**Software Requirements Specification**

For

**MALWARE DETECTION USING ML and Python**

Prepared by

|  |  |  |
| --- | --- | --- |
| **Specialization** | **SAP ID** | **Name** |
| B.Tech LLB-Cyber Law | 500068884 | Isha Mittal |
| B.Tech LLB-Cyber Law | 500067119 | Ramya Mihir |



Department of Informatics

School Of Computer Science

UNIVERSITY OF PETROLEUM & ENERGY STUDIES,

DEHRADUN- 248007. Uttarakhand

Table of Contents

|  |  |  |
| --- | --- | --- |
| **Topic** | | **Page No** |
| Table of Content | | 2 |
| 1 | Introduction | 3-4 |
|  | 1.1 Purpose of the Project | 3 |
|  | 1.2 Target Beneficiary | 3 |
|  | 1.3 Project Scope | 3 |
|  | 1.4 References | 3 |
| 2 | Project Description | 5-12 |
|  | 2.1 Reference Algorithm | 5 |
|  | 2.2 Data/ Data structure | 5 |
|  | 2.3 SWOT Analysis | 6 |
|  | 2.4 Project Features | 7 |
|  | 2.5 Design and Implementation Constraints | 7 |
|  | 2.6 Design diagrams | 8 |
|  | 2.7 Assumption and Dependencies | 12 |
| 3 | System Requirements | 12-13 |
|  | 3.1 User Interface | 12 |
|  | 3.2 Software Interface | 12 |
|  | 3.3 Database Interface | 13 |
| 4 | Non-functional Requirements | 13 |
|  | 4.1 Performance requirements | 13 |
|  | 4.2 Security requirements | 13 |
|  | 4.3 Software Quality Attributes | 13 |

1. **INTRODUCTION**
   1. **Purpose of Project:**

When COVID-19 hit the world, it altered the working pattern of all the people around the world. Along with this, there has been an exponential growth in the number of malwares and the cyber-crime rates.

Modern malwares use sophisticated techniques such as polymorphism and metamorphism to thwart the malware detection and analysis. Detecting malware on the basis of their features and behavior is critical for the computer security community. Most anti-virus depends on the signature-based detection which is relatively easy to evade and is ineffective for zero-day exploit-based malwares.

Also, it has been noticed that the Internet has become a vital part in our lives with increased usage of services like online banking, online reservation etc., and our dependence on the Internet is expected to grow. With the rise of the Internet, there has been huge growth in the amount of malwares in the world.

With this project, we provide a new approach to identify malwares using static analysis, i.e. without executing. With the help of different machine learning models, we will identify malwares if present in any file, in order to prevent any further attacks.

* 1. **Target Beneficiary:**

The target audience and the people who will majorly get benefitted from this project are the students as well as the working professionals who are these days working in online mode due to the pandemic.

This application will promote an easy use to identify the files that they receive over emails, or sms, or any other e-mode, to scan before opening any malware file and get trapped. So, it is for mainly all the students, and professionals, who are more likely to be active on internet.

* 1. **Project Scope:**

The project scope is to create an easy platform for malware detection in real-time environment where the students, working professionals, and even the new internet generation people can scan and check for malwares in any file, in order to prevent themselves from getting trapped in cyber world. This project is also to create cyber awareness among the people so that the crimes that are increasing these days due to internet, are reduced to a smaller extend.

Our main objective is to Identify and classify malwares using static analysis i.e. without executing with the help of Machine learning models. So, we are basically providing a software which will run and ask for the file to be checked for malwares and notify the user for either to open it or not.

* 1. **References:**

[1] Machine Learning Models and applications

(https://media.kaspersky.com/en/enterprise-security/Kaspersky-Lab-Whitepaper-Machine-Learning.pdf)

[2] ML models required for Malware Analysis

(https://www.ccsinet.com/blog/machine-learning-malware-analysis/)

[3] Working method for Malware Analysis

(https://sectigostore.com/blog/malware-analysis-what-it-is-how-it-works/)

[4] Malware Detection and Removal Types and Introduction

(https://www.imperva.com/learn/application-security/malware-detection-and-removal/)

[5] Images

(https://link.springer.com/chapter/10.1007/978-3-319-63673-3\_21) (https://ai.plainenglish.io/malware-detection-using-hybrid-analysis-4b0ef471e844) (https://www.researchgate.net/publication/289674086\_Detecting\_Malware\_for\_Android\_Platform\_An\_SVM-\_based\_Approach/figures?lo=1&utm\_source=google&utm\_medium=organic) (https://www.semanticscholar.org/paper/Malware-detection-using-assembly-code-and-control-Anju-Harmya/96b22134053f4a0cce2a8f9961131a42af831b8f)

1. **PROJECT DESCRIPTION**
   1. **Reference Algorithm**

The project basically follows quite a simple methodology and the reference algorithm explaining our methodology is as follows.

Steps:

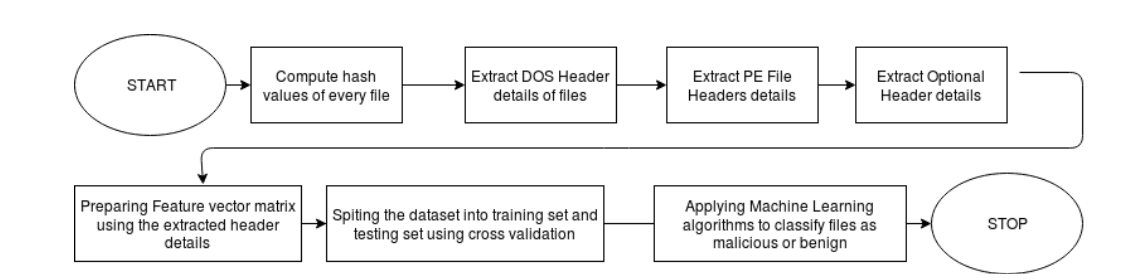
1. START
2. Compute hash values of every file and check whether any of the file in the corpus is duplicate and then remove the duplicate files.
3. Extract header details of the binaries with the help of PE File module functions of python for the analysis purpose.
4. The following header details are extracted

* DOS header
* PE File header
* Optional header

1. Prepare the feature vector matrix by selecting best features for training and testing purpose of the dataset.
2. With Cross-Validation split the dataset into training and testing set.
3. Apply ML algorithms to classify files as malwares and benign Algorithms used to evaluate:

* K-Nearest Neighbors
* Decision Trees
* Random Forest
* Logistic Regression
* SVM (Support Vector Machines)

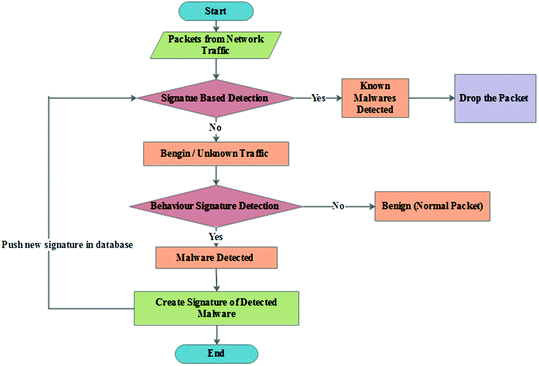
1. STOP



* 1. **Data/ Data Structure**
* **Static Analysis**: Decompiling, Parsing, Features Generations
* **Dynamic Analysis**: Emulation, Log Extraction, Feature Generation
* **Feature Selection**
* **ML and Malware Detection**: Model Building, Testing, Classification Report

This is the flowchart which shows how the file will be checked for it. First, its packets will be checked from the network traffic, then using the ML Models and algorithms, it which if it is malicious, if found malicious, it will drop that packet, else if it is not known, then will check its behavior and decide, else it will be normal to open.

This project will use libraries like Python PE file, hashlib, pandas, DOS Header in the program.



2.2.1 Flowchart

* 1. **SWOT Analysis**
* **Strengths**

The Support Vector machine and K-Nearest neighbor that are used to in the program, train the system which helps in providing higher accuracy when compared to other machine learning and deep learning models.

* **Weakness**

The model is quite heavy and requires a lot of processing power for testing stage. The available dataset for network trafﬁc are not very large, which affects the accuracy. In the project, we have not implemented the procedures for ﬁrewall protection, load description and handling, DDoS attacks etc. The rules for forwarding and dropping packets are also described, but are subject to human error.

* **Opportunities**

The work can be done in anomaly-based Network Intrusion Detection Systems (NIDS), ﬁrewall protection, load description and handling, and their respective Northbound and Southbound APIs. More research into this domain can help improve accuracy and handle various types of attacks.

* **Threats**

This project can only work with network packets. Adversarial malware attacks can bypass the discriminator classiﬁcation schema. Zero-day DDoS attacks can mostly cause network failure, as human errors could be present in ﬂow table handling.

* 1. **Project Features**

The application will allow the user to select the file from the database of the user.

Then, it will check and scan the file through different algorithms applied in it.

If the file is safe to use, it will show that the file can be opened, else will warn the user not to open the file as it contains a malicious apk file.

It requires an active internet connection of the user.

This project will help in reducing the phishing attacks, where the criminal sends malicious files through mails and the people get trapped.

This project will also help in avoiding the cross-site scripting attacks, where the criminal can send a malicious JavaScript code within the file.

It performs malware detection, where models will be trained on large corpus of executable using good set of discriminative predictors extracted through static analysis.

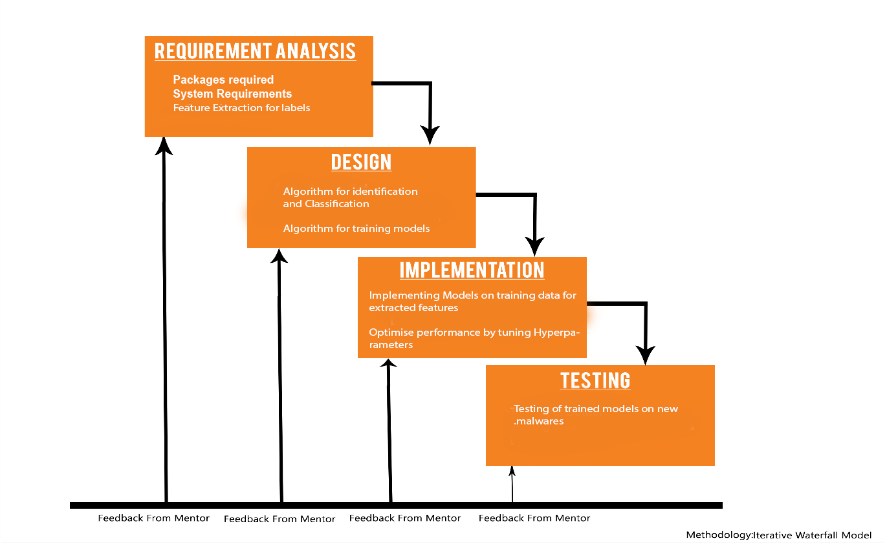
* 1. **Design and Implementation Constraints**

**Implementation Constraints:**

1. The model will only work on Windows Portable Executables (PEs). The number of available adversarial malwares is limited so it will restrict the robustness of the network.
2. The input test ﬁles will be correctly labeled, and there can be no misclassiﬁed PEs. Each input ﬁle will have a Windows API call. Processor and memory requirements are met beforehand.

**Design:**

* After the hash value for every file is computed.
* Header details are extracted.
* Applying the machine learning algorithms.
* We have classified Unseen Setup files as legitimate or malicious



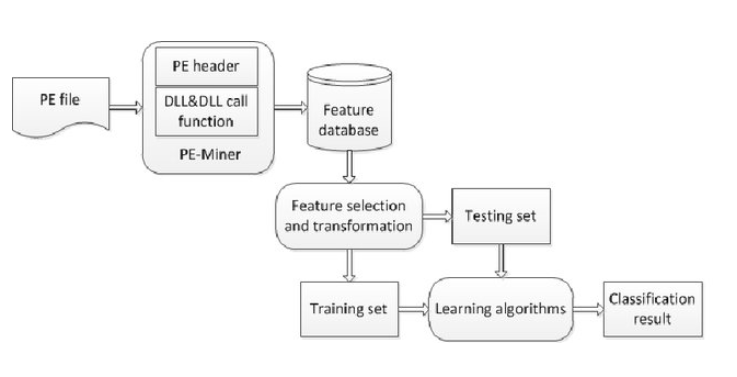
2.5.1 Waterfall Model

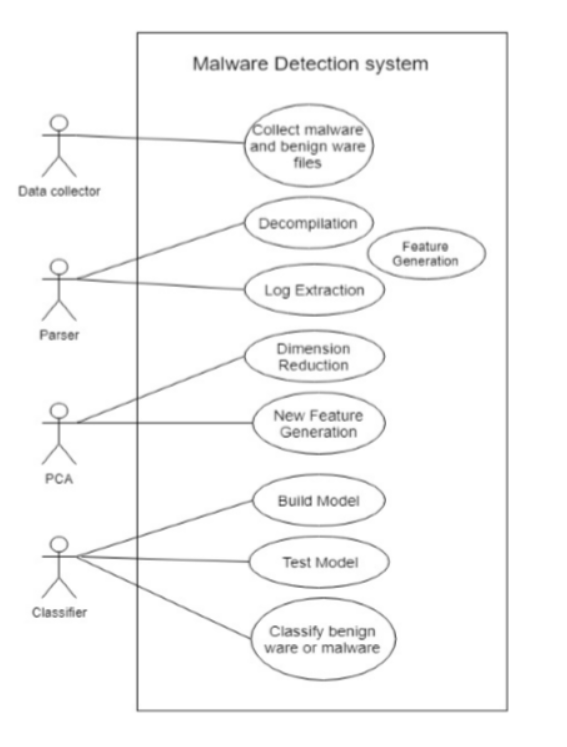
* 1. **Design Diagrams**

There are various UML Diagrams that we have created so as to explain the design and

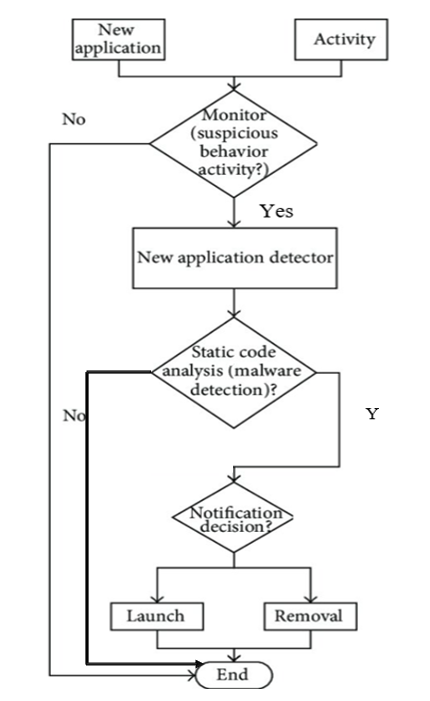
functionality of the project better.

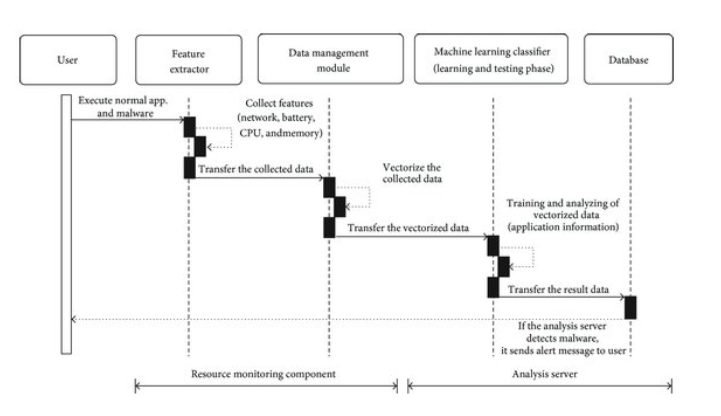
Steps:

1. Upload the file or the application to be checked for malware.
2. It will monitor the content in every file
3. If found suspicious, the detector will come into place and using the static code analysis method, will detect the malicious activity
4. Then will decide whether to give notification or not
5. Then will either launch it and tell the user, else will try to remove it if possible, and already present in the library.
6. If not found suspicious, the file is opened, and the process ends
7. END
   * 1. **Data Flow**
     2. **Use Case Diagram**



* + 1. **Activity Diagram**



* + 1. **Sequence Diagram**
  1. **Assumptions and Dependencies**

1. The basic assumptions includes that the user should know how to select a file before opening it.
2. The user should understand the output of whether the file is malicious or not; if malicious, what is that file name.
3. The project is dependent on the internet connectivity of the user. The project requires internet connection for multiple users.
4. **SYSTEM REQUIREMENTS**
   1. **User Interface**

Simple and easy to use system should be there, where the user is able to detect malware using the APK dataset. Machine learning is used to build the model using the selected features as input. Comparative analysis is done and a classiﬁcation report is generated.

* 1. **Software Interface**
* Programming Language: Python
* Operating System: Windows/Linux/Macintosh
* Dataset: APK dataset
  1. **Database Interface**
* Google Collaborator

1. **NON-FUNCTIONAL REQUIREMENTS**
   1. **Performance Requirements**

* The system should be based on web and has to run form a web server
* The system should take initial load time depending on internet connection strength of user which also depends on the media size uploaded on the application.
  1. **Security Requirements**
* The system must check for authorization.
* The system should check for malware file.
* The system must automatically warn the user if found malicious.
  1. **Software Quality Attributes**
* AVAILABILITY: Malware Detection will be for each user every time.
* CORRECTNESS: The detector should always connect to the correct malware apk file.
* MAINTAINABILITY: Detector should maintain its state even if any older or newer files is checked.
* USABILITY: Detector should satisfy a maximum number of users needs of having a malware free file.