APPLIED INFORMATION TECHNOLOGY AND COMPUTER SCIENCE
e-ISSN: 2773-5141

AITCS

Vol. 5 No. 2 (2024) 114-132
https://publisher.uthm.edu.my/periodicals/index.php/aitcs

Anti-Ransomware Tools to Detect Crypto Ransomware Based on Machine Learning Approach

Tan Sin Ming[1], Isredza Rahmi A Hamid[1]*

[1] *Fakulti Sains Komputer dan Teknologi Maklumat,*
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author: rahmi@uthm.edu.my
DOI: https://doi.org/10.30880/aitcs.2024.05.02.007

Article Info

Received: 15 July 2024
Accepted: 16 October 2024
Available online: 15 December 2024

Keywords

ransomware, ransomware detection, ransomware attack, machine learning

Abstract

Ransomware is harmful software that aims to block access to a computer system or files unless a certain amount of money is paid to the attackers, usually in cryptocurrency. Previous tools can scan and detect ransomware, report generation, and some extra features such as Virtual Private Network (VPN) with some payments. The problem with current tools is the high false positive rates. We proposed an Anti-Ransomware Tool that can effectively detect and address crypto-ransomware threats, primarily focusing on Tesla Crypt Ransomware. The tool is designed to overcome the common challenges existing in anti-ransomware tools, such as high false alarms that lead to a loss of trust, reduced productivity, and increased downtime costs due to false positives. Our tool adopts an object-oriented approach and employs a Random Forest (RF) algorithm to learn and identify patterns and behaviours indicative of ransomware threats. The development methodology aligns with the Agile Model. The intended users for this tool are adults who regularly use computers for various tasks on the Windows operating system. The outcomes of this project include the creation of a comprehensive tool capable of detecting and removing Tesla Crypt Ransomware, featuring a user-friendly interface with scanning, detection, and deletion modules, along with pop-up notifications and a summary module. The final developed Anti-Ransomware Tool could identify and differentiate between legitimate and ransomware files. It could also detect up to 14 various types of Crypto Ransomware and a significantly low 0.34% False Positive Rate.

1. Introduction

In recent years, ransomware has gained popularity as a tool that cybercriminals use to extort victims' money [1]. Ransomware is a type of malware that prevents users from accessing their systems or personal files by encrypting them [1]. It demands a ransom payment in exchange for the restoration of access. While some people may mistake it for a computer virus that has locked their system, it is important to understand that ransomware is typically classified as a separate type of malware rather than a virus [2].

Current issues in the field of anti-ransomware tools include false positives. These tools can often encounter false positives where legitimate software or system processes are mistakenly identified as behaving like ransomware. For instance, certain backup or synchronization tools that help users keep their files and data up to date and accessible across multiple devices or locations, such as Google Drive and Microsoft One Drive, innocently create file copies where it could potentially trigger false positives. Overly sensitive detection of encryption

processes can also lead to false positives, especially when legitimate encryption activities by applications or users are misinterpreted. The objectives of this project are as follows:

i. To design an Anti-Ransomware tool to detect Crypto Ransomware based on object-oriented approach.
ii. To develop an Anti-Ransomware tool to detect Crypto Ransomware by employing a Random Forest (RF) algorithm, and
iii. To implement the alpha and beta testing to the target user.

The developed anti-ransomware tool will provide the user a good experience compared to other existing tools since the false alarm will be low and without any surcharge. The anti-ransomware tool will be developed using Python programming language with the aid of the Agile model. The targeted users are all computer users who regularly use computers for various tasks. The tool will enable users to scan files on the device and detect ransomware with the involvement of a machine-learning algorithm to improve the accuracy of the result. The potential ransomware threat will then be deleted if detected by the tool. There will also be a pop-up notification to alert the user and a summary of scanning results to provide detailed information for the user.

The rest of the paper is organized as follows: Section 2 provides the literature review about ransomware and its type, ransomware detection techniques, and several machine learning algorithms. Some of the existing tools will also be compared and analyzed in section 2. Section 3 presents the methodology applied to the project while Section 4 explains the analysis and design of the anti-ransomware tool. Section 5 will include the discussion and testing results of the developed tool, and Section 6 will conclude the paper.

2. Related Work

Section 2 discusses ransomware, type of ransomware, techniques for detecting ransomware, machine learning algorithms, and existing anti-ransomware tools.

2.1 Ransomware

Ransomware is malicious software designed to block access to a computer system or files until a specific amount of money is paid to the attackers, typically in the form of cryptocurrency [7]. Unfortunately, in recent times, ransomware attacks have become more frequent and sophisticated, affecting individuals, businesses, and even governmental institutions [7]. The global survey 'The State of Ransomware 2021' commissioned by Sophos announced in its findings that, among roughly 2000 respondents whose organizations had been hit by a ransomware attack, the average total cost to an organization to rectify the impacts of a ransomware attack (considering downtime, people time, device cost, network cost, lost opportunity, ransom paid etc.) was US$1.85 million, which is more than double the US$761,106 cost reported in 2020 [7]. Understanding how ransomware works is essential for raising cybersecurity awareness, and detecting ransomware promptly allows quick action to mitigate the damage, prevent further encryption of files, and safeguard sensitive data. With the potential for significant financial and reputational losses, the ability to detect and prevent ransomware attacks is a critical aspect of modern cybersecurity strategies.

2.2 Type of Ransomwares

In the world of cybersecurity, there are several types of ransomwares that pose significant challenges. One notable form is crypto ransomware, which is a highly disruptive threat due to its widespread occurrence, financial motivation, and high-profile incidents. It is a type of ransomware that restricts access to computer systems and encrypts important files, forcing victims to pay a ransom in exchange for decryption keys [8]. Locker ransomware is different where it blocks access to entire computer systems, disrupting essential functions beyond file encryption [9]. The rise of Ransom-as-a-Service (RaaS) is also concerning, as it allows people without advanced technical skills to purchase ransomware as a service on the dark web [10]. Doxware is a more dangerous variant that threatens to release sensitive information unless a ransom is paid, exploiting privacy concerns for financial gain [11]. Lastly, scareware uses deceptive pop-up messages to scare users into purchasing fake security software or disclosing personal information [12]. These variations in ransomware demonstrate the ever-changing nature of cyber threats. It is essential to have comprehensive cybersecurity strategies that address technical vulnerabilities, socio-economic factors, and the importance of digital literacy to counter deceptive manipulation.

2.3 Ransomware Detection Technique

There are different strategies employed to detect and protect computer systems from ransomware attacks. One commonly used method is signature-based detection, which relies on unique signatures or patterns linked to known ransomware strains, acting as digital fingerprints [3]. However, this approach may encounter difficulties when dealing with new strains that exhibit polymorphic behavior. On the other hand, behavior analysis dynamically monitors software actions, detecting deviations from normal behavior without predefined patterns [4]. Although it is effective against new threats, it may produce false positives. Anomaly analysis, a proactive

technique, identifies ransomware by examining deviations from expected behavior and adapts to the ever-evolving cyber threat landscape [5]. Yet Another Recursive Acronym (YARA rules) provides a versatile approach that allows analysts to create custom rules for specific ransomware patterns, ensuring adaptability against rapidly evolving variants [6]. Each method contributes distinctively to ransomware defense, highlighting the importance of a multi-layered approach for comprehensive cybersecurity.

2.4 Machine Learning Algorithm

Machine learning algorithms are an essential component in the field of cybersecurity, especially in detecting ransomware. Different types of machine learning algorithms, such as Support Vector Machine (SVM), Decision Trees, Random Forests, and Neural Networks, offer various strengths in identifying ransomware behavior. SVM is suitable for classification tasks and can distinguish normal activities from ransomware behavior by creating a decision boundary [13]. Decision Trees provide transparency and interpretability, enabling the identification of potential ransomware threats through a structured decision-making process [14]. Random Forests use ensemble learning to consolidate decisions from multiple decision trees, improving accuracy and addressing overfitting [15]. Neural Networks can handle complex patterns within data, prioritize features associated with ransomware activities, and are inspired by the human brain [16]. These machine learning algorithms together form a robust cybersecurity toolkit capable of identifying and mitigating continuously evolving and sophisticated ransomware threats.

2.5 Existing Anti-Ransomware Tools

This section discusses various anti-ransomware tools such as Virus Total [17], F-Secure Online Scanner [18], and Norton Power Eraser [19].

2.5.1 VirusTotal

VirusTotal is a crucial tool for studying and detecting ransomware, providing a comprehensive approach by allowing simultaneous file scans using multiple antivirus engines [20]. It is a web-based tool that needs an internet connection to operate. So, it will not provide real-time protection if the user is not connecting to the network. VirusTotal does not provide scanning for user's devices instead this tool does provide file reputation scanning. It could generate reports like others but not alert users as it does not have a pop-up notification function. VirusTotal did employ a machine learning algorithm in the tool, but it did not define the type of algorithm used in their website. The types of ransomwares that could be detected are crypto ransomware, locker ransomware, and scareware.

2.5.2 F-Secure Online Scanner

The F-Secure Online Scanner is a ransomware detection tool that allows users to scan systems and identify potential threats [18]. This tool is developed by F-Secure, a leading cybersecurity company, this tool is easy to use and provides a reliable solution to detect ransomware in critical areas. It has a user-friendly interface that individuals and organizations can use to safeguard their systems against ransomware attacks. The tool uses its own engine for ransomware detection and will generate reports for the scanning result. It could scan for the user's device but does not check for file reputation. F-Secure Online Scanner cannot operate without an internet connection and will not provide real-time protection. The tool also employed various machine learning algorithms, but it does not mention the type of algorithms used. The tool could not alert the user as it does not have a pop-up notification function. The types of ransomwares that could be detected are crypto ransomware, locker ransomware, and scareware.

2.5.3 Norton Power Eraser

Norton Power Eraser is a powerful tool in the field of ransomware detection developed by Norton, a well-known name in cybersecurity [19]. It is designed to scan and remove potential threats from users' systems. Norton Power Eraser uses aggressive scanning techniques to identify deeply embedded malware, including ransomware, that might go unnoticed by traditional scans. Its proactive approach and capability to target persistent and hidden threats make it highly effective in the ever-changing ransomware landscape. It is a desktop application that scans devices using its own engine. The tool could generate reports for the scanning result, alert users to potential threats, and provide file reputation checking. It could provide real-time protection as it does not need an internet connection to operate. Norton Power Eraser employed various machine algorithms but did not specify which type they had used in the tool. The types of ransomwares that could be detected are crypto ransomware, locker ransomware, and scareware.

2.6 Comparison of Existing Tools with The Proposed Tool

Table 1 shows the comparison between existing anti-ransomware tools such as VirusTotal, F-Secure Online Scanner, Norton Power Eraser, and the proposed tool. The proposed tool shares some similarities with the F-Secure Online Scanner and Norton Power Eraser. All of these tools have the ability to scan devices for malware using their own detection engines. However, VirusTotal and Norton Power Eraser have a unique feature as they provide file reputation information, which the other tools lack of. The proposed tool can function without an internet connection but cannot offer real-time protection.

Table 1 *The Comparison of Existing Tools with the Proposed Tool*

Tools / Features	VirusTotal	F-Secure Online Scanner	Norton Power Eraser	Anti-Ransomware Tool
Platform	Web-based	Web-based	Desktop Application	Desktop Application
Scanning for devices	No	Yes	Yes	Yes
Uses own engine for malware detection	No	Yes	Yes	Yes
File reputation scanning	Yes	No	Yes	No
Report Generation of scanning result	Yes	Yes	Yes	Yes
Operate without an internet connection.	No	No	Yes	Yes
Real-time protection	No	No	Yes	No
Machine Learning Algorithm	Yes (Not Defined)	Yes (Not Defined)	Yes (Not Defined)	Yes (Random Forest (RF))
Popup notification	No	No	Yes	Yes
Types of Ransomware detected	Crypto Ransomware, Locker Ransomware, and Scareware	Crypto Ransomware, Locker Ransomware, and Scareware	Crypto Ransomware, Locker Ransomware, and Scareware	Crypto Ransomware

3. Methodology

We considered using the Agile Model as the project's methodology for this project. The Agile Software Development Life Cycle (SDLC) is a modern and flexible methodology that focuses on collaboration, adaptability, and responsiveness in software development. This methodology consists of several key phases, such as Requirement Gathering and Analysis, Designing, Implementation, Testing, Deployment, and Feedback. These phases ensure the Anti-Ransomware Tool is effective, user-friendly, and responsive to the ever-changing cybersecurity environment. All the phases below will be run through several iterations before the final product is produced. Fig. 1 shows the six phases of Agile SDLC.

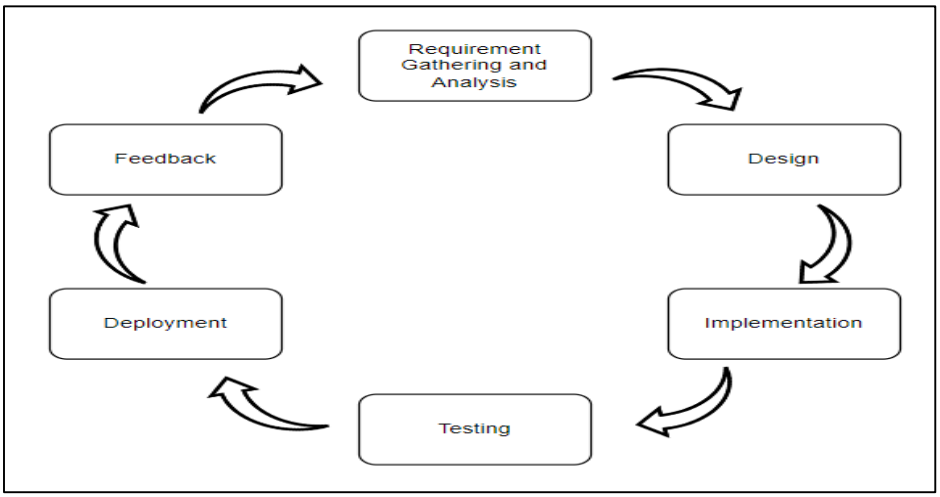

Fig. 1 *Phases in Agile Software Development Life Cycle (SDLC)*

3.1 Requirement Gathering and Analysis Phase

The requirement gathering and analysis phase is the project's initial phase, where all the requirements and problems related to detecting Crypto Ransomware are defined and analyzed. In this phase, we researched existing anti-ransomware tools, comparing their features and shortcomings to establish a clear problem statement. We engaged with potential end-users such as UTHM FSKTM students and daily computer users to help us identify essential features and functionalities for the Anti-Ransomware Tool. All the research sources are from journal articles and conference papers on ransomware and its detection using machine learning algorithms. The outcome of this requirement gathering phase was a proposal that included problem statements, objectives, scope, and expected outcomes of this project.

3.2 Design Phase

The Design phase involved creating a robust architecture that outlines the tool's functionality and interactions with users and systems for the Anti-Ransomware Tool. This included outlining the tool's modules: scanning, detecting, deletion, pop-up notifications, and summary reports. We also define the tool's user, functional, and non-functional requirements. We selected Python as the programming language due to its extensive libraries and ease of implementation. The design also incorporated machine learning algorithms, particularly Random Forest (RF), chosen for its accuracy and randomness in detecting complex ransomware behaviours. We also declare the machine learning process and the features that will be extracted. The outcome of the phase was all the requirements stated and the tool interface design with the aid of Unified Modeling Language (UML) Diagrams such as the use case diagram, activity diagram, and sequence diagram.

3.3 Implementation phase

In the implementation phase, we wrote the source code for the anti-ransomware tool based on the design specifications, which include implementing the identified features and functionalities mentioned in the requirement gathering and analysis phase. The coding was written in PyCharm Community Edition 2023.3.2. The software requirements are Windows Operating System and Microsoft Visual Studio Code, while the hardware specification is an Intel 11th Gen i5 central processing unit with 8.00GB of Random Access Memory (RAM). We implemented machine learning by extracting features from the datasets that include ransomware and non-ransomware samples, such as FileName, md5Hash, Machine, DebugSize, DebugRVA, MajorImageVersion, MajorOSVersion, ExportRVA, ExportSize, IatVRA, MajorLinkerVersion, MinorLinkerVersion, NumberOfSections, SizeOfStackReserve, DllCharacteristics, ResourceSize, BitcoinAddresses, and Benign. The data was split into training and testing sets with a ratio of 80:20 to train the machine learning model, with the best-performing Random Forest (RF) model integrated into the tool. The outcome of this phase was the development of an anti-ransomware tool that has the capability to scan, detect, delete, and provide a summary of the scanning result and a performance-trained machine learning model.

3.4 Testing Phase

In the testing phase, we implemented the test plan to validate the tool's functionality under various scenarios and simulate the potential real-world cyber threat. This involves assessing the tool's ability to detect Crypto Ransomware and differentiate non-ransomware files, evaluating its response to the attack. We tested 15 types of Crypto Ransomware and 5 legitimate files on our tool. We also ran through each button in the Anti-Ransomware Tool to ensure that all the buttons were functioning. The tool was also assessed for seamless operation within the Windows operating system, and any errors encountered were resolved through iterative testing and debugging processes to enhance the tool's overall performance. The outcome of this phase was no significant error in the developed tool.

3.5 Deployment Phase

The deployment phase focuses on delivering the Anti-Ransomware Tool to potential users such as UTHM FSKTM students and daily computer users. We provided clear instructions for the installation and configuration of the tool to ensure no environmental issues for the Anti-Ransomware Tool. In this phase, we found some unforeseen issues during the testing phase. The iterative process allowed us to adjust and optimize the tool's performance in real-world scenarios. The outcome of this phase was the adjustment for any unforeseen errors in the developed tool.

3.6 Feedback Phase

During the feedback phase, we encouraged potential users to share their experiences after using the Anti-Ransomware Tool. We distributed user acceptance testing forms using Google Form to 18 respondents, including

Applied Information Technology and Computer Science Vol. 5 No. 2 (2024) p. 114-132

UTHM FSKTM students and daily computer users, to gather detailed feedback. Then, we analyzed the results and suggestions provided by the user. This iterative feedback process ensured that the tool remained responsive to the dynamic nature of cyber threats and evolving user needs, resulting in a highly effective and user-approved Anti-Ransomware Tool.

4. Analysis and Design

This section explains the components of the analysis and design phase. This phase includes the user, functional and non-functional requirements, system architecture, machine learning process, file feature extraction, and the Unified Modelling Language (UML) diagram.

4.1 System Architecture

Fig. 2 shows the Anti-Ransomware Tool's system architecture [21] that outlines the key modules, including the scanning functionality, ransomware detection module, delete functionality, reporting and summary features, and the interactions within the system. The architecture revolves around the core scanning functionality, which can be initiated through three different scanning modes: Full Scan, Quick Scan, and Selective Scan. Once a scanning mode is selected, the tool begins scanning and triggers the detection module, which employs Random Forest (RF) machine learning. When a threat is identified, it will be deleted immediately. After the scanning and threat removal processes are completed, the tool generates a scanning result summary. All the results will be stored in a text (.txt) file, which serves as the local database. Users can access their scanning history and results through the "Summary" module, which retrieves data from the text file and presents it in a user-friendly format.

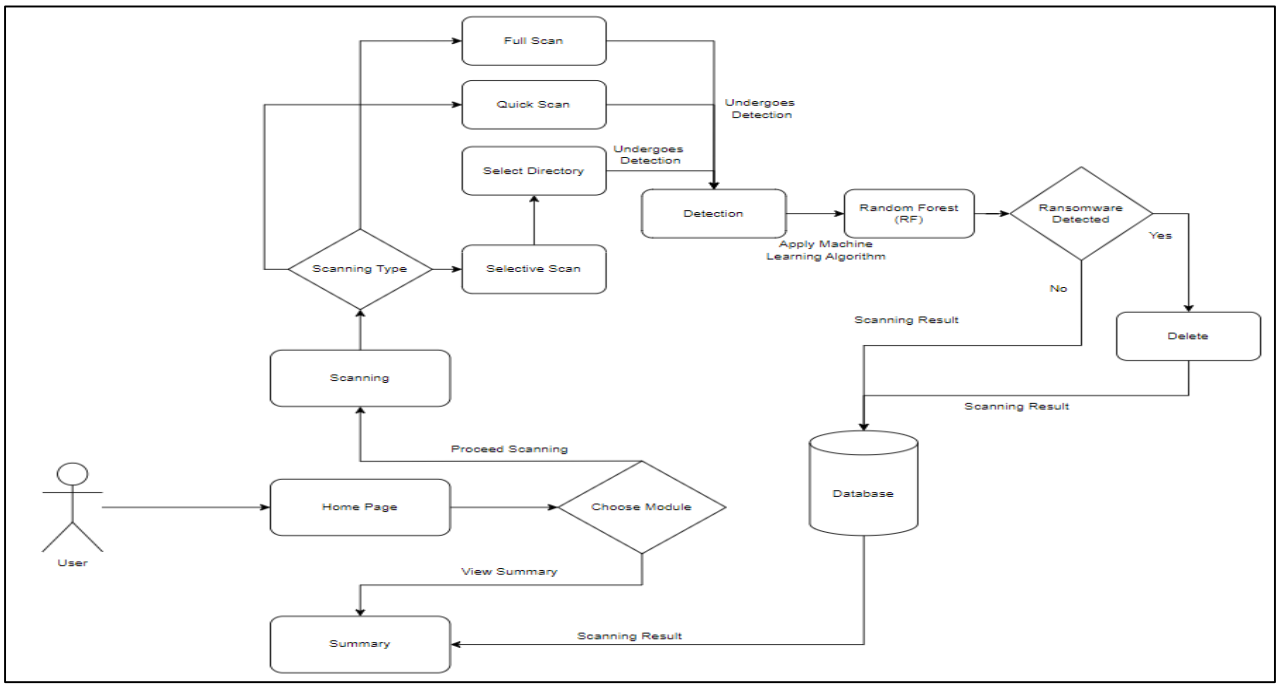

Fig. 2 *System Architecture of Anti-Ransomware Tool*

4.2 Machine Learning Process

Fig. 3 shows the machine learning process for the training model purpose for the Anti-Ransomware Tool. The dataset will initially be split into training and testing sets during the data preprocessing phase. The training set is the portion of the dataset used to train the machine learning model, which usually contains input features and corresponding target labels. Usually, the training set comprises the majority of the dataset. The testing set, or the validation set, is a separate portion of the dataset used to evaluate the trained model's performance. It contains input features similar to the training set, but the model has not seen these specific examples during training. The ratio of the training set to the testing set is 80:20, a common split used in machine learning. It means that 80% of the dataset is allocated to the training set, while the remaining 20% is used for the testing set. Then, we will undergo feature extraction for both sets of data and proceed to train the Random Forest (RF) model. After that, we will evaluate and test the model with a few iteration processes and produce the best model for the detection process.

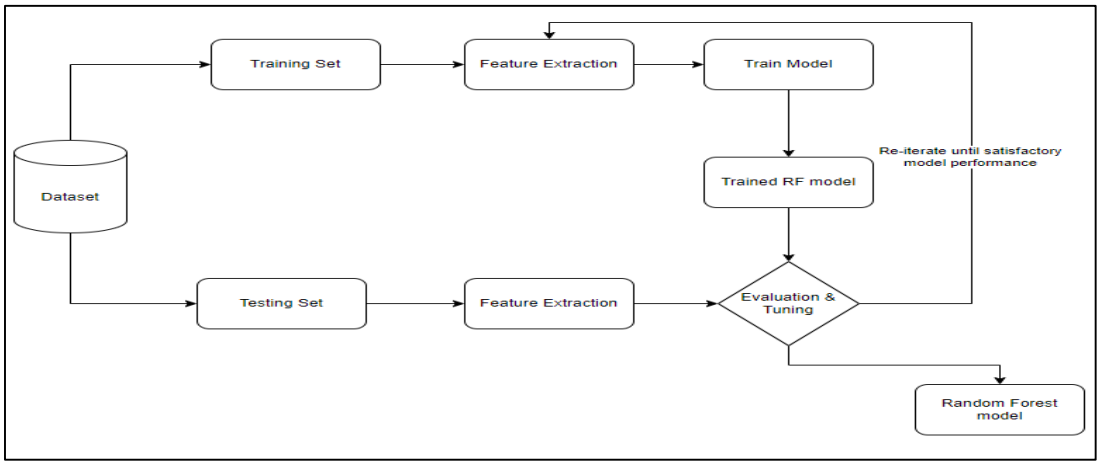

Fig. 3 *Machine learning process on training model of the Anti-Ransomware Tool*

4.3 File Feature Extraction

Table 2 describes each feature we extracted from a file corresponding to the chosen dataset. The features such as FileName, md5Hash, Machine, DebugSize, DebugRVA, MajorImageVersion, MajorOSVersion, ExportRVA, ExportSize, IatVRA, MajorLinkerVersion, MinorLinkerVersion, NumberOfSections, SizeOfStackReserve, DllCharacteristics, ResourceSize, BitcoinAddresses, and the target variable, Benign are explained in the table.

Table 2 *Feature Extraction for Ransomware Detection*

Features	Description
FileName	The file's name is directly obtained from the file system or metadata of the executable. This feature is not typically considered as an informative or discriminative feature.
md5Hash	The MD5 hash of the file is a unique fingerprint of the file's content. MD% hash is calculated by applying the MD5 hash function to the file's binary data using hashlib in Python. This feature is not typically considered as an informative or discriminative feature.
Machine	The type of target machine (architecture) for which the file is intended, such as x86, x64, or ARM. This feature could be extracted from the Portable Executable (PE) header using a library like pefile in Python.
DebugSize	Size of the debug information. This feature could be extracted from the PE header.
DebugRVA	Relative Virtual Address (RVA) of the debug information. This feature could be extracted from the PE header.
MajorImageVersion	The major version number of the image indicates the version of the executable. This feature could be extracted from the PE header.
MajorOSVersion	The major version number of the required operating system for running the executable file. This feature could be extracted from the PE header.
ExportRVA	RVA of the export table, which contains information about exported functions. This feature could be extracted from the PE header.
ExportSize	Size of the export table. This feature could be extracted from the PE header.
IatVRA	RVA of the Import Address Table (IAT) contains addresses of imported functions. This feature could be extracted from the PE header.
MajorLinkerVersion	The major version number of the linker is used to build the executable. This feature could be extracted from the PE header.
MinorLinkerVersion	The minor version number of the linker is used to build the executable. This feature could be extracted from the PE header.
NumberOfSections	The number of sections present in the PE file. This feature could be extracted from the PE header.
SizeOfStackReserve	The size of the stack to reserve in memory for the executable. This feature could be extracted from the PE header.
DllCharacteristics	Characteristics of flags associated with the DLL (Dynamic Link Library) file. This feature could be extracted from the PE header.

Table 2 *Feature Extraction for Ransomware Detection (cont)*

Features	Description
ResourceSize	The size of the resource section in the PE file contains data such as icons, images, and strings. This feature could be extracted from the PE header.
BitcoinAddresses	The presence of Bitcoin addresses within the file can indicate ransomware demanding payment. This could be obtained by searching for patterns resembling Bitcoin addresses within the file's binary data or strings.
Benign	A label indicating whether the file is benign (0) or malicious (1), serving as the Target Variable for classification. It is assigned based on pre-existing knowledge or labeling of the dataset, often obtained through manual analysis or reputable sources

4.4 User Requirement

Table 3 describes the requirements of the users. The elements such as an intuitive user interface, user-friendly design, clear scanning process, customization scanning options, compatibility, prompt notification, and summary module are explained in the table.

Table 3 *User Requirements for Anti-Ransomware Tool*

User Requirement	Description
Intuitive User Interface	Users should experience an intuitive interface in the tool for ease of use.
User-Friendly Design	Users prefer a design that is easy to navigate and understand.
Clear Scanning Process	Users will gain a clear and complete understanding of the scanning process.
Customization Scanning Options	Users can customize scanning preferences within the tool, such as selecting a specific directory to perform scanning.
Compatibility	The tool must be compatible with the user's systems and configurations.
Prompt Notification	Users will be notified immediately if there are any detected ransomware threats.
Summary	Users can receive detailed information on the scanning results.

4.5 Functional Requirement

Table 4 explains the functions for each module and components in the proposed tool. The main modules found in the Anti-Ransomware Tool are the scanning module, detection module, delete module, pop-up notification, and reporting and summary module. Each module and component will be explained in the table.

Table 4 *Functional Requirements for Anti-Ransomware Tool*

Functional Requirement	Description
Scanning Functionality	The tool is enabled to scan the entire system or partially for potential ransomware.
Ransomware Detection Module	The tool must be developed with a robust module that is capable of identifying and detecting crypto ransomware.
Delete Functionality	The tool will be able to remove the detected potential ransomware threat.
Machine learning Integration	The tool will integrate machine learning algorithms to train a model for enhancing the tool's ability to recognize and respond to potential threats.
Reporting and Summary	The tool will include features such as reporting and summary to show results and details of the scanning results.
Customizable Scanning Options	The tool will allow users to customize scanning frequency based on their needs.
Pop-up Notification	The tool will include a module that will alert users about detected ransomware threats.

4.6 Non-Functional Requirement

Table 5 describes the non-functional requirements for the Anti-Ransomware Tool, which will include the minimal requirement for the tool through aspects such as performance, usability, reliability, security, scalability, and compatibility.

Table 5 *Non-Functional Requirements for Anti-Ransomware Tool*

Non-Functional Requirement	Description
Performance	The tool would operate efficiently and respond to potential ransomware threats.
Usability	The tool's interface should be user-friendly and require minimal practice.
Reliability	The tool must reliably detect and remove ransomware without any error.
Security	The tool must have robust security measures to protect user data and tool integrity.
Scalability	The tool should scale effectively to adapt to the increased usage.
Compatibility	The tool should be compatible with various hardware configurations and environments.

4.7 Use Case Diagram

The Use Case Diagram is part of the Unified Modeling Language (UML) that visually displays the interactions between users and the tool [22]. Fig. 4 visualizes the use case diagram of the user. Users can perform actions such as Scanning, Choose Scanning Options, and receive Pop-up Notifications and summaries.

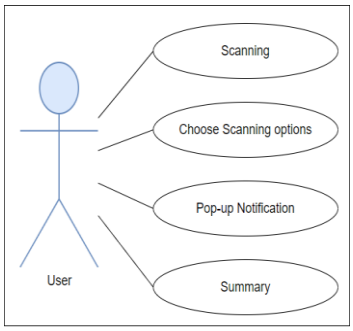

Fig. 4 *Use Case Diagram of Anti-Ransomware Tool*

4.8 Activity Diagram

The Activity Diagram is a visual representation of the dynamic aspects of the project. It illustrates the flow of activities among different components from the essential pages such as the "Home Page," "Scanning Page," and "Summary Page". The user could initiate a scan and access the summary module through the "Home Page". The user could also choose which type of scanning they are willing to perform, either a quick, full, or selective scan on the device. The activities on the "Summary Page" will involve presenting a comprehensive overview of the scanning results. Fig. 5 illustrates the user activity diagram of the Anti-Ransomware Tool.

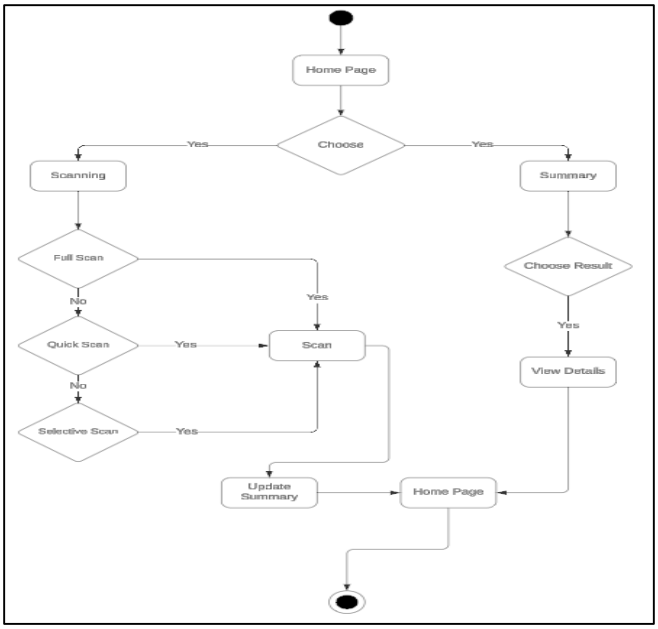

Fig. 5 *User Activity Diagram of Anti-Ransomware Tool*

4.9 Sequence Diagram

Applied Information Technology and Computer Science Vol. 5 No. 2 (2024) p. 114-132

The Sequence Diagram in UML provides a detailed representation of interactions between the components of the Anti-Ransomware Tool. Fig. 6 illustrates the sequence of actions of the user interacting with the tool. Initially, the user interacts with the “Home Page” to initiate a scan. The system then triggers the scanning process, activating the “Scanning Page,” where the user can customize scanning options. The user could also interact with the “Home Page” to proceed to the “Summary Page,” where the user could view the details of the scanning result.

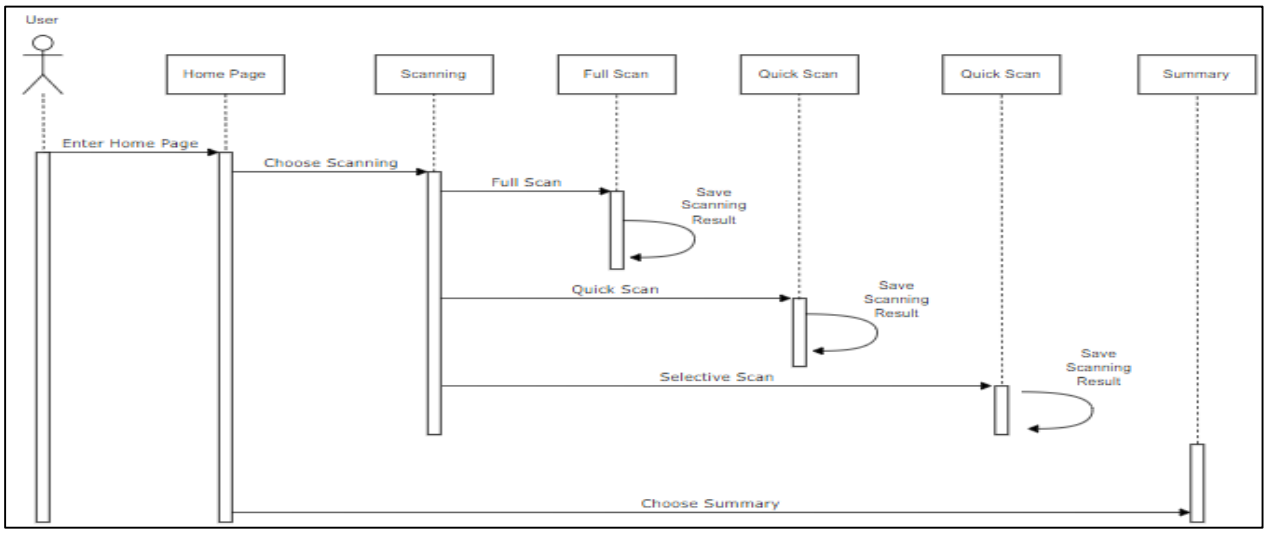

Fig. 6 *User Sequence Diagram of Anti-Ransomware Tool*

4.10 Design Interface

User Interface Design focuses on the visual and interactive elements of the Anti-Ransomware Tool, ensuring a user-friendly and intuitive experience. It involves creating three key pages, which are the "Home Page," “Scanning Page,” and “Summary Page.”

4.10.1 Home Page

Fig. 7 displays the homepage of the developed tool, which is the initial screen that users will encounter. The home page is the central hub of the Anti-Ransomware Tool, offering users a clear and intuitive interface to access its key features. The navigation bar at the left of the page allows users to easily switch between the home, scanning, and report pages, ensuring simple navigation throughout the application. When users click the "Scan" button, they are directed to the scanning page, where they can begin analyzing their system for any ransomware-related issues. The "Report" button, located next to the "Scan" button in the welcome section, provides users with access to a summary of previous scans and any detected ransomware threats.

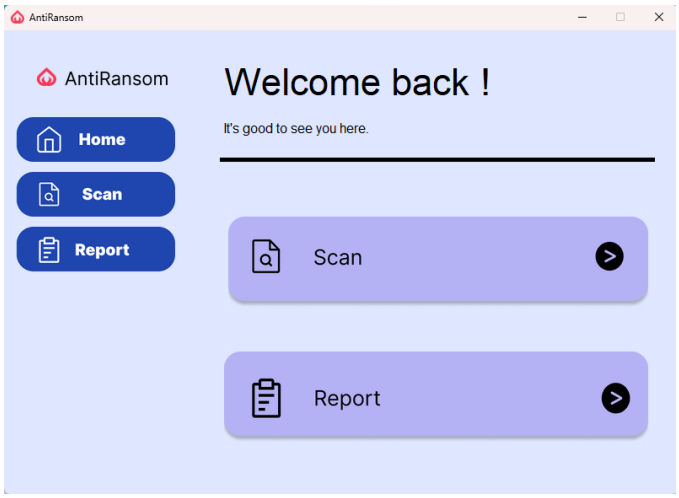

Fig. 7 *Home Page of Anti-Ransomware Tool*

4.10.2 Scanning Page

Fig. 8 shows the scanning module page of the developed tool. This module enables the user to perform quick, full, or selective scans by clicking each specific button. Upon selecting a scanning mode, the Anti-Ransomware Tool initiates the scanning process, utilizing a machine learning algorithm such as Random Forest (RF) to identify potential ransomware threats. During the scan, the tool thoroughly examines files and directories, using the trained Random Forest model to predict whether they are ransomware or not. If a threat is detected during the scan, the tool immediately alerts the user and permanently deletes the specific ransomware files.

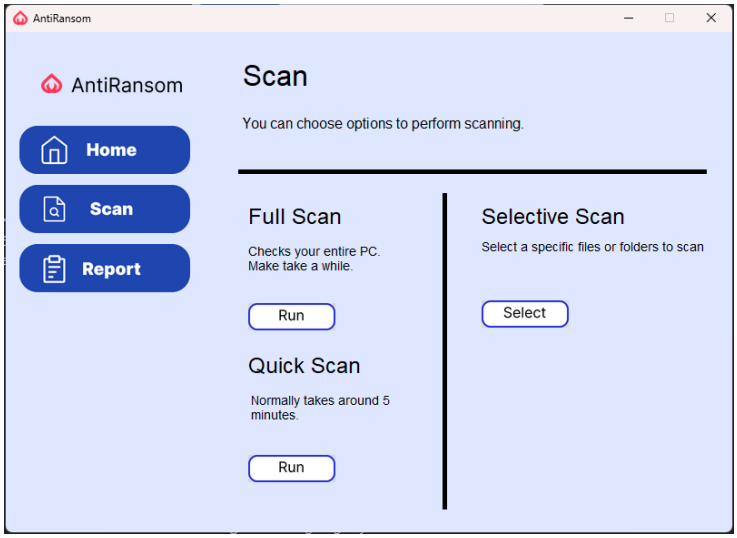

Fig. 8 *Scanning Page of Anti-Ransomware Tool*

4.10.3 Summary Page

Fig. 9 shows the summary module page of the developed tool. On this page, the user can easily view all the scanning history that has been done. The page presents a table that chronologically lists all previous scans performed by the user, along with relevant details for each scan, including the date and time when each scan was initiated, the type of scan performed (Full Scan, Selective Scan, or Quick Scan), the total number of files examined during each scan, and the outcome of each scan, indicating whether any threats were found. This clear and organized presentation of information allows users to easily review their scanning history and identify patterns or trends in their system's security status. For example, if a user notices an increasing number of ransomware threats detected over time, they may need to take additional measures to secure their system, such as updating their operating system and software or changing their browsing habits.

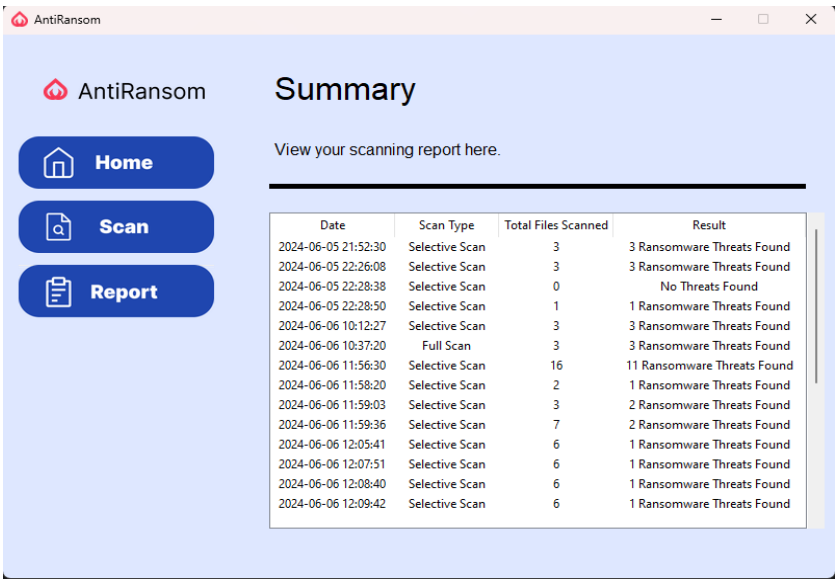

Fig. 9 *Summary Page of Anti-Ransomware Tool*

4.11 Test Plan

Table 6 displays the test plan of the Anti-Ransomware Tool. The test plan will be done in the Result and Discussion part after the development of the tool. The Test Plan serves as a guide for validating and ensuring the effectiveness of the Anti-Ransomware Tool in a systematic manner.

Table 6 *Test Plan of Anti-Ransomware Tool*

No.	Test List	Description	Actual Result
1	Home Page Functionality Test	Verify the proper functioning of the features accessible from the Home Page.	Pass / Fail
2	Scanning Process Test	Evaluate the initiation and monitoring of the scanning process on the Scanning Page.	Pass / Fail
3	Summary Page Test	Evaluate the accuracy and completeness of information presented on the Summary Page	Pass / Fail
4	Usability Test	Evaluate the overall user experience, such as navigation.	Pass / Fail
5	Performance Test	Assess the tool's response time and efficiency under various scenarios.	Pass / Fail
6	Compatibility Test	Ensure the tool's compatibility with different hardware configurations and environments.	Pass / Fail

5 Result and Discussion

This section explains the proposed tool's implementation, deployment, testing, and feedback phases. It includes the algorithm and result of feature extraction of a scanned file and model training, scanning results, tool testing, and verification. We used Python as the main programming language in PyCharm Community Edition 2023.3.2 during the implementation phase. Test Plan and User Acceptance Testing are checked in the testing, deployment and feedback phases.

5.1 Implementation and Result

The tool implementation phase includes the process and results of feature extraction and model training for ransomware detection. This section will also explain the feature extraction of the scanned file during the scanning process, the Random Forest algorithm, and the training result.

5.1.1 Feature Extraction of Scanned File During Scanning Process

Fig. 10 presents the pseudocode of the feature extraction process implemented during the scanning of files in directories. The pseudocode describes a function called extract_features that takes a file_path parameter. It starts by initializing a features dictionary with the file path, then reads the file's binary content, calculates its MD5 hash, and then stores it in the features dictionary. The function then attempts to parse the file as a PE (Portable Executable) file using the pefile library, extracting all the features mentioned in the Analysis and Design Phase from the PE headers, such as machine info, debug data, version info, export, and import data, linker info, section info, stack reserve size, DLL (Dynamic Link Library) characteristics, and resource size. It also searches for Bitcoin addresses in the file content using a regular expression and stores the result as a boolean in the features dictionary. If a PEFormatError is encountered during parsing, the function returns None. Finally, it closes the PE file handle and returns the features dictionary.

```
function extract_features(file_path):
    Initialize features dictionary with 'FileName' as file_path
    Open file at 'file_path' in binary read mode
    Read content of 'file' into file_content
    Calculate MD5 hash of file_content and store in features as 'md5Hash'

    Attempt to parse 'file' as a PE file:
        Initialize 'pe' object with the file content.

        Extract and store following features from PE headers:
            - Machine
            - DebugSize and DebugRVA from the IMAGE_DIRECTORY_ENTRY_DEBUG
            - MajorImageVersion
            - MajorOSVersion
            - ExportRVA and ExportSize from the IMAGE_DIRECTORY_ENTRY_EXPORT
            - IatVRA from IMAGE_DIRECTORY_ENTRY_IAT
            - MajorLinkerVersion and MinorLinkerVersion
            - NumberOfSections
            - SizeOfStackReserve
            - DllCharacteristics
            - ResourceSize from IMAGE_DIRECTORY_ENTRY_RESOURCE

        Compile a list of Bitcoin addresses using regular expression
        Store the result as a boolean in features with key 'BitcoinAddresses'

    if PEFormatError:
        return None

    Close the PE file handle

    return features
```

Fig. 10 *Pseudocode of Feature Extraction during the scanning process*

Fig. 11 shows the extracted feature results in the Comma-Separated Values (CSV) file. The extracted features will then be input into the trained Random Forest (RF) model for prediction.

	FileName	md5Hash	Machine	DebugSize	DebugRVA	MajorImageVersion	MajorOSVersion	ExportRVA	ExportSize	IatVRA	MajorLinkerVersion
2	0124e21d-018c-4ce0-92a3-b9e205a76bc0.dll	79755c51e413ed3c6be4635fd729a6e1	332	0	0	0	4	0	0	8192	8
3	05c8318f98a5d301d80000009c316005.vertdll.dll	95e19f3657d34a432eada93221b0ea16	34404	84	121728	10	10	126576	4930	0	14
4	06054fba-5619-4a86-a861-ffb0464bef5d.dll	85c32641d77a54e19ba8ea4ab305c791	332	0	0	0	4	0	0	8192	8
5	075822ac99a5d301660400009c316005.adhapi.dll	62e3b959d982ef534b66f819fe15f085	34404	84	19904	10	10	21312	252	18160	14
6	090607dd9ba5d301ca0900009c316005.SensorsNativ	ae38c5f7d313ad0ff3bfb88264767 67f	34404	84	97728	10	10	105792	1852	70592	14
7	0aedb43f9ba5d3014e0600009c316005.wlanapi.dll	28c98e000d02f8bfd9d050eb4e4c5d2e	34404	84	319776	10	10	374944	9208	312608	14
8	0bc194f9-b102-4833-85bd-603e216a9274.dll	706463ecc8e48e9e3a2a12a0cfcb8858	332	0	0	0	4	0	0	8192	8
9	0c7f9fdc9ba5d301c60900009c316005.wscsvc.dll	39da352fad220e83ce64de8dccb9736b	34404	84	197888	10	10	229024	112	187208	14
10	1.0.154_chromesetup_154_59.exe	e11e70ba243800626d17e3ffa6c9fb71	332	28	4240	0	4	0	0	4096	8
11	1035f45d9ca5d3015b0a00009c316005.srclient.dll	a5a106d5d03e6e59b0282e72bc9420f1	34404	84	64704	10	10	67632	404	57648	14
12	109b6ed1-e133-407f-b839-3fdd1ebc0d85.dll	212fdb291e0b5d8f34771bc1338c20b3	332	28	59496	0	4	0	0	8192	48
13	112ff2d99ca5d301590b00009c316005.mscoreei.dll	e313747a38c6bc267c9167d92bd2cb7f	34404	84	401484	10	6	418800	3784	405504	12

L	M	N	O	P	Q
MinorLinkerVersion	NumberOfSections	SizeOfStackReserve	DllCharacteristics	ResourceSize	BitcoinAddresses
0	3	1048576	34112	672	0
10	8	262144	16864	1024	0
0	3	1048576	34112	672	0
10	6	262144	16736	1040	0
10	7	262144	16736	1096	0
10	7	262144	16736	2072	0
0	3	1048576	34112	672	0
10	7	262144	16736	1328	0
0	3	1048576	256	8799820	0
10	6	262144	16736	1072	0
0	3	1048576	34112	1064	0
10	7	1048576	16736	1952	0

Fig. 11 *Feature Extraction result in CSV file*

5.1.2 Model Training

Fig. 12 presents the pseudocode of the machine learning process. It will start off with data preprocessing, in which some columns of data in the dataset CSV file will be dropped, and the Benign column will be set as the target variable for classification. The data is split into training and validation sets using train_test_split from the model_selection module of scikit-learn with a ratio of 80% for training and 20% for validation. A hyperparameter grid param_grid is defined for grid search, specifying different values for the hyperparameters of the Random Forest classifier, which is then created with a fixed random state. Grid search with cross-validation is performed using GridSearchCV to find the best hyperparameters. The scoring metric is 'f1', and the number of jobs is set to -1 to utilize all available CPU cores. The best model obtained from the grid search is then evaluated on the validation set using predict and various evaluation metrics such as accuracy, precision, recall, F1 score, ROC AUC score, and confusion matrix. Finally, a final model is trained on the entire training set using the best hyperparameters obtained from the grid search. This final model can be used to make predictions on new, unseen data.

```
Read CSV file into a pandas DataFrame 'df' using 'pd.read_csv('data_file.csv', sep=',')'
Drop columns 'FileName', 'md5Hash', and 'Benign' from 'df', and assign resulting data to 'X'
Assign 'Benign' column of df to 'y'
Split data into training and validation sets using 'model_selection.train_test_split' with 20% of the data for
validation and a random state of 42

Print number of training samples
Print number of validation samples

Define hyperparameter grid 'param_grid' with possible values for 'n_estimators', 'max_depth', 'min_samples_split',
and 'min_samples_leaf'
Create a Random Forest classifier 'rf' with a random state of 42
Perform a grid search with cross-validation using 'GridSearchCV' on 'rf', with 'param_grid', 5-fold cross-validation,
F1 scoring, and parallel jobs set to -1
Fit the grid search on the training data 'X_train' and 'y_train'
Get the best model from the grid search and assign it to 'best_model'
Predict the validation labels 'y_val_pred' using 'best_model' on 'X_val'

Evaluate best model by calculating:
    'accuracy' using 'accuracy_score' on 'y_val' and 'y_val_pred'
    'precision' using 'precision_score' on 'y_val' and 'y_val_pred'
    'recall' using 'recall_score' on 'y_val' and 'y_val_pred'
    'f1' using 'f1_score' on 'y_val' and 'y_val_pred'
    'roc_auc' using 'roc_auc_score' on 'y_val' and 'y_val_pred'
    'conf_matrix' using 'confusion_matrix' on 'y_val' and 'y_val_pred'

Print the evaluation metrics:
    Print 'accuracy'
    Print 'precision'
    Print 'recall'
    Print 'f1'
    Print 'roc_auc'
    Print 'conf_matrix'

Plot confusion matrix using 'plot_confusion_matrix'

Train 'final model' on entire training set using best parameters from 'grid_search':
    Create 'final_model' as a Random Forest classifier with '**grid_search.best_params_'
    Fit 'final_model' on 'X_train' and 'y_train'
```

Fig. 12 *Pseudocode for Model training process using the Random Forest (RF) algorithm*

Fig. 13 presents the result of various evaluation metrics such as accuracy, precision, recall, F1 score, ROC AUC score, and confusion matrix. The training and validation samples used are 62844 and 15712, respectively, in the 80:20 ratio. The validation accuracy (99.72%) represents the overall accuracy of the model on the validation set, which is also the percentage of correctly predicted samples. The validation precision (99.57%) is the ratio of true positive predictions to the total positive predictions, while the validation recall (99.8%) represents the ratio of true positive predictions to the total actual positive samples. The validation F1 score (99.69%) is the harmonic mean of precision and recall, providing a balanced measure of the model's performance. Next, the validation ROC AUC score (99.73%) measures the Area Under the Receiver Operating Characteristic (ROC) Curve, which indicates the model's ability to discriminate between classes.

The confusion matrix summarizes the model's performance by showing the counts of true positives, false positives, true negatives, and false negatives. The true positives (TP) of 6974 indicate that the model correctly identifies a high number of ransomware samples, which is desirable for a model designed to detect malicious files. On the other hand, the false positives (FP) of 30 represent legitimate files that were incorrectly classified as ransomware. The true negatives (TN) of 8694 show that the model accurately identifies a high number of legitimate files, which is crucial for maintaining a low false positive rate and ensuring that the model does not excessively flag benign files as ransomware, while the false negatives (FN) of 14 represent ransomware samples that the model fails to identify. However, false positives should still be minimized to avoid flagging benign files as threats, which could lead to unnecessary security measures or disruptions for users and false negatives should be minimized as much as possible, as they represent missed opportunities to detect and prevent ransomware attacks, potentially leading to data loss or system compromise.

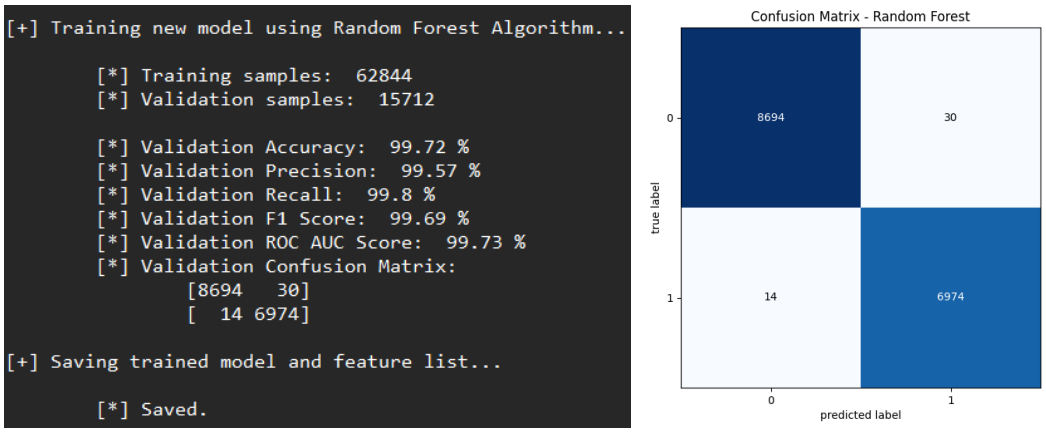

Fig. 13 *Evaluation Result of the trained Random Forest (RF) model*

5.1.3 Scanning and Detection

Fig. 14 displays the pseudocode for the scanning and detection module of the Anti-Ransomware Tool. At first, the model loads from a pickle file ('rf_model.pkl') using joblib.load() and then walks through all the directories and files on the "C:" drive using os.walk(). For each file encountered, the script calls the extract_features() function to extract relevant features from the file, such as machine type, debug size, debug RVA, major image version, major OS version, export RVA, export size, IAT VRA, linker versions, number of sections, size of stack reserve, DLL characteristics, resource size, and the presence of Bitcoin addresses. If the features are successfully extracted, the script creates a list called feature_values containing the values of the extracted features in a specific order. The script then uses the loaded Random Forest model (rf_model) to predict whether the file is ransomware or not based on the extracted features. If the model predicts that the file is ransomware, the script increments the ransomware_detected counter, prints a message indicating that ransomware is detected along with the file path, and attempts to delete the file using os.remove(). If the file is successfully deleted, it prints a message confirming the deletion and adds the file path to the deleted_files list. If an error occurs during deletion, it prints an error message. If the model predicts that the file is not ransomware, the script prints a message indicating it is legitimate.

```
Load trained Random Forest (RF) model from 'rf_model.pkl' using 'joblib.load'
Initialize 'ransomware_detected' and 'deleted_files'
Iterate over all 'files' in the directory "C:\" using 'os.walk'

For each 'file':
    Get 'file_path' by joining 'root' and 'file'
    Extract features from 'file' using 'extract_features' function

    If successfully extracted:
        Prepare 'feature_values':
            file_features['Machine'],
            file_features['DebugSize'],
            file_features['DebugRVA'],
            file_features['MajorImageVersion'],
            file_features['MajorOSVersion'],
            file_features['ExportRVA'],
            file_features['ExportSize'],
            file_features['IatVRA'],
            file_features['MajorLinkerVersion'],
            file_features['MinorLinkerVersion'],
            file_features['NumberOfSections'],
            file_features['SizeOfStackReserve'],
            file_features['DllCharacteristics'],
            file_features['ResourceSize'],
            file_features['BitcoinAddresses'],

    Use 'rf_model' to predict if the 'file' is ransomware

    If is ransomware:
        'ransomware_detected'++
        Print a message indicating ransomware detection with 'file_path'

        Attempt to delete the ransomware file:
            use 'os.remove' with 'file_path'
            Print a message indicating deleted file with 'file_path'

        if OSError as 'e':
            Print a message indicating error deleting file with 'file_path'
            Print error message with 'e'

    Else:
        Print a message indicating legitimate file with 'file_path'.
```

Fig. 14 *Pseudocode for scanning and detection module of Anti-Ransomware Tool*

Fig. 15 shows the Pop-Up Notifications of both ransomware threats found and not found. The detected ransomware threats will then be automatically deleted. If the user clicks the “Details” button, they will be directed to the “Summary” page, while if the user clicks the “OK” button, the pop-up notification will be closed.

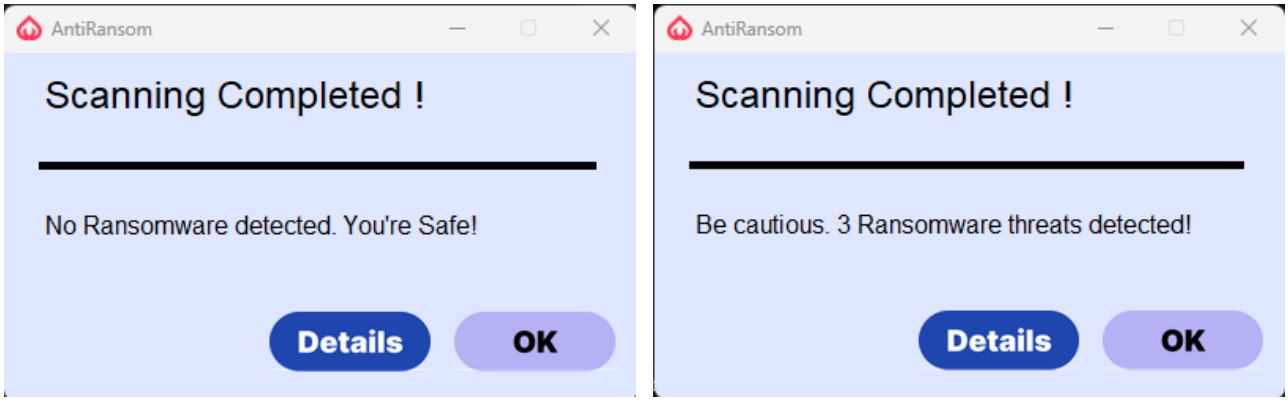

Fig. 15 *Pop-Up Notifications if ransomware threats detected or not found*

5.2 Testing and Verification

This section explains and shows how the Anti-Ransomware Tool is tested in its function, overall performance, compatibility, and user acceptance. This is to ensure that the developed tool is aligned with the requirements and objectives mentioned in the Requirement Gathering and Analysis phase.

5.2.1 Tool Testing

Table 7 displays the test plan of the Anti-Ransomware Tool. The test plan was after the development of the tool. The Test Plan serves as a guide for validating and ensuring the effectiveness of the Anti-Ransomware Tool in a systematic manner. It tested the functionality with different environments, such as different Windows operating

system computers, and the result is passed if the expected result is shown. The developed Anti-Ransomware Tool has passed all the tests.

Table 7 *Test Plan of Anti-Ransomware Tool*

No.	Test List	Description	Actual Result
1	Home Page Functionality Test	Verify the proper functioning of the features accessible from the Home Page.	Pass
2	Scanning Process Test	Evaluate the initiation and monitoring of the scanning process on the Scanning Page.	Pass
3	Summary Page Test	Evaluate the accuracy and completeness of information presented on the Summary Page	Pass
4	Usability Test	Evaluate the overall user experience, such as navigation.	Pass
5	Performance Test	Assess the tool's response time and efficiency under various scenarios.	Pass
6	Compatibility Test	Ensure the tool's compatibility with different hardware configurations and environments.	Pass

Table 8 shows the testing results for ransomware and non-ransomware samples in the Anti-Ransomware Tool. The Anti-Ransomware Tool was tested against 20 samples, including 15 ransomware and 5 non-ransomware samples. The results show that the tool successfully detected 14 of the 15 ransomware samples, indicating a high true positive (TP) rate. The tool accurately identified various types of ransomware, such as Cerber, Cryptowall, Jigsaw, Locky, Mamba, Matsnu, Petya, RedBoot, Satana, TeslaCrypt, UNNAMED, Vipasana, WannaCry, and WannaCry Plus. However, it failed to detect the Thanos Ransomware samples, resulting in false negatives (FN). The tool correctly classified all 5 non-ransomware samples, including Cisco Packet Tracer Installer, PyCharm IDE Installer, WinRar Installer, Bitwarden Installer, and Kaspersky Installer, as true negatives (TN). The detected ransomware threats will be automatically deleted by the Anti-Ransomware Tool, providing an effective defense against these malicious programs.

Table 8 *Testing Result on Ransomware and Non-Ransomware Samples in Anti-Ransomware Tool*

No.	Sample Type	Sample File Name	Result
1	Cerber Ransomware	cerber.exe	TP
2	Cryptowall Ransomware	cryptowall.bin	TP
3	Jigsaw Ransomware	jigsaw	TP
4	Locky Ransomware	Locky	TP
5	Mamba Ransomware	131.exe	TP
6	Matsnu Ransomware	Matsnu-MBRwipingRansomware_1B2D2A4B97C7C2727D571BBF9376F54F_Inkasso Rechnung vom 27.05.2013 .com_	TP
7	Petya Ransomware	4c1dc737915d76b7ce579abddaba74ead6fdb5b519a1ea45308b8c49b950655c.bin	TP
8	GandCrab Ransomware	new.bin	TP
9	Satana Ransomware	683a09da219918258c58a7f61f7dc4161a3a7a377cf82a31b840baabfb9a4a96.bin	TP
		unpacked.mem	TP
10	TeslaCrypt Ransomware	51B4EF5DC9D26B7A26E214CEE90598631E2EAA67	TP
		3372c1edab46837f1e973164fa2d726c5c5e17bcb888828ccd7c4dfcc234a370	TP
		E906FA3D51E86A61741B3499145A114E9BFB7C56	TP
11	Thanos Ransomware	5d40615701c48a122e44f831e7c8643d07765629a83b15d090587f469c77693d	FN
		C	FN
		ae66e009e16f0fad3b70ad20801f48f2edb904fa5341a89e126a26fd3fc80f75	FN
		C	FN

Table 8 *Testing Result on Ransomware and Non-Ransomware Samples in Anti-Ransomware Tool(cont)*

No.	Sample Type	Sample File Name	Result
12	UNNAMED Ransomware	Ransomware.Unnamed_0.exe	TP
13	Vipasana Ransomware	0442cfabb3212644c4b894a7e4a7e84c00fd23489cc4f96490f9988e6074b6ab	TP
		c0cf40b8830d666a24bdd4febdc162e95aa30ed968fa3675e26ad97b2e88e03a	TP
		e49778d20a2f9b1f8b00ddd24b6bcee81af381ed02cfe0a3c9ab3111cda5f573	TP
14	WannaCry Ransomware	ed01ebfbc9eb5bbea545af4d01bf5f1071661840480439c6e5babe8e080e41aa.exe	TP
15	WannaCry Plus Ransomware	Win32.Wannacry.exe	TP
16	Cisco Packet Tracer Installer	CiscoPacketTracer_821_Windows_64bit.exe	TN
17	PyCharm IDE Installer	pycharm-community-2023.3.2.exe	TN
18	WinRar Installer	winrar-x64-624.exe	TN
19	Bitwarden Installer	Bitwarden-Installer-2024.4.3.exe	TN
20	Kaspersky Installer	startup.exe	TN

5.2.2 User Acceptance Testing

The Anti-Ransomware tool is tested in terms of user acceptance and functionality. For the test plan, we received a positive result. Table 9 displays the User Acceptance Testing (UAT) form using Google Form for the Anti-Ransomware Tool that had been given to potential users, and Fig16 shows the form results. A total of 18 respondents from Universiti Tun Hussein Onn Malaysia and daily computer users had responded to the UAT form.

Table 9 *User Acceptance Testing (UAT) Form for Anti-Ransomware Tool*

Question	Description	Result	
		Yes	No
1	The tool's interface is user-friendly and easy to navigate.		
2	All the buttons (e.g., Scan, Quick Scan, Full Scan, Selective Scan, etc.) function as expected and perform their intended actions correctly.		
3	The scanning process for detecting ransomware threats works as expected.		
4	The summary page displays the scanning results in an informative manner.		
5	The tool provides clear and understandable notifications when ransomware threats are detected.		
6	The tool effectively detects and removes the Tesla Crypt Ransomware as intended.		
7	The tool demonstrates reasonable effectiveness in detecting and removing a range of crypto ransomware threats, but there is room for improvement in handling certain variants or emerging threats.		
8	The tool's performance in terms of speed and responsiveness is satisfactory.		
9	I would recommend this Anti-Ransomware Tool to others to protect their systems.		
10	Overall, I am satisfied with the tool's ability to enhance cybersecurity and protect against ransomware threats.		

Fig. 16 shows the user acceptance test result using Google Forms in the Bar Chart. All the respondents gave a positive result from Question 1 to Question 10. This shows that the developed Anti-Ransomware Tool could function and is usable. The only disagreement received from the respondent is for Question 4, where the user thinks that the summary page could display a more comprehensive and detailed scanning result, and it would be even better if it could generate a report.

Applied Information Technology and Computer Science Vol. 5 No. 2 (2024) p. 114-132

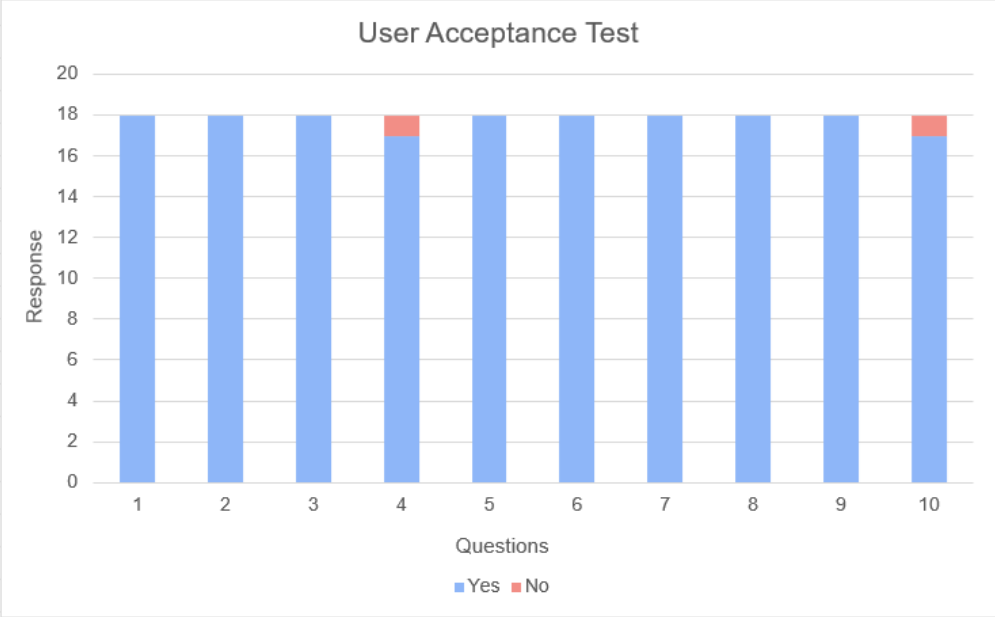

Fig. 16 *User Acceptance Test Result using Google Form*

6 Conclusion

The Anti-Ransomware Tool has successfully achieved all the objectives outlined in this project. The tool was designed using an object-oriented approach, developed by employing the Random Forest (RF) algorithm for effective detection of Crypto Ransomware and implemented with alpha and beta testing involving the target users. The developed tool addresses the problem statement by overcoming the common challenges existing in anti-ransomware tools, such as high false alarms that lead to a loss of trust, reduced productivity, and increased downtime costs due to false positives.

The Anti-Ransomware Tool offers several advantages. It provides a user-friendly interface with simple buttons and instructions to guide users throughout the scanning process. The tool also effectively detects and removes Tesla Crypt Ransomware, demonstrating its capability to handle specific ransomware threats, generate pop-up notifications to alert users about detected threats, and provide a summary of scanning results. However, there are also some disadvantages to the tool. While it demonstrates reasonable effectiveness in detecting and removing a range of Crypto Ransomware threats, there is room for improvement in handling certain variants or emerging threats.

Future improvements can be made to enhance the Anti-Ransomware Tool's capabilities and usability. Firstly, we could expand the tool's compatibility to support a wider range of file types for scanning, as this would increase its versatility. Moreover, we could also incorporate additional features and functionalities, such as real-time protection and customizable scanning frequencies, to further strengthen the tool's ability to combat evolving ransomware threats. Furthermore, the summary page should display more comprehensive and detailed scanning results, and the generation of reports would enhance the tool's usability. Lastly, it is advised to continuously update the machine learning model with new ransomware samples to ensure the tool stays effective against emerging threats.

In conclusion, the developed Anti-Ransomware Tool represents a significant step towards enhancing cybersecurity and protecting users against ransomware threats. The Anti-Ransomware Tool has the potential to be a highly valuable asset in the ongoing battle against ransomware with its continuous improvement and development.

Acknowledgment

The authors would like to thank the Faculty of Computer Science and Information Technology, University Tun Hussein Onn Malaysia, for its support.

References

[1] A. Alraizza and A. Algarni, "Ransomware Detection Using Machine Learning: A Survey," *Big Data and Cognitive Computing*, vol. 7, no. 3, 2023, doi: 10.3390/bdcc7030143.

[2] P. O'Kane, S. Sezer, and D. Carlin, "Evolution of ransomware," *IET Networks*, vol. 7, no. 5, pp. 321–327, 2018, doi: https://doi.org/10.1049/iet-net.2017.0207.

[3] C. Security, P. Rupasinghe, C. Liyanapathirana, and S. Punyasiri, "Signature & Behavior Based Malware Detection." Nov. 2023. doi: 10.13140/RG.2.2.22127.20640.
[4] V. Chew Christopher J.W. and Kumar, "Behaviour based ransomware detection," vol. 58. EasyChair, pp. 127–136, 2019. [Online]. Available: https://hdl.handle.net/10289/12975
[5] S. Aljawarneh, M. Aldwairi, and M. B. Yassein, "Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model," *J Comput Sci*, vol. 25, pp. 152–160, 2018, doi: https://doi.org/10.1016/j.jocs.2017.03.006.
[6] F. Bilstein and D. Plohmann, "YARA-Signator: Automated Generation of Code-based YARA Rules," Nov. 2019. doi: 10.18464/cybin.v5i1.24.
[7] C. Beaman, A. Barkworth, T. D. Akande, S. Hakak, and M. K. Khan, "Ransomware: Recent advances, analysis, challenges and future research directions," *Comput Secur*, vol. 111, p. 102490, 2021, doi: https://doi.org/10.1016/j.cose.2021.102490.
[8] L. Y. Connolly and D. S. Wall, "The rise of crypto-ransomware in a changing cybercrime landscape: Taxonomising countermeasures," *Comput Secur*, vol. 87, p. 101568, 2019, doi: https://doi.org/10.1016/j.cose.2019.101568.
[9] D. Su, J. Liu, X. Wang, and W. Wang, "Detecting Android Locker-Ransomware on Chinese Social Networks," *IEEE Access*, vol. 7, pp. 20381–20393, 2019, doi: 10.1109/ACCESS.2018.2888568.
[10] P. H. Meland, Y. F. F. Bayoumy, and G. Sindre, "The Ransomware-as-a-Service economy within the darknet," *Comput Secur*, vol. 92, p. 101762, 2020, doi: https://doi.org/10.1016/j.cose.2020.101762.
[11] R. Moussaileb, R. E. Navas, and N. Cuppens, "Watch out! Doxware on the way…," *Journal of Information Security and Applications*, vol. 55, p. 102668, 2020, doi: https://doi.org/10.1016/j.jisa.2020.102668.
[12] A. Gautam and N. Rahimi, "Viability of Machine Learning in Android Scareware Detection," Nov. 2023. doi: 10.29007/n5ft.
[13] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez, "A comprehensive survey on support vector machine classification: Applications, challenges and trends," *Neurocomputing*, vol. 408, pp. 189–215, 2020, doi: https://doi.org/10.1016/j.neucom.2019.10.118.
[14] B. Charbuty and A. M. Abdulazeez, "Classification Based on Decision Tree Algorithm for Machine Learning," *Journal of Applied Science and Technology Trends*, 2021, [Online]. Available: https://api.semanticscholar.org/CorpusID:233421415
[15] M. Schonlau and R. Y. Zou, "The random forest algorithm for statistical learning," *Stata J*, vol. 20, no. 1, pp. 3–29, 2020, doi: 10.1177/1536867X20909688.
[16] B. Mahesh, "Machine Learning Algorithms -A Review." Nov. 2019. doi: 10.21275/ART20203995.
[17] G. Sood, "virustotal: R Client for the virustotal API." 2021.
[18] "F-Secure Online Scanner." F-Secure Corporation, Dec. 2023. [Online]. Available: https://community.f-secure.com/other-home-en/kb/articles/5593-what-is-f-secure-online-scanner
[19] "Norton Power Eraser." NortonLifeLock, Dec. 2023. [Online]. Available: https://us.norton.com/?inid=support-nav_norton.com-homepage_logo
[20] K. van Liebergen, J. Caballero, P. Kotzias, and C. Gates, "A Deep Dive into VirusTotal: Characterizing and Clustering a Massive File Feed." 2022.
[21] Z. Q. Chong and Sapiee Jamel, "Web-Based Auction System with Dual-Authentication for Syarikat Perniagaan Fong Yuen Sdn. Bhd.," *Applied Information Technology And Computer Science*, vol. 4, no. 2, pp. 83–102, Nov. 2023, [Online]. Available: https://publisher.uthm.edu.my/periodicals/index.php/aitcs/article/view/12033
[22] X. N. Kee and I. R. A Hamid, "Feature Extraction Tool for Phishing Dataset," *Applied Information Technology And Computer Science*, vol. 4, no. 2, pp. 248–265, Nov. 2023, [Online]. Available: https://publisher.uthm.edu.my/periodicals/index.php/aitcs/article/view/12219