High Level Design (HLD)

Phishing Domain Detection

April 2, 2022

Abstract

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.

The goal of this project is to build a machine learning model which will predict whether a website is real or malicious.

Revision number: 1.0

Last date of revision: April 2, 2022

Contents

1 Introduction **3**

1.1 Description of High Level Design (HLD) document 4

1.2 Scope 4

1.3 Definitions 4

2 General Description **5**

2.1 Product perspective 6

2.2 Problem statement 6

2.3 Proposed solution 6

2.4 Future improvements 6

2.5 Technical requirements 6

2.6 Data requirements 7

2.6.1 URL-based features 7

2.6.2 Domain-based features 7

2.6.3 URL directory-based features 8

2.6.4 URL file name-based features 9

2.6.5 URL parameter-based features 10

2.6.6 Features based on resolving URL and external services 10

2.7 Tools used 12

2.7.1 Hardware requirements 14

2.8 Constraints 14

2.9 Assumptions 14

3 Design details **15**

3.1 Process flow 16

3.1.1 Model training and evaluation 16

3.1.2 Deployment process 17

3.2 Event log 17

3.3 Error handling 17

4 Performance **18**

4.1 Reusability 19

4.2 Application compatibility 19

4.3 Resource utilization 19

4.4 Deployment 19

5 Conclusion **20**

6 References **22**

Chapter 1

Introduction

1.1 Description of High Level Design (HLD) document

Return to table of contents

The purpose of this document is to add the necessary details 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. In general a HLD document:

1. presents design aspects and defines them in detail
2. describes user interface
3. describes hardware and software interfaces
4. describes performance requirements
5. lists and describes non-functional attributes like security, reliability, maintainability, portability, reusability, application compatibility, resource utilization, serviceability, et cetera.

1.2 Scope

Return to table of contents

The document presents the structure of the system, such as the database architecture, application architecture, application flow, and technology architecture. The HLD uses non-technical and slightly-technical terms which should be understandable to administrators of the system.

1.3 Definitions

Return to table of contents

Term	Definition
ML	Machine Learning
Database	Phishing Websites Dataset hosted on Astra DB

Chapter 2

General Description

2.1 Product perspective

Return to table of contents

Phishing detection is a ML solution which allows to identify malicious websites.

2.2 Problem statement

Return to table of contents

To create a ML solution for phishing detection.

2.3 Proposed solution

Return to table of contents

The solution provided in this document is a phishing detection tool constructed on Phishing Websites Dataset. The tool accepts some characteristics of a website as an input. Then it processes the information provided and concludes whether the website is malicious or not.

2.4 Future improvements

Return to table of contents

The solution should be evaluated on continuous bases depending on availability of new info. This will make the tool flexible and up-to-date as new type of phishing can emerge.

2.5 Technical requirements

Return to table of contents

Phishing detection tool is accessible via computer, laptop, tablet, and smartphone.

2.6 Data requirements

Return to table of contents

The phishing detection tool needs the following information:

2.6.1 URL-based features

1. Number of "." signs (qty_dot_url)
2. Number of "-" signs (qty_hyphen_url)
3. Number of "_" signs (qty_underline_url)
4. Number of "/" signs (qty_slash_url)
5. Number of "?" signs (qty_questionmark_url)
6. Number of "=" sings (qty_equal_url)
7. Number of "@" signs (qty_at_url)
8. Number of "&" signs (qty_and_url)
9. Number of "!" signs (qty_exclamation_url)
10. Number of " " signs (qty_space_url)
11. Number of "~" signs (qty_tilde_url)
12. Number of "," signs (qty_comma_url)
13. Number of "+" signs (qty_plus_url)
14. Number of "*" signs (qty_asterisk_url)
15. Number of "#" signs (qty_hashtag_url)
16. Number of "$" signs (qty_dollar_url)
17. Number of "%" signs (qty_percent_url)
18. Top level domain character length (qty_tld_url)
19. Number of characters (length_url)
20. Is email present or not(email_in_url)

2.6.2 Domain-based features

1. Number of "." signs (qty_dot_domain)
2. Number of "-" signs (qty_hyphen_domain)

3. Number of "_" signs (qty_underline_domain)
4. Number of "/" signs (qty_slash_domain)
5. Number of "?" signs (qty_questionmark_domain)
6. Number of "=" signs (qty_equal_domain)
7. Number of "@" signs (qty_at_domain)
8. Number of "&" signs (qty_and_domain)
9. Number of "!" signs (qty_exclamation_domain)
10. Number of " " signs (qty_space_domain)
11. Number of "signs (qty_tilde_domain)
12. Number of "," signs (qty_comma_domain)
13. Number of "+" signs (qty_plus_domain)
14. Number of "*" signs (qty_asterisk_domain)
15. Number of "#" signs (qty_hashtag_domain)
16. Number of "$" signs (qty_dollar_domain)
17. Number of "%" signs (qty_percent_domain)
18. Number of vowels (qty_vowels_domain)
19. Number of domain characters (domain_length)
20. URL domain in IP address format (domain_in_ip)
21. "server" or "client" in domain (server_client_domain)

2.6.3 URL directory-based features

1. Number of "." signs (qty_dot_directory)
2. Number of "-" signs (qty_hyphen_directory)
3. Number of "_" signs (qty_underline_directory)
4. Number of "/" signs (qty_slash_directory)
5. Number of "?" signs (qty_questionmark_directory)
6. Number of "=" signs (qty_equal_directory)
7. Number of "@" signs (qty_at_directory)
8. Number of "&" signs (qty_and_directory)
9. Number of "!" signs (qty_exclamation_directory)

10. Number of " " signs (qty_space_directory)
11. Number of "signs (qty_tilde_directory)
12. Number of "," signs (qty_comma_directory)
13. Number of "+" signs (qty_plus_directory)
14. Number of "*" signs (qty_asterisk_directory)
15. Number of "#" signs (qty_hashtag_directory)
16. Number of "$" signs (qty_dollar_directory)
17. Number of "%" signs (qty_percent_directory)
18. Number of directory characters (directory_length)

2.6.4 URL file name-based features

1. Number of "." signs (qty_dot_file)
2. Number of "-" signs (qty_hyphen_file)
3. Number of "_" signs (qty_underline_file)
4. Number of "/" signs (qty_slash_file)
5. Number of "?" signs (qty_questionmark_file)
6. Number of "=" signs (qty_equal_file)
7. Number of "@" signs (qty_at_file)
8. Number of "&" signs (qty_and_file)
9. Number of "!" signs (qty_exclamation_file)
10. Number of " " signs (qty_space_file)
11. Number of "signs (qty_tilde_file)
12. Number of "," signs (qty_comma_file)
13. Number of "+" signs (qty_plus_file)
14. Number of "*" signs (qty_asterisk_file)
15. Number of "#" signs (qty_hashtag_file)
16. Number of "$" signs (qty_dollar_file)
17. Number of "%" signs (qty_percent_file)
18. Number of file name characters (file_length)

2.6.5 URL parameter-based features

1. Number of "." signs (qty_dot_params)
2. Number of "-" signs (qty_hyphen_params)
3. Number of "_" signs (qty_underline_params)
4. Number of "/" signs (qty_slash_params)
5. Number of "?" signs (qty_questionmark_params)
6. Number of "=" signs (qty_equal_params)
7. Number of "@" signs (qty_at_params)
8. Number of "&" signs (qty_and_params)
9. Number of "!" signs (qty_exclamation_params)
10. Number of " " signs (qty_space_params)
11. Number of "signs (qty_tilde_params)
12. Number of "," signs (qty_comma_params)
13. Number of "+" signs (qty_plus_params)
14. Number of "*" signs (qty_asterisk_params)
15. Number of "#" signs (qty_hashtag_params)
16. Number of "$" signs (qty_dollar_params)
17. Number of "%" signs (qty_percent_params)
18. Number of parameters characters (params_length)
19. Top-Level Domain present in parameters (tld_present_params)
20. Number of parameters (qty_params)

2.6.6 Features based on resolving URL and external services

1. Domain lookup time response (time_response)
2. Domain has Sender Policy Framework (domain_spf)
3. Autonomous System Number (asn_ip)
4. Domain activation time (in days) (time_domain_activation)
5. Domain expiration time (in days) (time_domain_expiration)
6. Number of resolved IPs (qty_ip_resolved)
8. Number of resolved Name Servers (qty_nameservers)

9. Number of Mail eXchanger servers (qty_mx_servers)
10. Time-To-Live (TTL) (ttl_hostname)
11. Has valid TLS /SSL certificate (tls_ssl_certificate)
12. Number of redirects (qty_redirects)
13. Is URL indexed on Google (url_google_index)
14. Is domain indexed on Google (domain_google_index)
15. Is URL shortened (url_shortened)
16. Is phishing website (phishing)

2.7 Tools used

Return to table of contents

Jupyter notebook is used as a notebook environment to create visualizations for exploratory data analysis.

Pycharm is used as an IDE.

Visualizations are done using matplotlib and seaborn.

Heroku is used for deployment.

Astra DB hosts the data.

CQL is used to retrieve the data.

Front end development is done using HTML and CSS.

Flask is used for backend development.

Github is used as a version control system

CircleCi is used for CI-CD.

2.7.1 Hardware requirements

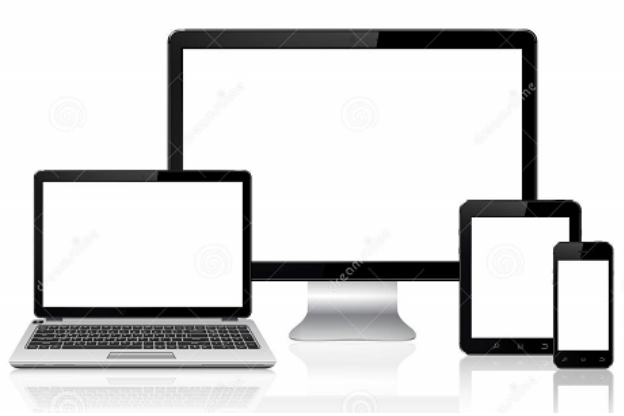
Return to table of contents

Computer

Laptop

Tablet

Smartphone



2.8 Constraints

Return to table of contents

The phishing detection solution must be user friendly and users do not need to know the inner workings to be able to exploit it.

2.9 Assumptions

Return to table of contents

The main objective of this project is to detect malicious websites. All of that will be possible due to a ML model applied to the database. Moreover all aspects of the project have the ability to work together in accordance with a designer's expectation.

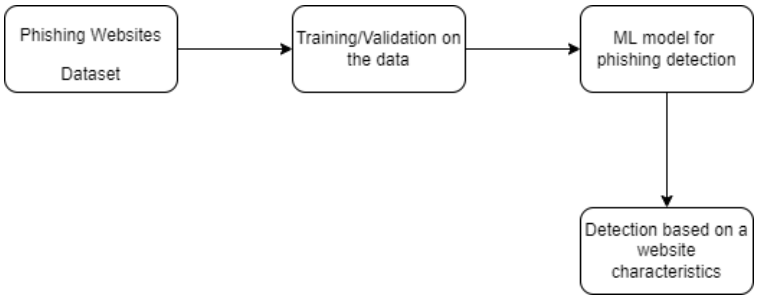
Chapter 3

Design details

3.1 Process flow

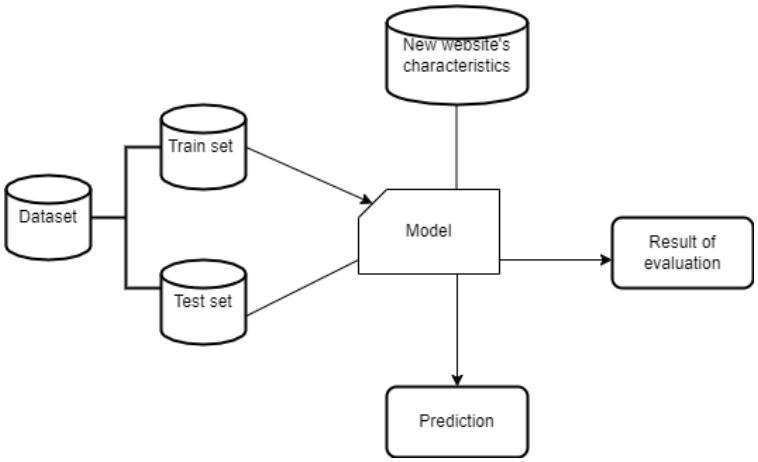
Return to table of contents

For phishing dectection we will use one of the following ML models: Support Vector Machine, XGBoost, Naive Bayes, Logistic Regression, Random Forest, and LightGBM. We will choose the best one in terms of accuracy (ROC-AUC score). Process flow diagram is depicted below.



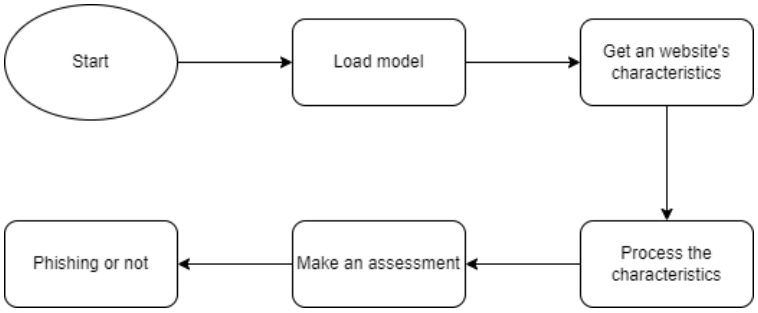
3.1.1 Model training and evaluation

Return to table of contents



3.1.2 Deployment process

Return to table of contents



3.2 Event log

Return to table of contents

The logging is applied to every process, but logging results will not be accessible for end users. Moreover logging is made accessible via console and a file for developers. Logging is mandatory as it allows to find and debug issues more easily.

3.3 Error handling

Return to table of contents

Anything falling outside the normal and intended usage should be considered as an error. The user interface is constructed in a way that, during inputting a website's URL or a list of URLs, chances to make a mistake leading to an error are negligible. Therefore an end user does not see error messages during exploitation of the tool.

Chapter 4

Performance

The phishing detection tool is used to identify malicious websites. Therefore it should be as accurate as possible. It is very important to retrain the model for enhancing its performance as soon as new data will be available.

4.1 Reusability

Return to table of contents

The code and its components should be reused with no problems.

4.2 Application compatibility

Return to table of contents

This project has different components and python is used as an interface between them. Each component has its own task to perform, and python ensures proper transfer of information.

4.3 Resource utilization

Return to table of contents

When any task is performed, it will likely use all the processing power available until it is finished.

4.4 Deployment

Return to table of contents

Chapter 5

Conclusion

The phishing detection tool will identify malacious websites thus preventing an end-user from visiting them.

Chapter 6

References

1. Phishing websites dataset
2. Phishing websites dataset description
3. Github link to phishing websites dataset project
4. PhishTank
5. URL feature extractor codes
6. google.com
7. filesmerge.com
8. app.diagrams.net