

**Sri Lanka Institute of Information Technology**

**Malware detection system**

Project Proposal

Information Systems Project 2023

|  |  |
| --- | --- |
| Student Name | Student ID |
| W.A.C.S. Weerasinghe | It20609580 |

Table of Contents

Table of Contents

1. Introduction 1

1.1 Purpose 1

1.2 Intended Audience and Reading Suggestions 1

1.3 Project Scope 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Features 2

2.3 Operating Environment 2

2.4 Design and Implementation Constraints 2

2.5 Assumptions and Dependencies 3

3. System Features 3

3.1 System Feature 1 3

3.2 System Feature 2 (and so on) 4

4. External Interface Requirements 4

4.1 User Interfaces 4

4.2 Hardware Interfaces 4

4.3 Software Interfaces 4

4.4 Communications Interfaces 4

5. Other Nonfunctional Requirements 5

5.1 Performance Requirements 5

5.2 Safety Requirements 5

5.3 Security Requirements 5

6. References 5

# Introduction

## Purpose

## *The purpose of this project is to develop an effective and efficient machine learning-based malware detection system with a web interface. The system is designed to analyze and identify potentially malicious .apk files using advanced neural network and Support Vector Classifier (SVC) algorithms, contributing to enhanced cybersecurity.*

## Intended Audience and Reading Suggestions

This report is intended for a diverse audience, including cybersecurity professionals, software developers, researchers, and stakeholders interested in malware detection technologies. For those looking for a high-level overview, start with the introduction and overall description sections Technical readers can delve deeper into the system features and implementation details.

## Project Scope

The scope of this project encompasses the development and deployment of a web-based malware detection system for .apk files. It includes the implementation of neural network and SVC algorithms for robust malware classification. The report covers the design, functionality, and constraints of the system..

# Overall Description

## Product Perspective

This malware detection system plays a vital role in the broader context of cybersecurity. It offers a user-friendly web interface that allows users to upload .apk files for analysis. The system leverages machine learning techniques to classify these files as either benign or potentially malicious, aiding in the identification and mitigation of cybersecurity threats.>

## Product Perspective

The architecture of the system comprises a front-end web interface and a back-end engine powered by neural network and SVC algorithms. The interaction between these components ensures accurate and efficient malware detection.

## Operating Environment

The system is designed to operate within a web-based environment, making it accessible to users across various platforms. It can be hosted on cloud infrastructure or on-premises servers, requiring minimal hardware resources.

## Design and Implementation Constraints

The project is subject to certain constraints, including budget limitations and a predetermined timeline for development and testing. Additionally, the effectiveness of the malware detection model heavily depends on the quality and quantity of training data available.

## Assumptions and Dependencies

Key assumptions for this project include the availability of a sufficient dataset of labeled .apk files for training and the compatibility of the chosen machine learning algorithms with the system architecture. Dependencies include access to web hosting services and the use of machine learning libraries for algorithm implementation

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## System Feature 1

3.1.1 Description and Priority

This feature involves the implementation of a neural network-based malware detection model, which is a high-priority component of the system. The neural network will be trained on a diverse dataset of .apk files to learn patterns and characteristics of both benign and malicious applications.

3.1.2 Stimulus/Response Sequences

When a user uploads an .apk file via the web interface, the neural network model will analyze the file's attributes and behavior, producing a response indicating whether the file is likely malicious or benign.

3.1.3 Functional Requirements

* The system must preprocess and extract relevant features from uploaded .apk files.
* The neural network architecture should be designed, trained, and fine-tuned for optimal malware detection performance.
* Real-time feedback and results should be provided to the user via the web interface.

## System Feature 2 (and so on)

3.2.1 Description and Priority

This feature involves integrating a Support Vector Classifier (SVC) into the malware detection system. The SVC will provide an additional layer of analysis, enhancing the accuracy of malware classification. It is considered a critical component.

3.2.2 Stimulus/Response Sequences

The SVC will receive preprocessed data from uploaded .apk files and provide a classification decision, complementing the neural network's output.

3.2.3 Functional Requirements

* + The system must preprocess data in a format suitable for input to the SVC.
  + The Support Vector Classifier must be trained and optimized for efficient malware detection.
  + The combined output of the neural network and SVC should contribute to the final malware classification.

# External Interface Requirements

## User Interfaces

The user interface of the malware detection system is web-based, providing an intuitive and user-friendly experience. Users interact with the system through a standard web browser. The interface allows users to upload .apk files for analysis, view analysis results, and access system settings..

## Hardware Interfaces

the system does not have specific hardware interface requirements beyond standard web hosting infrastructure. It is designed to be hardware-agnostic, allowing it to run on various hosting platforms without imposing hardware constraints.

## Software Interfaces

The malware detection system relies on several software interfaces for its functionality:

Web Server: The system communicates with a web server to handle user requests and serve the

web interface.

Machine Learning Libraries: It utilizes machine learning libraries (e.g., TensorFlow, Scikit-learn) to implement the neural network and Support Vector Classifier algorithms.

Database: The system may require a database for storing user information, logs, and potentially malicious .apk file data.

Operating System: Compatibility with common operating systems (e.g., Linux, Windows) is necessary for deployment.

## Communications Interfaces

The system communicates with users and other components through standard HTTP/HTTPS protocols for web-based interactions. It also communicates with the machine learning models for analysis and classification tasks, using API calls or library functions.

# Other Nonfunctional Requirements

## Performance Requirements

Response Time: The system should provide near-real-time feedback to users, with a response time of under a few seconds for malware analysis.

Scalability: It should be designed to handle a scalable number of concurrent users and file uploads.

Accuracy: The neural network and SVC models should achieve a high level of accuracy in malware detection, with a minimal false-positive rate.

## Safety Requirements

User Privacy: Ensure the privacy of user data and uploaded .apk files. Implement appropriate data protection measures, including encryption and secure storage.

## Security Requirements

* Data Security: Protect the system and its components from security threats such as data breaches, DDoS attacks, and unauthorized access.
* Malware Containment: Implement measures to isolate potentially malicious .apk files to prevent them from causing harm to the system or other users.
* Regular Updates: Keep the system and its dependencies (e.g., machine learning libraries) up-to-date with security patches to mitigate vulnerabilities.

# References

Abualganam, O. (2023, 1). *Android Malware Detection System Based on Ensemble Learning*. Retrieved from researchgate: https://www.researchgate.net/publication/370603532\_Android\_Malware\_Detection\_System\_Based\_on\_Ensemble\_Learning

Panda, G. (2021, 11). *Machine Learning Based Improved Malware Detection Schemes*. Retrieved from IEEE: https://ieeexplore.ieee.org/document/9377123

Sewak, M. (2018). *An investigation of a deep learning based malware detection system*. Retrieved from researchgate: https://www.researchgate.net/publication/327008061\_An\_investigation\_of\_a\_deep\_learning\_based\_malware\_detection\_system

Singh, J. (2021, 01). *A survey on machine learning-based malware detection in executable files*. Retrieved from sciencedirect: https://www.sciencedirect.com/science/article/abs/pii/S1383762120301442