-score to guarantee dependable detection abilities.

Flask is utilized to create a simple web application for deployment. This application takes a URL provided by the user via an easy interface and sends it to the backend for processing. The backend component retrieves the required features from the URL, loads the models that have been pre-trained, and produces predictions. To improve precision, the results from the three models are combined, and the final result—indicating whether the URL is fraudulent or authentic—is relayed back to the user instantly.

The whole system is engineered to be efficient and scalable, needing few computational resources while producing swift and precise outcomes. The modular design guarantees simple upkeep and facilitates potential integration with browser add-ons or more extensive cybersecurity systems.

Results

To evaluate the performance of the phishing URL detection system, various machine learning models were developed and tested utilizing a dataset that included both phishing and legitimate URLs. The assessment was based on important classification metrics including Accuracy, Precision, Recall, and F1-Score, offering a thorough perspective on the models' effectiveness.

The classifiers evaluated comprised Random Forest, LightGBM, and XGBoost, all of which are ensemble methods recognized for their predictive power. Of these, XGBoost reached the highest overall accuracy, while LightGBM and Random Forest were not far behind. Each model showed impressive performance across all metrics, highlighting the system's capability to effectively identify phishing attempts while keeping low instances of false positives and missed detections.

Table 1 - Evaluation Summary

Metric	Random Forest	LightGBM	XGBoost
Accuracy	**90%**	**86%**	**88%**
Precision	**91%**	**87%**	**87%**
Recall	**87%**	**63%**	**67%**
F1-Score	**89%**	**73%**	**76%**

Conclusion

This research proposed and executed a machine learning method to efficiently identify phishing URLs through the examination of their structural and lexical attributes. In contrast to conventional approaches that rely on blacklists or manual checks, the suggested system utilizes automated classification methods to detect malicious URLs instantly. By utilizing comprehensive feature extraction and leveraging robust ensemble learning models like Random Forest, LightGBM, and XGBoost, the system showcased exceptional accuracy and resilience in identifying phishing threats.

The evaluation results clearly demonstrate that machine learning provides a scalable, effective, and flexible approach to address the changing nature of phishing attacks. By concentrating exclusively on URL attributes, the system reduces dependence on external services, allowing for quicker and more dependable identification. Incorporating this model into a user-friendly web application based on Flask significantly improves accessibility, rendering it suitable for real-world applications. In summary, this research represents a meaningful advancement in creating safer online browsing spaces by delivering timely and precise alerts about phishing sites.

REFERENCES

1. Anti-Phishing Working Group, "Phishing Activity Trends Report," [Online]. Available: https://apwg.org.
2. Sahoo, D., Liu, C., & Hoi, S. C. (2017). "Malicious URL Detection using Machine Learning: A Survey." arXiv preprint arXiv:1701.07179.
3. Jain, A. K., & Gupta, B. B. (2018). "Phishing Detection: Analysis of Visual Similarity Based Approaches," Security and Privacy, 1(2), e19.
4. Bergholz, A., et al. (2010). "A Real-Life Study of Phishing Emails," Proceedings of the ACM CEAS.
5. Fette, I., Sadeh, N., & Tomasic, A. (2007). "Learning to Detect Phishing Emails," WWW '07: Proceedings of the 16th International Conference on World Wide Web, pp. 649–656.
6. Abutair, H., & Belghith, A. (2017). "Intelligent phishing detection system using deep learning techniques," 2017 International Conference on Wireless Networks and Mobile Communications (WINCOM).
7. Chen, T., & Guestrin, C. (2016). "XGBoost: A Scalable Tree Boosting System," Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794.
8. Ke, G., et al. (2017). "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," NeurIPS, pp. 3146–3154.
9. Mohammad, R. M., Thabtah, F., & McCluskey, L. (2015). "Intelligent phishing detection system using associative classification mining," Information Security Journal: A Global Perspective, 24(1-3), 89–106.
10. PhishTank, OpenPhish, Alexa – Public datasets for phishing and legitimate URLs.
11. Marchal, S., et al. (2014). "PhishStorm: Detecting Phishing with Streaming Analytics," IEEE Transactions on Network and Service Management.

12. Guyon, I., & Elisseeff, A. (2003). “An Introduction to Variable and Feature Selection,” Journal of Machine Learning Research, 3, 1157–1182.
13. Pedregosa, F., et al. (2011). “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, 12, 2825–2830.
14. Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O’Reilly Media.