High-Level Design(HLD): Phishing Domain Detection

---

1. Overview

Phishing is a prevalent cyber threat where attackers masquerade as legitimate entities to deceive users into revealing sensitive data. This project uses machine learning to develop a system that automatically detects phishing URLs before users interact with them.

This HLD outlines the problem, proposed solution, system architecture, technical requirements, and recommendations for implementation.

---

2. Goals of the HLD

1. **Define the Problem Statement**: Highlight the challenges posed by phishing and its effects on organizations and individuals.
2. **Propose a Solution**: Present a machine learning-based approach to detecting phishing domains.
3. **Technical Framework**: Define the architecture, technologies, and tools required for implementation.
4. **Provide Strategic Insights**: Offer recommendations for security and system scalability.

---

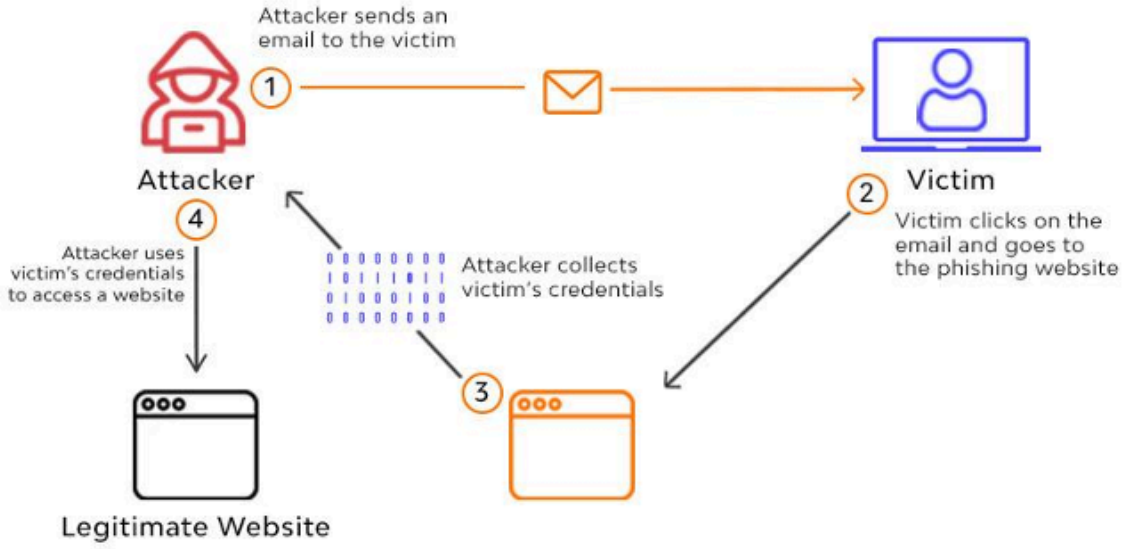
3. Problem Statement

Phishing Attacks

- **Definition**: Phishing involves deceptive practices, such as sending fraudulent emails or messages, to trick victims into providing sensitive information or clicking on malicious links.
- **Methods of Attack**:
    - Email-based phishing (96% of attacks).
    - Social engineering through instant messages or websites.
    - Spear phishing targeted at specific individuals or organizations.

Impact

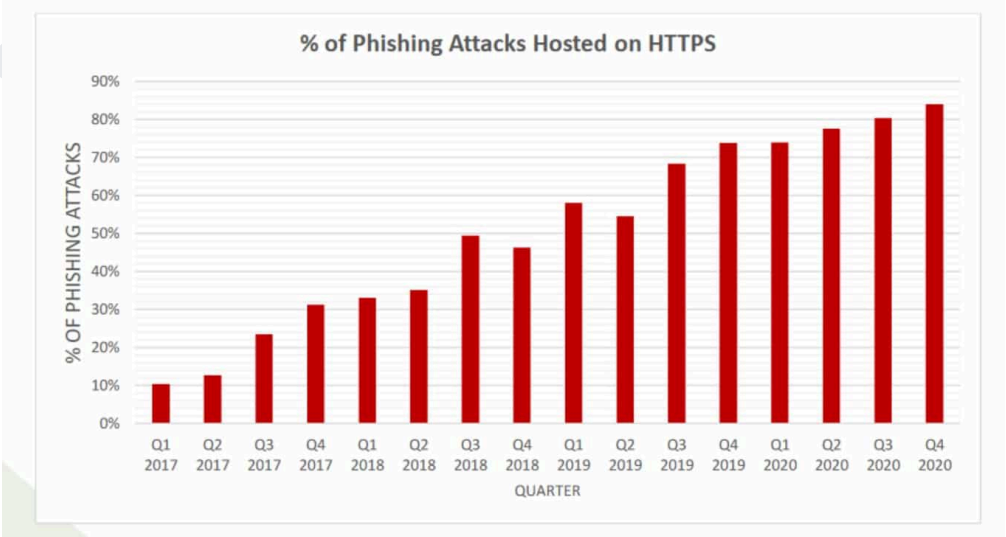
- **Financial Losses**:
    - Average cost of a phishing attack is $4.65 million per breach.



    - Organizations may experience a 5% drop in stock prices after a breach.
- **Operational Damage**:
    - Downtime and reduced productivity.
    - Loss of intellectual property and customer trust.
- **Increased Threat with Remote Work**:
    - Employees working remotely face a 37% higher risk of phishing attacks.

Statistics

- 31% of employees click on phishing links, and 67.5% provide credentials.
- Phishing websites increased significantly between 2016 and 2023 (Google Safe Browsing).

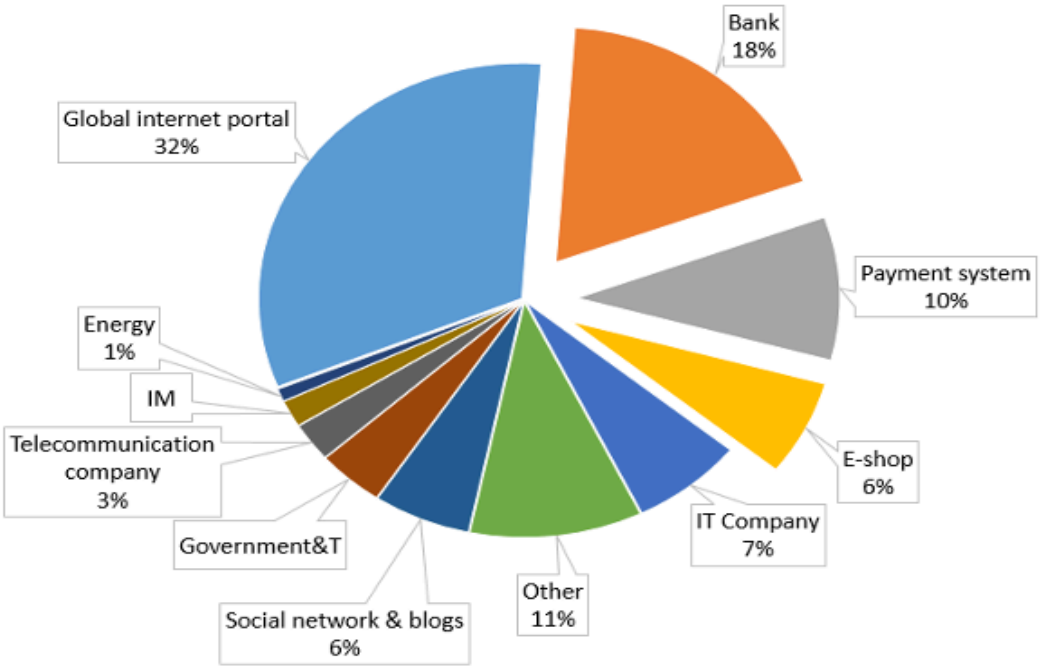


4. Proposed Solution

The project's solution leverages machine learning to identify malicious URLs. This automated system will classify URLs into two categories: *phishing* and *legitimate*.

Key Objectives

1. **Prevent User Interaction with Malicious Links**:
    - Analyze URLs before users engage with them.
2. **Automate Detection**:
    - Reduce dependency on manual identification.
3. **Scalable Integration**:
    - Enable deployment in browsers, email filters, and enterprise security systems.



5. System Architecture

5.1 Architecture Components

1. **Frontend (Web Interface)**

    - A Simple HTML, CSS and JavaScript -based user interface for inputting URLs and displaying results.

    - Features: Input field, Submit button, and Result display with probability scores.
2. **Backend (REST API)**
    - Built with FastAPI or Flask to connect the frontend with the ML model.
    - Processes user inputs and communicates with the model for predictions.
3. **Machine Learning Model**
    - A binary classifier trained to predict phishing URLs based on extracted features.
    - Models used: Random Forest, Neural Networks, and Gradient Boosting.
4. **Database (Optional)**
    - Stores previously analyzed URLs and their classifications for quick retrieval.
5. **Deployment Infrastructure**
    - Hosted on cloud platforms like Heroku or AWS for scalability and reliability.

5.2 Data Flow

1. User inputs a URL through the web interface.
2. The frontend sends a request to the REST API.
3. The backend validates the URL and extracts features.
4. The ML model predicts whether the URL is phishing or legitimate.
5. The result is returned to the frontend for user display.

5.3 High-Level Architecture Diagram

Include a diagram illustrating:

- User → Frontend → REST API → ML Model → Database (Optional) → Response → User.

---

6. Technical Requirements

6.1 Software Requirements

- **Backend Framework**: FastAPI or Flask (Python).
- **Frontend Framework**: ReactJS, CSS.
- **Machine Learning Libraries**:

    - Scikit-learn, TensorFlow, Pandas, NumPy.
- **Visualization Tools**: Matplotlib, Seaborn, Plotly.
- **Database**: SQLite, MongoDB (Optional).

6.2 Hardware Requirements

- **Server Specifications**:
    - CPU: Quad-core, 2.5 GHz.
    - RAM: 8 GB or higher.
    - Storage: 50 GB SSD.

6.3 Deployment Requirements

- **Platform**: Heroku, AWS, or Google Cloud.
- **Security**:
    - SSL/TLS for secure communication.
    - API authentication for backend access.

---

7. Security Considerations

1. **HTTPS Enforcement**: Secure data communication between the user and server.
2. **Input Validation**: Prevent injection attacks by sanitizing user inputs.
3. **Rate Limiting**: Mitigate DDoS attacks on the REST API.
4. **Error Logging**: Log errors for debugging without exposing sensitive information.
5. **Employee Training**: Complement the solution with cybersecurity training for users.

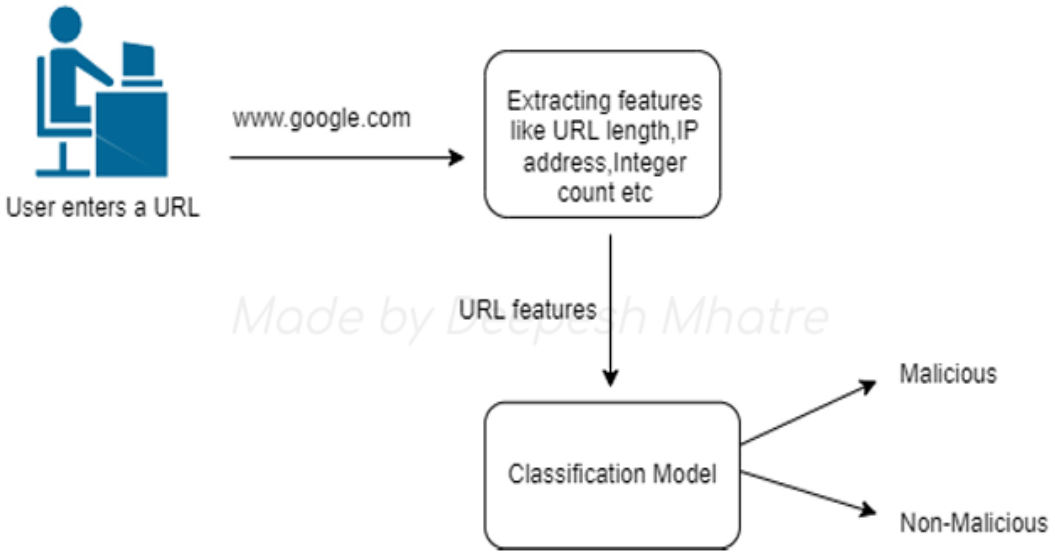
8. Data Requirements

Dataset Characteristics:

- Balanced representation of phishing and legitimate URLs.
- Diversity in features to improve model robustness.
- Real-world data with minimal artificially generated samples.

Sources:

1. UNB URL 2016 Dataset.
2. ScienceDirect Phishing Websites Dataset.



9. Future Enhancements

1. **Real-Time Integration**:
    - Embed the solution in web browsers and email filters for real-time protection.
2. **Mobile Compatibility**:
    - Develop mobile apps to detect phishing links on-the-go.
3. **Advanced Features**:
    - Incorporate WHOIS lookups and Natural Language Processing (NLP) to analyze webpage content.

4. **User Authentication**:
    - Add login functionality for personalized results and analytics.

---

10. Conclusion

The Phishing Domain Detection project combines machine learning and user-friendly interfaces to tackle phishing attacks effectively. By automating URL analysis, the system reduces dependency on human judgment, providing a scalable and robust solution for organizations and individual users.