DETECTION OF PHISHING IN E-BANKING WEBSITES THROUGH ASSOCIATIVE CLASSIFICATION AND DATA MINING ALGORITHMS

Project Reference No.: *499_BE_3117*

College : Sai Vidya Institute of Technology, Bengaluru
Branch : Department of Information Science and Engineering
Guide(s) : Prof. Shivaram A M
Student(s) : Ms. Kathyaini E
Ms. Vaishnavi Srinivasa Murthy
Mr. Vasanth Kumar B
Ms. Harshini S

Keywords:
Phishing Detection, Cybersecurity, Associative Classification, Data Mining Algorithms, Feature Extraction, Fraud Detection.

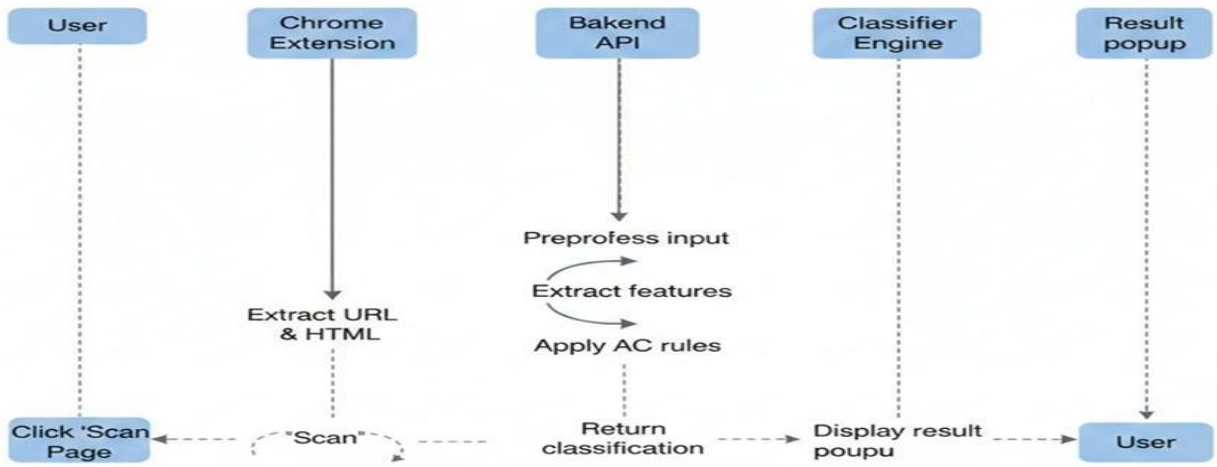
Introduction:

1. With the rapid digitization of financial services, e-banking has become a vital component of modern banking systems. However, this shift has also made users more vulnerable to phishing attacks, where malicious actors mimic genuine banking websites to steal sensitive information such as usernames, passwords, and credit card details.
2. This project, titled “Detecting Phishing in E-Banking Websites through Associative Classification and Data Mining Algorithms,” aims to develop an intelligent system capable of identifying phishing websites with high accuracy and speed. The proposed system leverages associative classification—a technique that combines association rule mining with classification—to uncover hidden patterns in data that distinguish phishing websites from legitimate ones. Alongside this, various data mining algorithms are applied to extract and analyze key features from URLs, such as the use of special characters, domain age, IP-based URLs, and redirection behavior. These features help in training the model to detect suspicious traits commonly associated with phishing.
3. A user-friendly interface allows end-users to enter website URLs and receive instant feedback on whether the site is phishing or legitimate. Meanwhile, an admin interface supports the uploading of training datasets and system updates. The project not only focuses on accuracy and efficiency but also emphasizes interpretability of results, allowing users to understand why a site was flagged as suspicious.

Objectives:

1. Building a functional Chrome extension capable of URL and HTML extraction.
2. Implementing secure communication between client (browser) and server using HTTPS.
3. Developing backend services for preprocessing, feature extraction, and phishing classification.
4. Integrating a classifier engine (rule-based / ML-based).
5. Designing a user-friendly frontend experience (popup result display).
6. Ensuring system scalability, modularity, and security.

Methodology:



1. Client-Side Data Acquisition (The Frontend Component):
    - This phase focuses on the development and deployment of the user-facing component responsible for collecting the necessary data for detection.
    - Component Development: Develop the Chrome Extension (using JS/HTML/CSS). This extension serves as the primary data collection artifact.
    - Data Extraction Logic: Implement the Feature Extraction logic. This involves observing the page after the user initiates a scan and extracting crucial attributes like the URL and HTML content.
    - Local Checks (Optional): Implement basic Local Heuristics for quick, preliminary checks within the browser before sending data to the server.
2. Real-Time Server Processing and Persistence (The Backend API)
    - This phase covers the development of the cloud infrastructure and the logic for handling immediate requests.
    - API Service Implementation: Develop the Detection API Service (e.g., using Python) responsible for receiving client requests. This service must be implemented to handle the HTTPS (TLS/SSL) protocol for secure communication.
    - Initial Validation Logic: Before engaging the complex classifier, implement the Input Validation/Blacklist Check logic. This includes checks on domain age and certificate features to provide a rapid classification for obvious cases.

3. ML Model Application and Classification
    - This phase details the specific steps for classifying the threat level using the machine learning model.
    - Request Handling Sequence: Define the operational sequence triggered when the API receives the data: 1. Preprocess input.
      2. Extract features
    - Model Execution: Implement the Classifier Engine to Apply AC rules (Associative Classification rules) to the extracted features. This is the core 6 step that determines the final classification.

Result and Conclusion

In conclusion,

- Developed a fully functional Chrome extension for real-time phishing detection.
- Enabled automatic extraction of URL and HTML content from active webpages.
- Built a secure backend API to process webpage data and return classification results.
- Implemented a machine learning–based phishing classifier trained on real phishing datasets.
- Created a feature extraction pipeline for URL features, HTML/DOM features, and content-based indicators.
- Integrated rule-based (AC) detection for identifying common phishing patterns.
- Delivered instant classification results to users via a popup interface.
- Improved user safety by warning about suspicious or malicious websites.
- Increased cybersecurity awareness about phishing attacks.
- Contributed to research by demonstrating real-time, ML-driven phishing detection.

References:

- Anti-Phishing Working Group, “Phishing Activity Trends Reports.” [Online]. Available: https://apwg.org/trendsreports/
- Google, “Google Transparency Report – Safe Browsing.” [Online]. Available: https://transparencyreport.google.com/safe-browsing
- PhishTank, “PhishTank – Phishing URL Database.” [Online]. Available: https://www.phishtank.com/
- OpenPhish, “OpenPhish – Phishing Intelligence Feed.” [Online]. Available: https://openphish.com/
- Kaspersky, “Phishing and Scam Protection Resources.” [Online]. Available: https://www.kaspersky.com/resource-center/threats/phishing

Future Scope:

The future scope of this project includes:

1. **Integration with Advanced Machine Learning Models**
   Future work can combine associative classification with advanced algorithms like deep learning, Random Forest, and SVM to improve accuracy and handle complex phishing patterns.

2. **Real-Time Detection Systems**
   The model can be developed into a real-time system integrated with web browsers or banking apps to instantly warn users about phishing websites.
3. **Enhanced Feature Engineering and Datasets**
   Expanding datasets with recent phishing data and extracting advanced features (URL patterns, domain age, SSL details) can improve robustness and detection performance.
4. **Use of Natural Language Processing (NLP)**
   NLP techniques can be applied to analyze website content and detect deceptive language commonly used in phishing attacks.
5. **Mobile and Cross-Platform Security**
   Future implementations can extend protection to mobile banking applications and multiple platforms, ensuring wider user coverage.
6. **Explainable AI (XAI) for Transparency**
   Incorporating explainable AI can help users and analysts understand why a website is classified as phishing, increasing trust and usability.