**SYNOPSIS**

**Report on**

**<<Phishing website detection using Machine learning>>**

**by**

<<Yukta Pal>> <<2200290140189>>

**Session:2023-2024 (IV Semester)**

Under the supervision of

**Dr. Akash rajak**

**KIET Group of Institutions, Delhi-NCR, Ghaziabad**



**Department Of Computer Applications**

**KIET GROUP OF INSTITUTIONS, DELHI-****NCR, GHAZIABAD-201206**

(2023-2024)

**ABSTRACT**

Phishing attacks pose a significant threat to cybersecurity, with malicious actors attempting to deceive individuals into revealing sensitive information through fraudulent websites. Traditional approaches to detecting phishing websites often struggle to keep pace with evolving attack techniques. In this context, machine learning (ML) methods offer a promising avenue for enhancing detection capabilities. This thesis explores the application of ML in the context of phishing website detection. It begins with a comprehensive review of existing literature, highlighting various methodologies and their limitations. Subsequently, the research delves into the selection of appropriate ML techniques and datasets tailored to the problem statement. Challenges such as dataset acquisition and feature extraction are addressed, paving the way for the development of a robust detection system. The core of the thesis lies in the implementation phase, where ML algorithms are applied to the selected dataset. Various classifiers are evaluated, and the effectiveness of each approach is assessed through rigorous testing. The results are compared against expected outcomes, providing insights into the performance and reliability of the detection system. Finally, the thesis concludes with a discussion on the implications of the findings and potential avenues for future research. By leveraging ML techniques, this work aims to contribute to the ongoing efforts in combating phishing attacks and bolstering cybersecurity defenses .

**TABLE OF CONTENTS**

|  |  |
| --- | --- |
| **Topic** | **Page No** |
| Introduction | 4-5 |
| Objective | 5-6 |
| Modules | 6-7 |
| Hardware and Software requirement | 7-8 |
| Functional and Non-Functional Requirement | 8-11 |
| Project Outcome | 11-12 |

**INTRODUCTION**

The prevalence of phishing attacks in today's digital landscape poses a significant threat to individuals and organizations alike. Despite the existence of anti-phishing tools and security measures, such as Google Safe Browsing (GSB) and antivirus software like Quick Heal and Avast, phishing remains a persistent problem. These tools, while helpful, have limitations in accurately detecting phishing websites and may not always provide comprehensive protection.

Motivation: The shortcomings of existing anti-phishing tools highlight the need for more advanced and effective detection methods. While tools like GSB rely on blacklists and heuristic analysis, they may fail to detect evolving phishing tactics. Additionally, the reliance on standalone tools may lead to a false sense of security among users, highlighting the importance of awareness and education about phishing threats.

Problem Statement: Given the limitations of current anti-phishing measures, the project aims to address the need for a more accurate and efficient phishing website detection system. The primary objectives are to develop a system capable of accurately classifying websites as either legitimate or phishing and to minimize the time and cost associated with detection.

**Objective:**

The primary objective of this project is to develop a robust phishing website detection system utilizing machine learning (ML) techniques. The project aims to address the growing threat of phishing attacks, which continue to pose significant risks to individuals and organizations worldwide.

In detail, the objectives of the project include:

**Detection Accuracy**: Design and implement ML algorithms capable of accurately distinguishing between legitimate websites and phishing websites. The system should achieve high precision and recall rates to minimize false positives and false negatives, respectively, thus ensuring reliable detection of phishing attempts.

**Adaptability and Generalization**: Develop ML models that can adapt to evolving phishing techniques and generalize well to unseen phishing scenarios. The system should be capable of learning from new data and updating its detection capabilities to stay ahead of emerging threats.

**Real-Time Detection**: Implement real-time phishing detection capabilities to analyze URLs and web content instantaneously. The system should provide immediate feedback on the legitimacy of websites accessed by users, allowing for timely intervention in case of phishing attempts.

**Scalability**: Design the system to be scalable, capable of handling large volumes of URL requests and processing them efficiently. This ensures that the detection system can accommodate increasing demands and effectively mitigate phishing threats across diverse user populations.

**User-Friendly Interface**: Develop an intuitive user interface that allows users to easily interact with the detection system. The interface should enable users to input URLs for analysis, view detection results, and access additional features such as reporting and logging of phishing attempts.

**Performance Optimization**: Optimize the performance of ML algorithms and the overall system to minimize computational resources and response times. This includes optimizing feature extraction, model training, and inference processes to ensure efficient operation of the detection system.

**SYSTEM REQUIREMENT SPECIFICATION**

# Hardware Requirements:

* + - Processor CPU - Intel Pentium Dual Core and Higher
    - Hard Disk capacity - 512MB Space required minimum
    - RAM - 4GB minimum

# Software requirements

* + - Programming language - Python
    - Operating system - Windows 8.1 or above
    - IDE - Anaconda , Python version 3.x

# Supporting Python modules

Python has an approach to place definitions in a document and use them in a content or in an intuitive case of the interpreter. Such a file is known as a module; definitions from a module can be brought into different modules or into the fundamental module. Some of the modules used in the project are as shown in Table 3.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Python Modules** | | | **Description** |
| 1 | I.P address | | | IP address gives the capacities to generate, control and work on IPv4 and IPv6 ad-  Dresses and networks. |
| 2 | Re | | | This module gives regular expression matching activities like those found in Perl. |
| 3 | urllib.request | | | The urllib request module characterizes functions and classes which help in opening URLs (for the most part  HTTP) in a complex world. |
| 4 | Beautiful Soup | | | Beautiful Soup is a pack- age in python for parsing HTML and XML records. It makes a parse tree for parsed pages that can be utilized to extricate information from HTML, which  is valuable for web scraping. |
| 5 | Socket | | | The BSD interface of socket is given access by this module |
| 6 | | | Requests | The HTTP requests are allowed to send by this module making use of Python. | | |
| 7 | | | Whois | WHOIS is an inquiry and response convention that is comprehensively used for addressing databases that store the selected customers or trustees of an Internet re- source. for example, a do- main name, an autonomous framework or an IP address block , also simultaneously used for broad extend of information. | | |

Table 3.1: Supporting python modules.

26

# Other Non-Functional Requirements

A non-functional requirement is a determination that depicts the framework’s activity abilities and requirements that improve its usefulness.

Some of them are as follows:

* + - Reusability: the same code with limited changes can be used for detecting phishing attacks variants like smishing, vishing, etc.
    - Maintainability: The implementation is very basic and includes print statements that makes it easy to debug.
    - Usability: The software used is very user2 friendly and open source.

**Project Outcome**

The project successfully addressed the growing threat of phishing in today's rapidly evolving technological landscape by developing a robust phishing website detection mechanism using machine learning (ML) techniques. The outcome of the project encompasses several key achievements and contributions:

1. **Efficient Phishing Detection**: The developed detection mechanism leverages ML algorithms to efficiently identify and classify phishing websites from legitimate ones. Through the use of advanced ML models, the system can analyze website features and patterns, enabling accurate detection of phishing attempts in real-time.
2. **Cost-Effective Solution**: The project outcome provides a cost-effective solution for phishing detection, as it utilizes open-source tools and libraries such as Anaconda IDE and Python programming language. This approach ensures accessibility and affordability, making the detection mechanism accessible to a wide range of users and organizations.
3. **Performance Evaluation**: A comprehensive performance evaluation of four ML classifiers - K-Nearest Neighbor, Kernel Support Vector Machine, Decision Tree, and Random Forest Classifier - was conducted. The evaluation revealed promising results, with the Random Forest Classifier achieving the highest accuracy score of 96.82%. This underscores the effectiveness of the proposed detection mechanism in accurately distinguishing between phishing and legitimate websites.
4. **Real-Time Deployment**: The developed detection mechanism is capable of real-time deployment, enabling proactive identification and mitigation of phishing threats as users interact with websites. This real-time capability enhances cybersecurity measures and helps prevent users from falling victim to phishing attacks.
5. **Scalability and Adaptability**: The project outcome is designed with scalability and adaptability in mind, allowing for seamless integration with existing cybersecurity infrastructure and compatibility with various operating systems and web browsers. This ensures widespread adoption and usability across diverse environments and platforms.