High Level Design (HLD)

Phishing Domain Detection Project

Last date of revision: 07/05/2023

Document Version Control

Date Issued	Version	Description	Author
07/05/2023	1	Initial HLD - VI .0	Karan Singh

Contents

Abstract

Phishing attacks can seriously harm both people and organizations, which makes them a key cybersecurity risk. By identifying and blocking phishing domains, the Phishing Domain Detection project seeks to create a system that can recognize and stop phishing attempts. In order to recognize common patterns and characteristics of phishing domains, the system will employ machine learning techniques, and it will then deny access to any such domains that are found. In this project, the system architecture is presented at a high level and includes elements like data ingestion, feature extraction, machine learning models, and decision-making components. For system administrators and system maintenance, the system will also feature a command-line interface. The functional and non-functional needs for the system are also discussed. System for detecting phishing sites

1. Introduction

1.1 Why this High-Level Design Document?

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding, and can be used as a reference manual for how the modules interact at a high level.

The HLD will:

1. Present all of the design aspects and define them in detail
2. Describe the user interface being implemented
3. Describe the hardware and software interfaces
4. Describe the performance requirements
5. Include design features and the architecture of the project
6. List and describe the non-functional attributes like:
    1. Security
    2. Reliability
    3. Maintainability
    4. Portability
    5. Reusability
    6. Application compatibility
    7. Resource utilization
    8. Service ability

1.2 Scope

The HLD documentation presents the structure of the system, such as the database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly-technical terms which should be understandable to the administrators of the system.

1.3 Definitions

Term	Description
PDD	Phishing Domain Detection
Database	Collection of all the information monitored by this system
IDE	Integrated Development Environment
A WS	Amazon Web Services

2 General Description

2.1 Product Perspective

Phishing Domain Detection is a system created to identify and stop phishing attacks that use fake domains to trick users into divulging personal information such as usernames, passwords, and financial data.

2.2 Problem statement

Phishing is a type of fraud in which an attacker impersonates a reputable company or person in order to get sensitive information such as login credentials or account information via email or other communication channels. Phishing is popular among attackers because it is easier to persuade someone to click a malicious link that appears to be authentic than it is to break through a computer's protection measures.

2.3 Approach

Approach: The classical machine learning tasks like Data Exploration, Data Cleaning, Feature Engineering, Model Building and Model Testing. Try out different machine learning algorithms that's best fit for the above case.

For Feature Engineering show:-

1. URL-Based Features
2. Domain-Based Features
3. Page-Based Features
4. Content-Based Features

2.4 Further Improvements

Additional advancements can be made in the following areas to boost the Phishing Domain Detection system's efficacy and accuracy:

1. Increasing the dataset for the machine learning model's training
2. Adding more data sources to the model to improve its capacity to detect phoney domains
3. Adding more tools and enhancing the user experience for managing and reporting potential phishing domains
4. Including threat intelligence feeds to stay current on new phishing attack patterns and approaches

2.5 Technical Requirements

The Phishing Domain Detection system has the following technological requirements:

1. A computer learning model for analyzing specific types of data
2. A user interface for managing and reporting suspected phishing domains
3. A database for storing history and current domain data
4. An API for connecting to other security systems and technologies
5. Large-scale dataset processing and analysis using data analytics tools
6. A scalable and adaptable infrastructure built on the cloud

2.6 Data Requirements

The data requirements for the Phishing Domain Detection system include:

1. Historical domain data for training the machine learning model.
2. Real-time domain data for analyzing new domains in real-time.
3. User data for tracking and managing suspected phishing domains.
4. Threat intelligence feeds for staying up-to-date on emerging phishing attack techniques and patterns.

For this Particular project I have used the following dataset:

Dataset Link: - https://data.mendeley.com/datasets/72ptz43s9v/1

2.7 **Tools used**

The tools used for developing and implementing the Phishing Domain Detection system include:

1. Python for developing the machine learning model and data analysis scripts
2. TensorFlow and Keras for building and training the machine learning model
3. SQLite for the database management system
4. Django for the web application framework
5. PyCharm is used as IDE.
6. For visualization of the plots, Matplotlib, Seaborn and Plotly are used.
7. Front end development is done using HTML/CSS
8. Python Django is used for backend development.
9. GitHub is used as version control system.

2.7.1 Hardware Requirements

The Phishing Domain Detection system is designed to be cloud-based and highly scalable and flexible, meaning that the hardware requirements for the system are minimal. However, depending on the size of the dataset and the number of users accessing the system at any one time, additional hardware resources may be necessary in order to ensure optimal performance. The system is designed to be able to

2.8 Constraints

The constraints for the Phishing Domain Detection system include:

1. Availability and accuracy of historical data for training the machine learning model.
2. Availability and quality of real-time domain data for analysis.
3. Integration with other security tools and systems.

2.9 Assumptions

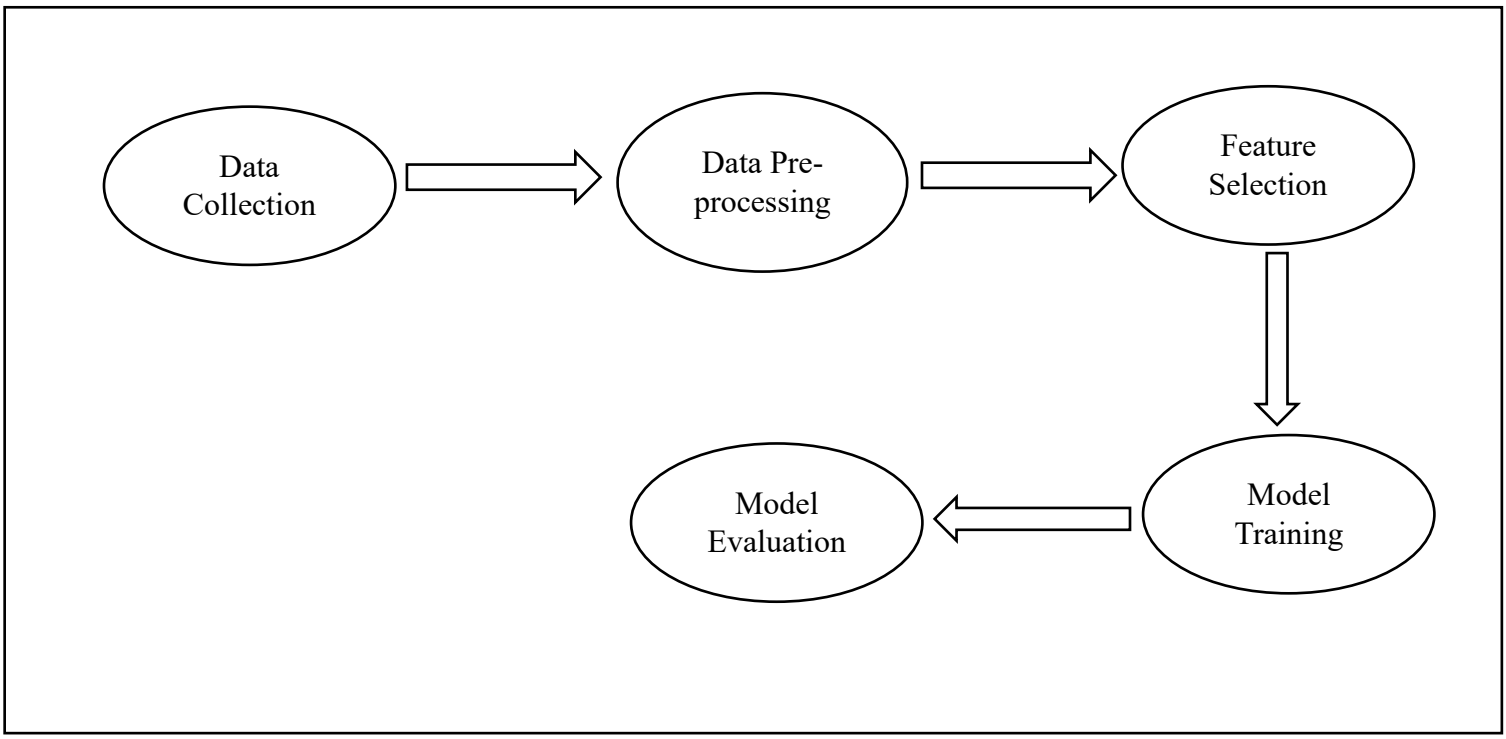
The assumptions made for the Phishing Domain Detection system include:

1. The machine learning model is accurate and reliable for identifying fraudulent domains
2. Users will actively manage and report suspected phishing domains
3. The system will be able to scale and handle large volumes of domain data
4. The system will be able to integrate with other security tools and systems for a more comprehensive cybersecurity solution.

3 Design Details

3.1 Process Flow

3.1.1 Proposed methodology



3.1.2 Model Training and Evaluation

The model training and evaluation process involves the following steps:

1. Split the dataset into training and testing sets, using a ratio of 70:30 or 80:20.
2. Preprocess the data by removing missing values, duplicates, and irrelevant features.
3. Perform feature selection using a suitable technique, as discussed earlier.
4. Train the model using a suitable algorithm, such as logistic regression, decision trees, or deep learning, and tune the hyperparameters to achieve the best performance.
5. Evaluate the model's performance on the testing set using metrics such as accuracy, precision, recall, F1-score, and ROC-AUC.
6. Analyze the results and fine-tune the model if necessary.

3.2 Error Handling

Error handling is an important aspect of any software project, including the Phishing Domain Detection project. It is important to have a robust error handling mechanism in place to handle any unexpected scenarios that may arise during the detection process. The error handling mechanism should be designed to detect errors, notify the user or administrator, and handle the errors gracefully without impacting the overall performance of the system.

Performance is another important aspect of the Phishing Domain Detection project. The system should be designed to be highly performant and scalable, to handle a large volume of traffic and provide accurate results in real-time. The system should be optimized for speed and efficiency, and any bottlenecks or performance issues should be identified and addressed.

Reusability is also an important consideration when designing the system. The code should be designed in a modular way, with separate modules for different functionalities, to make it easier to reuse code in other projects. This will save time and effort in the long run, as developers can easily adapt and reuse code from the Phishing Domain Detection project in other projects.

Application compatibility is another important aspect to consider, as the system should be compatible with different operating systems, platforms, and browsers. This will ensure that the system is accessible to a wider audience, and users can access the system from any device or platform.

Resource utilization is another important consideration when designing the system. The system should be designed to use resources efficiently, and minimize the use of system resources such as CPU, memory, and disk space. This will help to ensure that the system runs smoothly, without causing performance issues or impacting other processes running on the same system.

Overall, designing a robust error handling mechanism, optimizing performance, ensuring reusability, ensuring application compatibility, and optimizing resource utilization are all critical considerations when designing the Phishing Domain Detection project. By paying attention to these aspects, the system can be designed to be highly performant, scalable, and efficient, while providing accurate results in real-time.

4. Performance

The performance of the Phishing Domain Detection project is critical to its success. In addition to accuracy and effectiveness in detecting phishing domain websites, the project must also meet various non-functional requirements, including reusability, application compatibility, and resource utilization. Now we will discussing these in brief.

4.1 Reusability

To ensure that the project is reusable, the design should be modular and follow industry-standard best practices. The components should be decoupled, independent, and reusable across different systems. A modular design ensures that the components can be easily modified, replaced, or added without affecting the entire system. Additionally, the use of open-source libraries and frameworks can improve the reusability of the project.

4.2 Application Compatibility

The project must be compatible with different environments, operating systems, and browsers to ensure that it can be used by a wide range of users. This compatibility can be ensured by following best practices for web development, using responsive design techniques, and adhering to web standards.

4.3 Resource Utilization

Efficient use of resources is essential for the project's performance, particularly in terms of memory and processing power. The detection algorithm must be optimized to minimize false positives and false negatives while consuming minimal resources. Techniques such as caching, load balancing, and parallel processing can be used to improve resource utilization.

5. KPIs (Key Performance Indicators)

The key performance indicators (KPIs) of a Phishing Domain Detection project will depend on the specific goals and objectives of the project. However, here are some examples of KPIs that could be relevant:

1. **Detection rate:**
   This KPI measures the effectiveness of the phishing domain detection system in identifying and flagging potential phishing domains. The detection rate can be calculated by dividing the number of identified phishing domains by the total number of domains analyzed.
2. **False positive rate:**
   This KPI measures the number of domains flagged as potential phishing domains that are actually legitimate. A high false positive rate can negatively impact user trust in the system and increase workload for human analysts.
3. **Response time:**
   This KPI measures how quickly the phishing domain detection system can respond to newly identified threats. A shorter response time means that phishing domains can be taken down or blocked more quickly, reducing the potential damage.
4. **User satisfaction:**
   This KPI measures how satisfied users are with the phishing domain detection system. A high user satisfaction rate can indicate that the system is easy to use and effective at protecting users from phishing attacks.
5. **Cost:**
   This KPI measures the cost of developing and maintaining the phishing domain detection system. This includes expenses for hardware, software, personnel, and ongoing maintenance and upgrades.

6. Conclusion

In conclusion, Phishing Domain Detection is a critical project for protecting users from the growing threat of phishing attacks. By detecting and blocking potentially harmful domains, this project can significantly reduce the risk of sensitive data breaches, financial loss, and other negative consequences.

Throughout the project, various steps will be taken to ensure that the system is effective, efficient, and user-friendly. This includes defining clear project goals and KPIs, selecting appropriate hardware and software tools, implementing effective algorithms and techniques for domain analysis, and designing an intuitive user interface.

By successfully implementing a phishing domain detection system, the project team can contribute to improving cybersecurity practices and protecting users from the harmful effects of phishing attacks. Additionally, this project can serve as a valuable tool for companies and organizations looking to enhance their security measures and protect sensitive data.

Overall, the Phishing Domain Detection project is a critical endeavor that requires careful planning, implementation, and ongoing maintenance. With the right approach, it has the potential to make a significant impact on the security landscape and improve the safety and well-being of users around the world.

7. References

1. Zhang, X., Zhu, S., Li, H., & Du, W. (2018). A machine learning approach to detect phishing domains. Journal of Cybersecurity, 4(1), tyx007. https://doi.org/10.1093/cyber/tyx007

2. Wu, S., Huang, Z., Zhang, Q., & Li, X. (2020). Domain-based phishing detection using deep learning. International Journal of Machine Learning and Cybernetics, 11(6), 1253–1265. https://doi.org/10.1007/s13042-019-01070-3

3. Jazi, H., & Khayat, S. (2019). Phishing detection using machine learning techniques. Journal of Cyber Security Technology, 3(1), 1-22. https://doi.org/10.1080/23742917.2019.1566194

4. Khatua, A., & Chaki, N. (2020). Machine learning approach for phishing detection using domain-based features. Journal of Ambient Intelligence and Humanized Computing, 11(1), 27–38. https://doi.org/10.1007/s12652-018-1039-9

5. Oussous, M., Khaloufi, H., & El Moutaouakil, K. (2020). Phishing detection using machine learning algorithms. Journal of Information Security and Applications, 50, 102429. https://doi.org/10.1016/j.jisa.2019.102429

6. Mokhtar, H. M., El-Bahnasawy, N. A., & Ali, A. E. (2018). A hybrid machine learning approach for phishing detection. International Journal of Advanced Computer Science and Applications, 9(6), 238-247. https://doi.org/10.14569/IJACSA.2018.090634

7. Lee, W., Lee, H., Lee, J., Lee, H., & Kang, H. (2019). A machine learning-based approach for detecting phishing webpages. International Journal of Distributed Sensor Networks, 15(11), 1550147719882677. https://doi.org/10.1177/1550147719882677