Browser Extension To Detect Phishing Links Through a Backend System

This project proposes the development of a phishing detection and prevention system integrating a centralized phishing tracking database with a real-time browser extension. The system will allow administrators to manage and verify reported phishing links while utilizing multi-engine verification for accuracy. Implemented with Golang, HTML, CSS, and Bootstrap, this solution aims to enhance online security by proactively detecting and blocking malicious websites.

Ken Lumumba
Group A
3/21/2025

Proposal for a Phishing Detection and Prevention System

1. Introduction

Phishing attacks are among the most common cybersecurity threats, leading to data breaches, financial losses, and identity theft. This project aims to develop a **comprehensive phishing detection and prevention system** that integrates a **browser extension** with a **centralized phishing tracking system**. The system will allow administrators to manage reported phishing links, analyze threat levels, and update a database that is utilized by the browser extension for real-time protection.

2. Project Objectives

The primary objectives of this project are:

1. Develop a **web-based phishing tracking system** with administrative controls to manage phishing links.
2. Create a **browser extension** that queries the phishing database and blocks malicious websites.
3. Implement **multi-engine verification**, allowing links to be marked malicious based on detection from other engines (e.g., VirusTotal, PhishTank).
4. Provide an **administrator panel** to review, approve, or remove reported phishing links.
5. Ensure a **user-friendly interface** using **HTML, CSS, and Bootstrap**.
6. Use **Golang for the back-end** to ensure performance and scalability.

3. System Architecture

The system will consist of the following key components:

3.1 Phishing Tracking System (Backend & Web App)

- **Backend (Golang)**: A RESTful API to handle link submission, storage, verification, and retrieval.
- **Database**: Stores reported phishing links, metadata (source, date, status), and detection history.
- **Multi-engine verification**: Integrates with external services to cross-check the malicious nature of links.
- **Admin Panel (HTML, CSS, Bootstrap)**:
    - Admin account creation and authentication.
    - Dashboard to review and manage reported links.
    - Ability to delete or mark links as safe/malicious.
    - Statistics and visualization of phishing trends.

3.2 Browser Extension

- **Real-time phishing protection** by querying the phishing database.
- **Automated detection** of suspicious JavaScript behavior (e.g., redirects, keyloggers).
- **User warning system** with detailed explanations when a site is blocked.

- **User-controlled blocklist and whitelist** features.

3.3 Communication Flow

1. A user reports a phishing link via the web app.
2. The link is added to the database and reviewed by an admin.
3. If confirmed, it is marked as malicious and distributed to the browser extension.
4. The browser extension queries the API in real time and blocks flagged sites.
5. If a site is detected as suspicious by other engines, its malicious score increases.

4. Implementation Plan

The project will be implemented in the following phases:

Phase 1: System Design & Setup (Weeks 1-2)

- Define database schema and API endpoints.
- Design user interface mockups.
- Set up Golang backend and Bootstrap-based frontend.

Phase 2: Backend & Database Development (Weeks 3-4)

- Develop API for phishing link management.
- Integrate multi-engine verification.
- Implement authentication and admin roles.

Phase 3: Frontend & Admin Panel (Weeks 5-6)

- Develop web-based admin panel.
- Implement user authentication and link review workflows.
- Add dashboard analytics for phishing trends.

Phase 4: Browser Extension Development (Weeks 7-8)

- Implement phishing link querying and blocking.
- Add behavioral analysis for phishing detection.
- Allow user customization of blocklists.

Phase 5: Testing & Deployment (Week 9)

- Perform security and performance testing.
- Deploy backend and frontend.

5. Expected Outcome

By the end of this project, the system will:

- Provide a **centralized phishing detection platform** that updates in real-time.
- Offer a **browser extension** that actively protects users from phishing threats.
- Allow administrators to **track, analyze, and manage** phishing links efficiently.
- Improve **user security awareness** by educating users on why sites are blocked.

6. Conclusion

This project will contribute significantly to cybersecurity by reducing phishing attack success rates. By combining **automated phishing detection, administrator oversight, and real-time browser protection**, the system will provide an effective, scalable, and easy-to-use solution for preventing phishing threats.