Software engineering approach to software development

Technology, Computer

LO3 – Software Engineering Approach to Software and Systems Development

Design and Modeling of the Phishing Detection Solution

The Software Solution, is based on heuristics implementation of an expert system that is capable of detecting possible phishing websites using URL, Domain, and Abnormal features. The proposed system will be capable of accepting a URL as user input and detect whether it is a legitimate, suspicious or a phished URL. This determination will be based on several grouped heuristics implementations as per feature category.

System Architecture

The system architecture entails a User Interface design used for URL input by a user, which then displays the results to the user. Upon input of the URL by the user, the phishing detection system utilizes standard inbuilt JAVA functions to extract the resulting URL's site features, which are collected and utilized in the URL input classification phase. This is illustrated in figure 1 below, which represents the system architecture. It involves five major steps: (1) User Input, (2) Features Extraction, (3) Classification of the URL, (4) Results Output, and (5) Evaluation.

User Input– Involves input of the URL (either legitimate or Phishing) or upload of URL containing file. Systems then extracts URL features such as: number of page visits, number of visitors to URL, giving an overview of URL category.

Feature Extraction– Relevant URL features are extracted to help categorize whether legitimate or phishing URLs based on three groups: URL address

based features, domain-based features, and abnormal-based features.

URL Classification– Entails subjecting URL features to different heuristics to determine whether it is legitimate, suspicious or phished.

Classified URLs Output –Classification results are displayed in the form of tables and graphs providing a visual representation of the URLs. In addition, in case of phishing category, a severity scale (of 1-5) is applied to each URL to gauge its impact.

Evaluation –Classification results are evaluated based on recall and precision methods. A confusion matrix is applied in the evaluation of the classification process using true negatives, false negatives, false positives and true positives.

Figure 1: System Architecture of the Phishing Detection System Based on Heuristics

UML Approach

Use Case Diagrams

Phishing Detection Based on Heuristics Use Case Diagram

URL-related Features Use Case Diagram

Abnormal-based Features Use Case Diagram

Domain-based Features Use Case Diagram

Class Diagrams

System Class Diagram

Features Class Diagram

Classification Class Diagram

Activity Diagrams

Home Page

Features Page

Classification Page

Visualization Page

Sequence Diagrams / Communication Diagrams

URL Input

Feature Extraction

URL Classification

Results Output

Results Evaluation

Statechart Diagram

URL Classification and Evaluation

Data Verification and Validation

Data verification and validation of the phishing detection system based on heuristics will be based on Use Case Testing, where negative and positive test cases will be executed. This is will involve feeding the system with both malicious and legitimate URLs to validate and verify whether URL classification is accurate, correct or not. In addition, various heuristics combinations will be applied.

Unit testing method will be applied to validate each software module unit as per the Usecase model. For each test case the following elements will be specified for system validation: the test case description; the input URL; heuristics used; results; and severity of results.

Importance of Data Verification and Validation

Ensure correct implementation of system data flows as outlined in the system design specifications
Ensure adequacy of implemented system, where the system solution satisfies proposed requirements
Eliminate possible redundancy, by removing unnecessary system design elements, hence significantly improving system design quality
Promote logical consistency of solution design across the entire system, while eliminating any contradictory portions
Ensure there is internal completeness of the system design based on various conditions combinations
Promote design principle compliance, such as, low coupling, high cohesion, and concerns separation

To ensure module integration and interfacing is accurate through explicit communication between modules (Kung 2007)

References

Kendall, K & Kendall, J 2011, *Systems Analysis and Design,* Prentice Hall, New Jersey, USA.

Kung, D 2007, ‘ Software Verification and Validation’, *Encyclopedia of Computer Science and Engineering,* viewed 2 March 2017 < >