iNeuron

Phishing Domain Detection System

1. Introduction

This document outlines the high-level design for a Phishing Domain Detection System. The system aims to leverage machine learning to distinguish between real and malicious domains.

2. System Requirements

- **Functional Requirements:**
  - Predict whether a given domain is real or fake based on various features using Scikit-learn.
  - Allows for model testing through an API or user interface.
- **Non-Functional Requirements:**
  - Code: Modular, Safe, Testable, Maintainable, Portable
  - Database: Cassandra database
  - Cloud: Azure
  - Logging: Implemented with Python logging library
  - Deployment: cloud, Azure

3. System Architecture

The system will consist of the following components:

- **Data Acquisition:** Downloads and preprocesses the provided phishing domain dataset.
- **Data Preprocessing:** Cleans, transforms, and prepares the data for feature engineering.
- **Feature Engineering:** Extracts relevant features from URLs, domains, pages, and content.
    - URL-Based Features (e.g., length, presence of special characters)
    - Domain-Based Features (e.g., age, registration information)
    - Page-Based Features (e.g., presence of security certificates, visual elements)
- **Model Building:** Trained various machine learning models to classify domains (e.g., Random Forest, Support Vector Machine).
- **Model Evaluation:** Evaluates model performance using metrics like accuracy, precision, and recall.
- **Model Selection:** Selected the best performing model for deployment.
- **API/UI:** Provides an interface for users to test the model with new domains.
- **Deployment:** Deploys the chosen model to Azure.

4. Technology Stack

- Programming Language: Python
- Machine Learning Libraries: Scikit-learn
- Database: Cassandra
- Cloud Platform: Azure
- Logging Library: Python logging
- Framework: **Flask**, **Flask-RESTful**

5. Data Flow

1. Data is downloaded from the provided source.
2. Data is preprocessed and cleaned.
3. Features are engineered from various aspects of the domains.
4. Multiple machine learning models are trained on the prepared data.

5. Model performance is evaluated.
6. The best performing model is selected and deployed.
7. Users interact with the system through an API or UI for url testing.
8. Model predictions are returned to the user.

6. Success Criteria

- The system accurately predicts the legitimacy of domains.
- The system is modular, maintainable, and scalable.
- Deployment is on Azure.
- The system provides a user-friendly interface.

7. Conclusion

This HLD provides a high-level overview of the Phishing Domain Detection System. The system leverages machine learning to combat phishing attacks by identifying malicious domains. Following this design, the system will be developed and deployed to improve online security.