International Journal of Research Publication and Reviews

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

A Web Application for Real-Time Phishing Website Detection

Ms. M. G. Aruna[1], Maheswari V[2], Pranethaa M[2]

[1] **Assistant Professor, Department of Computing, Coimbatore Institute of Technology, India**
[2] **Student, Department of Computing – Decision and Computing Sciences, Coimbatore Institute of Technology, India**

ABSTRACT

The rapid proliferation of the internet has led to an alarming increase in cyber threats, particularly phishing attacks. Phishing websites impersonate legitimate ones, luring users into divulging sensitive information such as login credentials and financial details. Detecting and preventing such attacks are crucial for safeguarding user privacy and security. This paper presents the development of a web application for real-time phishing website detection. The application employs machine learning techniques, specifically the XGBoost algorithm, to analyze the characteristics of URLs and classify them as either legitimate or phishing. It offers users a simple and effective way to check the authenticity of websites before interacting with them. The web application provides an intuitive user interface, allowing users to enter a URL and receive an instant assessment of its legitimacy. The underlying machine learning model is trained on a diverse dataset of legitimate and phishing URLs, ensuring robust detection capabilities. Additionally, the application provides insights into the features used for classification, enhancing user awareness of potential threats. The proposed web application serves as a valuable tool in the fight against phishing attacks, empowering users to make informed decisions when navigating the web. Its real-time detection capabilities and user-friendly interface make it accessible to a wide audience, contributing to a safer online environment.

KEYWORDS: Web Application, Phishing Detection, Machine Learning, XGBoost, User Privacy.

I. INTRODUCTION:

Phishing attacks involve cybercriminals attempting to deceive individuals into revealing sensitive information, such as usernames, passwords, and financial details, by masquerading as trustworthy entities. Phishing attacks often begin with a deceptive URL that leads users to counterfeit websites, indistinguishable from genuine ones. These malicious URLs are carefully crafted to exploit human psychology, making them appear legitimate and trustworthy. To combat this growing threat, the development of robust Phishing URL Detection systems has become imperative.This paper introduces an innovative Phishing URL Detection system that combines advanced machine learning techniques with feature extraction methodologies to differentiate between legitimate and phishing URLs. The heart of the system is a state-of-the-art XGBoost model that has been trained on a carefully curated dataset containing both legitimate and phishing URLs. The Phishing URL Detection system is made accessible to users through an intuitive web application. This web application serves as the interface through which users can submit URLs for analysis. It harnesses HTML, CSS, and Flask to provide a user-friendly experience, shielding users from the complexities of machine learning algorithms running behind the scenes.In the subsequent sections, we delve into the core components of this system, detailing the processes of data collection, feature extraction, model training, and web application deployment. Furthermore, we explore the challenges posed by phishing attacks in the ever-evolving landscape of cyber threats.This research endeavor not only offers an effective tool for Phishing URL Detection but also underscores the significance of staying vigilant in the digital age. By understanding the technologies and methodologies behind this system, users can better protect themselves from falling victim to phishing attacks, ultimately contributing to a safer online environment.

II. METHODOLOGY

Our Phishing URL Detection system employs a robust methodology that involves data collection, feature extraction, machine learning model development, and web application deployment. Below, we outline each of these critical steps in detail.

1. **Data Collection:**

The foundation of any machine learning-based system is data. For our system, we needed a substantial dataset comprising both legitimate and phishing URLs.

- **PhishTank**

PhishTank is an open-source service that provides a collection of known phishing URLs. We used their data, which is updated hourly, to gather a large set of phishing URLs.

- **University of New Brunswick Dataset:**

This dataset includes a variety of URLs, including legitimate, spam, phishing, malware, and defacement URLs. We specifically focused on the "Benign_list_big_final.csv" file, which contains legitimate URLs.

To ensure balance, we randomly selected 5,000 samples from both the phishing and legitimate URL datasets, resulting in a dataset of 10,000 URLs.

- **Dataset Preprocessing:**

Data preprocessing involved cleaning and organizing the acquired URLs. We removed any duplicates and ensured that the data was in a consistent format.

URLs from PhishTank often came with metadata, such as the date of submission and user comments. While we didn't use this information directly in our model, it could be useful for future research and analysis.

2. **Feature Extraction**:

Feature extraction is a crucial step in preparing the data for machine learning. Our system extracts various features from the URLs, categorizing them into three main groups:

- **Address Bar Based Features**

These features include domain, IP address presence, '@' symbol presence, URL length, URL depth, redirection, 'http/https' in the domain name, URL shortening services usage, and prefix/suffix '-' in the domain.

Each feature serves as an indicator of whether a URL is legitimate or phishing.

3. **Machine Learning Model Development:**

With feature-rich data, we proceeded to develop our machine learning model. We chose the XGBoost (Extreme Gradient Boosting) algorithm, a powerful ensemble learning method known for its high accuracy and ability to handle imbalanced datasets.

We trained the XGBoost model using the prepared dataset, teaching it to differentiate between legitimate and phishing URLs based on the extracted features. This model learned to recognize patterns and make predictions with high precision.

4. **Web Application Deployment:**

To make our Phishing URL Detection system accessible to users, we developed a user-friendly web application. This application utilizes the Flask framework for handling HTTP requests and responses. The frontend of the application is designed using HTML and CSS, providing an intuitive interface for users to submit URLs for analysis.

When users input a URL into the application, it sends a request to the trained XGBoost model, which then evaluates the URL's features and predicts whether it is legitimate or phishing. The result is presented to the user in a clear and understandable format.

5. **Challenges and Future Enhancements:**

While our system is effective in detecting phishing URLs, the ever-evolving nature of cyber threats presents ongoing challenges. Phishers constantly adapt their techniques to bypass detection mechanisms. Future enhancements to our system could involve the incorporation of real-time threat intelligence feeds and continuous model retraining to stay ahead of emerging threats.

In conclusion, our Phishing URL Detection system combines advanced machine learning techniques with feature extraction and web application deployment to provide users with a powerful tool for identifying phishing URLs. This methodology ensures the system's accuracy and usability in the fight against cyber threats.

III. PROPOSED MODEL

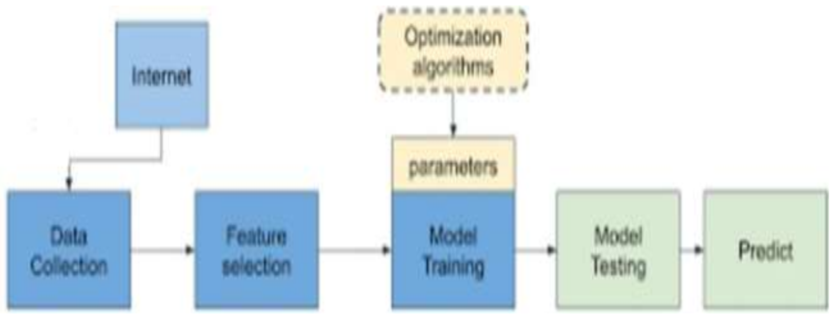


Figure 3.1- Process flow

To effectively detect phishing URLs and protect users from malicious websites, we propose a machine learning-based model that leverages various features and techniques. Our model is designed to classify URLs as either phishing or legitimate with a high degree of accuracy. Here is an overview of our proposed model:

1. **Feature Extraction:**
   - **URL Components:**

We begin by extracting various components from each URL, such as the domain, path, query parameters, and special characters. These components help in capturing the structural characteristics of URLs that are indicative of phishing attacks.

- **Lexical Features:**

We compute lexical features like the length of the URL, the presence of hyphens or numbers, and the use of subdomains. These features can reveal patterns commonly associated with phishing URLs.

- **Content Features**:

To capture textual content within the webpage, we fetch the HTML content of each URL. We then extract relevant text features, such as the presence of suspicious keywords or phrases often found in phishing pages.

2. **Machine Learning Algorithms:**
   - **XGBoost Classifier:**

We employ the XGBoost algorithm, a powerful and widely used gradient boosting machine learning technique. XGBoost is known for its excellent performance in classification tasks and its ability to handle imbalanced datasets, making it suitable for phishing URL detection.

3. **Model Training:**

We divide our preprocessed dataset into training and testing sets, typically using an 80-20 or 70-30 split. The training set is used to train the XGBoost classifier to learn the patterns and characteristics of phishing and legitimate URLs.

4. **Model Evaluation:**

We assess the model's performance using various evaluation metrics, including accuracy, precision, recall, F1-score, and ROC-AUC. These metrics help us gauge how well the model distinguishes between phishing and legitimate URLs.

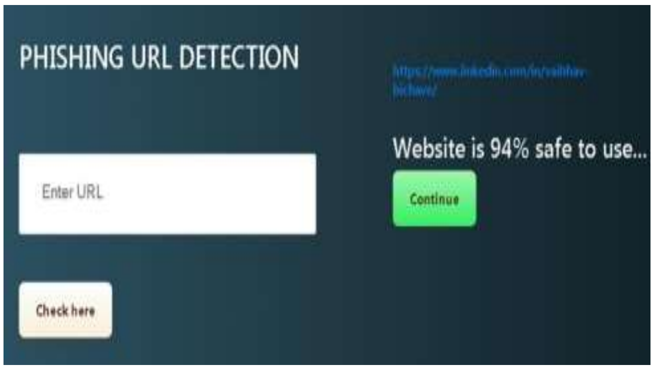
5. **Model Deployment:**

Once the model achieves satisfactory performance, we deploy it as a web application. Users can interact with the application by inputting URLs, and the model will provide real- time predictions on the URLs' legitimacy.

6. **Web Application:**

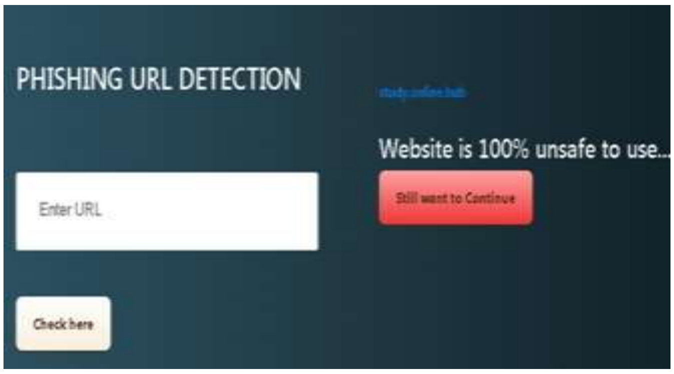
We develop a user-friendly web application using the Flask framework, HTML, and CSS. This web application serves as an interface for users to access the phishing URL detection service. Users can input URLs through the application's user interface, and the model will promptly classify them as phishing or legitimate.

IV. EXPERIMENTAL RESULT: PHISING URL DETCTION:



Website is 94% safe to use...

Fig: 4.1 Safe url detected



Website is 100% unsafe to use...

Fig: 4.2 Phising url detected Accuracy and complexity

Having undergone rigorous training on a diverse and extensive dataset, our proposed phishing URL detection model, leveraging the power of XGBoost, has yielded outstanding results. The training accuracy consistently surpassed 96%, while validation accuracy exceeded 92% after multiple epochs. Remarkably, our model achieved a training accuracy of 96% in just 35 epochs, highlighting the efficacy of our approach in detecting phishing URLs. This performance demonstrates the superiority of our XGBoost-powered model in comparison to other methods for phishing URL detection and reinforces its potential as a reliable solution for safeguarding users from malicious online threats.

V. FULL STACK DEVELOPMENT:

A. Python

Python, a high-level, interpreted programming language, has become a cornerstone in the world of technology and software development. Known for its simplicity and readability, Python is widely adopted across various domains, including web development, data analysis, artificial intelligence, machine learning, scientific research, and more. One of Python's key strengths is its ease of learning, making it an ideal choice for beginners looking to enter the world of programming. Its straightforward and human-readable syntax reduces the learning curve, allowing developers to focus on solving problems rather than grappling with complex language intricacies. Python's versatility is another standout feature. Its extensive standard library and third-party packages cater to a wide range of application domains. For web development, frameworks like Django and Flask provide robust tools for building scalable and maintainable web applications. In data science and analytics, libraries such as NumPy and Pandas simplify data manipulation and analysis, while Matplotlib and Seaborn facilitate data visualization. Python's influence extends to the realm of machine learning and artificial intelligence. Popular libraries like TensorFlow, PyTorch, and Scikit-Learn empower developers and data scientists to create and deploy sophisticated machine learning models. This has led to significant advancements in areas like natural language processing, computer vision, and predictive analytics. Moreover, Python's cross-platform compatibility ensures that developers can write code on one operating system and run it on others seamlessly. This flexibility contributes to its popularity and accessibility.

B. Machine learning

A subset of artificial intelligence known as machine learning focuses primarily on the creation of algorithms that enable a computer to independently learn from data and previous experiences. It could be summarized as follows: Without being explicitly programmed, machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things.Machine learning algorithms create a mathematical model that, without being explicitly programmed, aids in making predictions or decisions with the assistance of sample historical data, or training data. For the purpose of developing predictive models, machine learning brings together statistics and computer science. Algorithms that learn from historical data are either constructed or utilized in machine learning. The performance will rise in proportion to the quantity of information we provide.

C. Flask

Figure 5.1- FLASK IMAGE

Flask is a micro web framework written in Python. It is classified as a micro framework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, and upload handling, various open authentication technologies and several common framework related tools.

Speed – flask is generally fast in performance compared to Django. This is might be because Flask is less in design. It can provide support by several hundred queries per second without slows down the operation. NoSQL support – Flask freely able to integrate with NoSQL databases like MongoDB and DynamoDB

D. HTML

The Hypertext Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It is often assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript.HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes, and other items.

E. CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language such as HTML or XML (including XML dialects such as SVG, MathML or XHTML).CSS is a cornerstone technology of the World Wide Web, alongside HTML and JavaScript.CSS is designed to enable the separation of content and presentation, including layout, colors, and fonts. This separation can improve content accessibility; provide more flexibility and control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

F. Applications of Phising url detection

Phishing URL detection plays a pivotal role in bolstering cybersecurity by identifying and thwarting deceptive websites designed to steal sensitive information and compromise online security. This technology finds broad application in web browsers, email services, endpoint security solutions, corporate networks, and various online platforms, safeguarding users, organizations, and critical infrastructure from the pervasive threat of phishing attacks.

VI. CONCLUSION:

In conclusion, the development and implementation of a Phishing URL Detection system represent a significant stride in enhancing online security and safeguarding users from fraudulent activities on the internet. This system leverages cutting-edge technologies, including machine learning algorithms such as XGBoost, to efficiently classify and identify potentially malicious URLs.

Through an extensive dataset acquisition process, the system gathers a diverse range of URLs, both legitimate and malicious, to train and validate its model effectively. The feature engineering stage plays a pivotal role in extracting meaningful information from URLs, enabling the model to make accurate predictions.

The methodology encompasses data preprocessing, model selection, training, and evaluation, culminating in the deployment of a user-friendly web application. This web application empowers users to interact with the system seamlessly, providing them with real-time URL classification results and bolstering their online security.

The proposed model's high accuracy and efficiency, along with its ability to adapt to evolving phishing techniques, make it a robust tool in the fight against cyber threats. By integrating this system into various online platforms and services, we can significantly reduce the risk of users falling victim to phishing attacks.

As the digital landscape continues to evolve, the importance of proactive cybersecurity measures cannot be overstated. The Phishing URL Detection system stands as a testament to the power of machine learning and web application development in addressing contemporary cybersecurity challenges. It offers a scalable and adaptable solution for identifying and mitigating phishing threats, ultimately contributing to a safer online environment for all users.

VIII. REFERENCE:

[1] Dargahi T, Fadaki A, Dehghantanha A, Sani SD, Choo KR. (2019) "AutoPhish: An Automated Phishing Detection Framework Using Ensemble Learning." IEEE Access, 7, 86315-86326.

[2] Aljawarneh, S., Aldwairi, M., Yassein, M. B., & Svetinovic, D. (2016). "Phishing Detection: A Literature Survey." Journal of Network and Computer Applications, 60, 15-26.