Applied Information Technology And Computer Science Vol. 4 No. 1 (2023) 079-091

AITCS

Homepage: http://publisher.uthm.edu.my/periodicals/index.php/aitcs
e-ISSN :2773-5141

A Dropper Remover Tool

Muhammad Alif Hakim[1], Nurul Azma Abdullah[1]*

[1]Faculty of Computer Science and Information Technology,
Universiti Tun Hussein Onn Malaysia, Parit Raja, Batu Pahat, 86400, MALAYSIA

*Corresponding Author Designation

DOI: https://doi.org/10.30880/aitcs.2023.04.01.005
Received 15 September 2023; Accepted 27 May 2023; Available online 30 June 2023

Abstract: All devices need protection to prevent any data being breached. Antiviruses are created specifically to help, prevent, and remove malware. Dropper is one of malware that is usually called payload because once it penetrates the system, it will release malware into it. To protect devices being attack by dropper, A Dropper Remover Tool developed in this project. This tool could protect the devices from being attacked by detecting and removing the dropper. Object-oriented analysis and design were used to develop the tool. Users can choose the file that user wants to scan it. After that, the user can scan the file that will indicate whether the file is clean or infected. If infected, the user can remove the file. Users also can list out all the files that are in the selected folder. Development of this tool will protect the devices at least from dropper. The tool can easily be installed and used since it is only available in Windows operating system. Besides, this tool can be improved by being able to use in Linux and any operating system. This tool can add more features in future such as firewall, blocking websites and so on to avoid user download free application in unsecure website. In the end, the system has been completed and able to execute the function as proposed. The system able to detect the file that contains dropper. Users need to choose the file to scan whether the file infected or not. The system can tell that files are infected and the aim for this project has been achieved.

Keywords: Dropper, Object-Oriented, Remover Tool

1. Introduction

Malware infiltration can have catastrophic consequences, including data theft, extortion, or paralysis of network systems. Over 92% of malware is delivered by email because most people use email to contact other people instead of meeting them in person [3].

For malware detection tools, there are a lot of creators who invented detection tools such as CrowdStrike Flacon, SolarWinds, McAfee and so on. They offer a lot of features and options to protect our devices. They are creating technology awareness among network users to prevent data leaks and theft, implement and enforcing policies, ensuring the physical safety of hardware devices. The most

*Corresponding author: azma@uthm.edu.my

important is the tools need to be updateable and patching the operating system and application software [4].

However, some of the detection tool also cannot detected dropper since dropper is not harmful. The dropper is one type of trojan that is usually not harmful but acts as payload. Once the dropper can penetrate the devices, it will release all kinds of malware [2].

In development of the tools, of course it cannot be perfect, and a lot of difficulties and issues need to be outcome. Some of these tools cannot protect the devices being attacked by malware threats as example trojan. The most famous trojans is Dropper also known as downloader Trojans [5]. The tool that I proposed which is A Dropper Remover Tool can detect any of trojans that come with dropper and avoid it to penetrate the system. The aim for this project is first to analyze the important aspects and features in detection and remover tools. Also, to design the user interface (UI) become more user friendly and give the best user experience (UX). Second is to develop A Dropper Remover Tool. Finally, to evaluate the competent detection and remover tool by detecting the dropper threats and remove it.

The rest of this paper is organized as follows. Section 2 presents related works of malware and antivirus system. Section 3 describes the proposed model and system design of the tool. Lastly, section 4 presents the discussion and concluding remarks for future works.

2. Related Work

In this related work will review about malware, the dropper or any activity that related to detection and remover tool. While making a review, a lot of information that have been collected about dropper behaviour, types, and flow how the dropper penetrates the system. Review of existing system and comparison table also included in this related work.

2.1 Malware

The definition of Malware is the short term for “Malicious Software”, programs, files or any intrusive software that developed by cybercriminal usually intends to steal or hijack the devices. these Malware could even damage the computer system whenever they able to penetrate the system. Malware has various physical and virtual types of attack to spread the malware itself to infect the devices and networks [1]. For physical, malware can be spread by using USB drive which automatically installs the malicious software to the system without the user knowing.

2.2 Trojan Dropper

A downloader trojan or mainly known as Dropper is a type of malware developed to launch viruses by “dropping” (installing) them. A Dropper called as transportation for other malware to get into the devices. Their presence usually cannot be detected because they disguise themselves within computer systems or directories.

2.3 Comparison Existing System

For the existing system, two systems had been reviewed in the previous section. First is Avast with free security while the second is Avira with Free security and lastly the proposed system Dropper Remover Tool. In this section, this two existing software are compared together with the proposed tool.

Table 1: Comparative analysis of different tools

Features	Avast	Avira	Dropper Remover Tool
Security Status	Yes	Yes	Yes
Specific file Scan	No	Yes	Yes
List all files in the infected folder	No	No	Yes
Dropper Trojan Detection	No	No	Yes

Table 1 shows the comparative analysis of different tools. The first one is Security Status which is all the Anti-Virus application have it to indicates the status of their devices whether the devices or file detect the suspicious threat. Second, specific file scanning needs to be implemented in any detection tool so the user can choose the file to scan it. Unfortunately, Avast free security did not have this type of scan while the proposed system has this function. Third is list all files inside the infected folder. Avast free security and Avira free security unfortunately did not have those features while the proposed system implemented the feature. Lastly, dropper trojan detection which Avast and Avira failed to detect the malware based on experiment that has been conducted while Dropper Remover Tool could detect and remove it from devices [8].

3. Methodology

The selected methodology models for the development of this project are object-oriented analysis and design (OOAD) methodology model. Object-oriented analysis and design (OOAD) are a widely acknowledged technical approach for manipulating a business or system model and a simple graphical diagram for analyzing and improving product quality using the object-oriented prototype method [7]. The software life cycle is usually separated into stages, starting with requirement phase, and progressing through designs, implementation, testing, and Maintenance as shown in Figure 1.

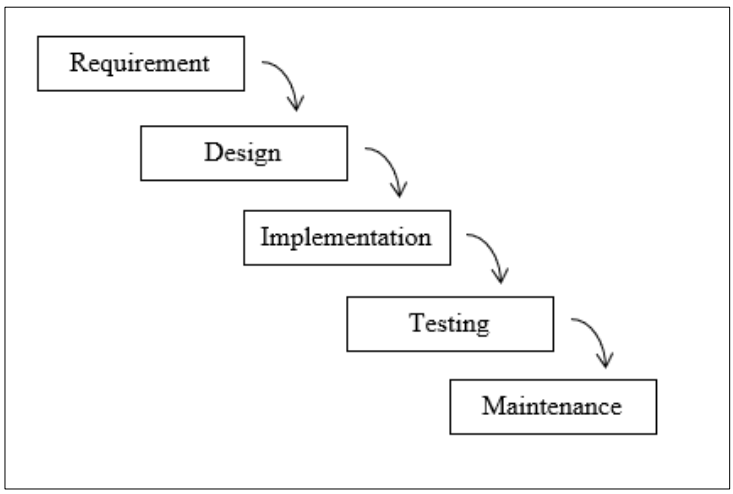

Figure 1: Object-oriented Analysis and Design

The iterative models as example object-oriented analysis and design methodology consist of five phases which requirement, design, implementation, testing and maintenance. This methodology is like waterfall life cycle which each phase completely separates from other phases [6].

3.1 Requirement

The first phase of object-oriented analysis and design methodology is requirement also known as the planning phase. All the requirements are gathered to create a figure of how to represent a real-world application based on the requirement.

During the requirement phase, the analysis is done to research on the requirement of the system and the expected result for the proposed system. The user requirement for the proposed system will be shown in Table 2.

Table 2: User Requirement

No.	Requirement	Descriptions
1	Security Status	The security status will show the status of the devices.
2	Run Smart Scan	Run a quick scan to the devices detect dropper.
3	Select the file	Select the infected or dropper file to scan.
4	Remove The Dropper	The proposed system will be specifically only removed dropper
5.	List all the files in the folder	The proposed system can list all the files that contains in the infected folder.

Table 2 shows the user requirement of the proposed system. There are five user requirements which are security status. Smart scan, select file, remove dropper and list of the files inside the folder.

Second, Functional requirement will describe what the user expects when they interact with the tool. It is important to ensure the tool follows what the user wants. Table 3 shows the functionalities requirement for the proposed system. This proposed system has five functional requirements which security status, smart scan, select file, removed dropper and list all the file inside the folder.

Table 3: Functionalities requirement for the system

Module	Functionalities
Security Status -user	• user able to see the security status of their devices whether bad or good.
Smart Scan -user	• user able run the quick scan on their devices.
Select File -user	• user able to see the dropper behaviour.
Remove the dropper -user	• user able to remove the dropper if detected.
List all the file contains on the folder -user	• user able to list all the file contains on the infected folder.

Non-functional requirement basically describes how the tool works in terms of performance, interface and operational. Table 4 shows the non-functional requirement for the A Dropper Remover Tool. This proposed system has three non-functional requirements which are performance, interface and operational. Performance of the system is all the menus and button in the tool should be easy to use and function. Interface of the system must comfortable and satisfy user experience. The operation of the system is tool only could be run on Windows 10 and 11.

Table 4: Non-functionalities requirement analysis.

Requirement	Description
Performance	All menus and button in the tool should be able to easy to use.
Interface	The user interface must be comfortable and the button, navigation bar should be functional.
Operational	The tool may only be available in windows 10 and 11.

The requirement phase also has the hardware and software requirement to develop the system. The hardware used as physical requirement that needed to execute the system and use the software to create coding of the system. The Software is where the coding will be developed, and the system created. The hardware and software requirement for the proposed system is shown in Table 5.

Table 5: Hardware and Software requirement

Requirement	Description
Hardware	Laptop will be used to develop the system with this OS: - • Windows 10 • Windows 11
Software	Microsoft Visual Studio 2019 used to create the coding including the language: - • C#

3.2 Design

After requirement analysis ended, now move to object-oriented design phase which will take all the concept and ideas from the previous phase. In design and development, helps in design of the system architecture or layout in system design and development, usually following the completion of an object-oriented requirement.

3.2.1 Use-case

Figure 2 illustrates the function of the use-case diagram of the system. Users can choose the file that they wanted to check. Users are also able to run a smart scan to indicate the security status. The user can check the security status of the file. User can remove the dropper that has been detected and lastly user can list all the file in the folder.

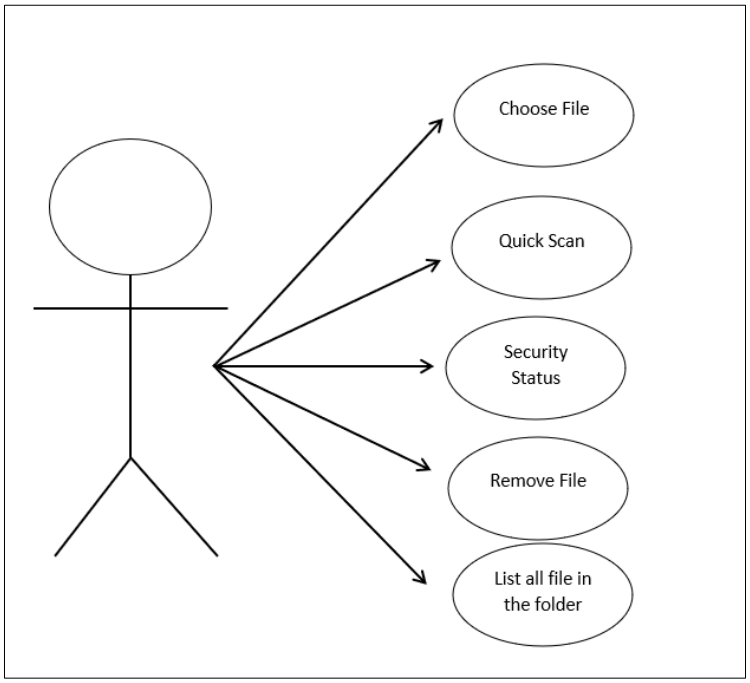

Figure 2: Use-case for the user

3.2.2 Sequence Diagram

Figure 3 shows the sequence diagram of the proposed system. The system starts when the user installs the system. After installing it, the user will move to home page which consists of Select file function, quick scan, security status of file, remove dropper file and list all the file in the folder function.

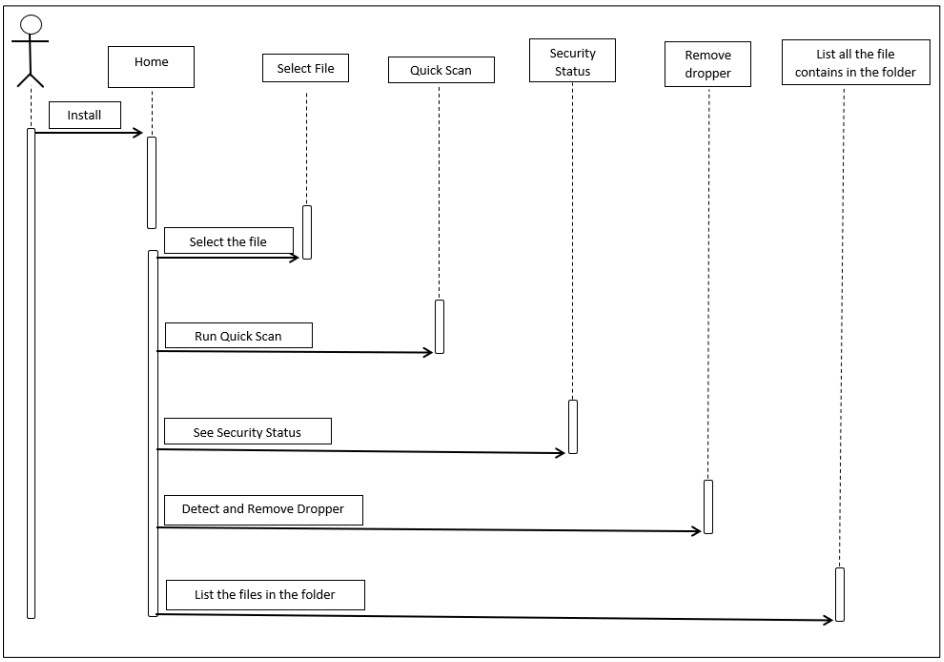

Figure 3: Sequence diagram of the user

3.2.3 Activity Diagram

Figure 4 shows the activity diagram of the proposed system. In the main page, user can choose the file that user want to scan. After choosing the file, the user can scan the file to indicate whether the file is infected or not. If the file clean means the file is not infected but if the file infected, user have option to remove the file. Lastly the user can also see the file contained in the infected folder.

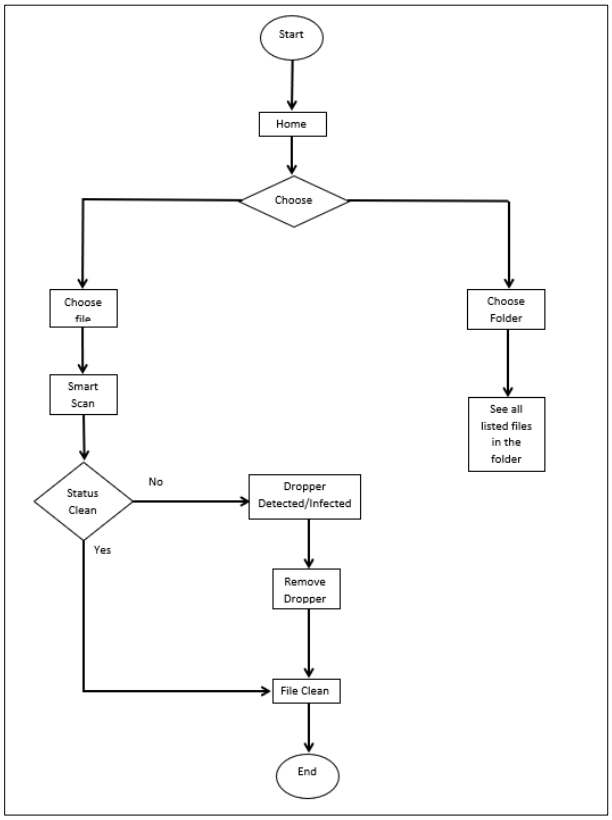

Figure 4: Activity diagram of the system

3.2.4 System Architecture

Figure 5 shows the system architecture of the proposed system. Users can choose the file that user want to scan. After choosing the file, the user can scan the file to indicate whether the file whether infected or not. If the file clean means the file is not infected but if the file infected, user have option to remove the file. Lastly the user can also see the file contained in the infected folder.

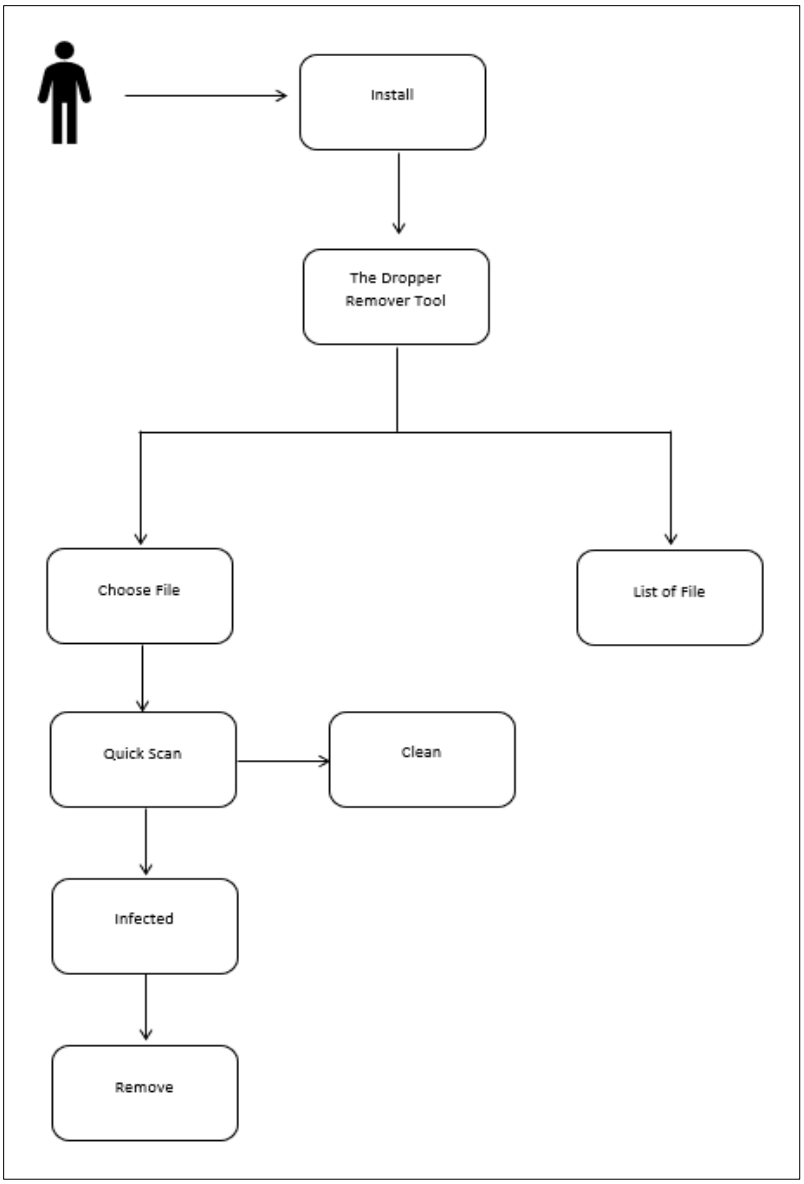

Figure 5: System Architecture of the system

3.2.5 Interface

The design phase where the bone of the system is created and run the function and features that has been offered. The design phase feeds the object-oriented design phase with the conceptual systems model, use cases, system relational model, user interface (UI), and other analysis data. This is how design determines, defines, and designs system classes and objects, as well as their relationships, interfaces, and implementation. Figure 6 shows the initial wireframe for the proposed system. The function of the proposed system is select file, scan, remove and select folder.

Figure 6: Interface for main menu

3.3 Implementation

For object-oriented implementation, the web application or tools can be categorized into front-end and backend coding. For the frontend and backend development, the proposed system uses markup language C# to design the interface of the proposed system also known as bone of the system itself which including the structure of the wording and arrangement, decorate the interfaces by implementing the colors, fonts and reposition the element in the proposed system and used for the validation and interact with the user. For backend development, the proposed system uses C# language to connect A Dropper Remover Tool to the frontend coding. C# is used to create the detection the dropper malware and remove the threat. The proposed system also has smart scan and lists all the files in the folder.

3.4 Testing

The testing is done by using the test plan that has been developed. The test plan consisted of the requirement that listed in requirement phase. Each test gives a fail or pass because of the testing. The testing phase will see if the proposed system could detect the dropper or not and if the proposed system could remove the dropper.

3.5 Maintenance

The maintenance of the system is done by recording all the errors and malfunction features that have been tested by the user in the testing phase. Note will take and written to inspect the system for improvement in the future work. The errors and bugs of the system are also being recorded during the test of the system. However, due to time constraint, the maintenance will not be fully completed.

4. Results and Discussion

Figure 7 shows the result of the proposed project using Microsoft Visual Studio 2019. The system has four functional buttons to complete this program expectation which select file, scan, remove and select folder button.

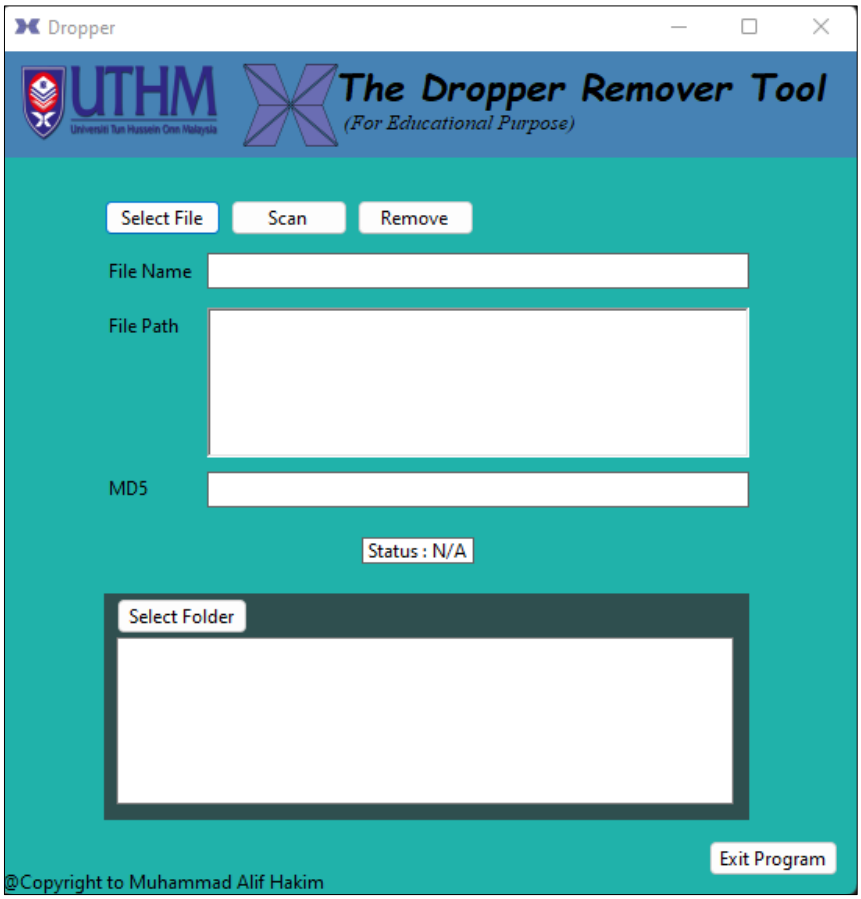

Figure 7: The interface of system

4.1 Select File

The select file button is where user can click on it and browse/choose the specific file that user wants to scan it. Figure 8 shows the code for the select file button. After the user chooses the file, user able to see the file name, file path and md5 of the file.

```
1 reference
private void btnSelect_Click(object sender, EventArgs e)
{
    OpenFileDialog ofd = new OpenFileDialog();
    ofd.Filter = "Allfiles | *.*";
    if (ofd.ShowDialog() == DialogResult.OK)
    {
        tbMD5.Text = GetMD5FromFile(ofd.FileName);
        rtbFilepath.Text = ofd.FileName;
        tbFilename.Text = Path.GetFileName(ofd.FileName);

    }

}
```

Figure 8: Code for Select File Function

4.2 Scan

In this function is for user to scan file. This function is important for this project because it is used to detect the file if the file is infected or not. Figure 9 shows the code for scan button. The code function is where the user can scan the file whether the file is clean or infected. If the file clean means file is not dropper or infected but if the file infected means file is dropper. It will scan the file based on md5 of the file. If the md5 of the file is on the "md5.text" means the file is dropped.

```
1 reference
private void btnScan_Click(object sender, EventArgs e)
{

    var md5signatures = File.ReadAllLines("C:\\Users\\Legion 5i\\OneDrive\\Desktop\\fyp\\Dropper\\MD5.txt");



    if (md5signatures.Contains(tbMD5.Text))
    {

        lbStatus.Text = "Infected Files";
        lbStatus.ForeColor = Color.Red;
    }
    else
    {
        lbStatus.Text = "Clean Files";
        lbStatus.ForeColor = Color.Green;
    }
}
```

Figure 9: Code for Scan Function

4.3 Remove

This function is for the user to remove the file. Remove function is where user can decide to remove the infected file. Figure 10 shows the code for the remove button. The code function is where the user can remove the file. When a user scans the file and detects the file is infected, user able to remove the file by click on remove button.

```
1 reference
private void btnRemove_Click(object sender, EventArgs e)
{
    MessageBox.Show("Are you sure want to delete the file?");

    string filenPath = rtbFilepath.Text;

    if (File.Exists(filenPath))
    {
        File.Delete(filenPath);
        MessageBox.Show("File Deleted");
    }
    else
    {
        MessageBox.Show("File not Exists");
    }
}
```

Figure 10: Code for Remove Function

4.4 Select Folder

This function is for the user to browse and choose the folder to list out all the files in the folder. Figure 11 shows the code for the select folder button. The code function is for the user to browse and select infected folder so user can see all the file contains in the folder.

```
1 reference
private void btnSelect2_Click(object sender, EventArgs e)
{
    FolderBrowserDialog fbd = new FolderBrowserDialog();

    if (fbd.ShowDialog() == DialogResult.OK)
    {
        listBox2.Items.Clear();
        string[] files = Directory.GetFiles(fbd.SelectedPath);
        string[] dirs = Directory.GetDirectories(fbd.SelectedPath);

        foreach (string file in files)
        {
            listBox2.Items.Add(Path.GetFileName(file));
        }
        foreach (string dir in files)
        {
            listBox2.Items.Add(Path.GetFileName(dir));
        }
    }
}
```

Figure 11: Code for Select Folder Function.

4.5 Testing

Figure 12 shows the testing that has been done by the system. First, the user will choose the any file that user wants to scan. In this testing, the file that might have the dropper inside is selected. The system will show the file name, file path and md5 of the file. The md5 of the file will be compare by the txt file that has been made that list all the dropper md5. If the md5 of the file same as in txt file, that means the file is infected by dropper. So, the status of the file will change to infected. In this case, the user can remove the file by clicking the remove button and the file will be removed.

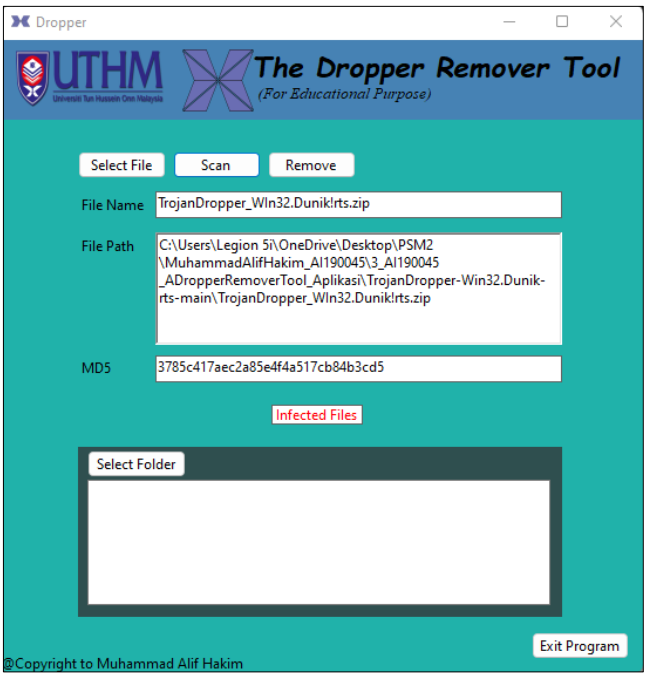

Figure 12: Scanning the dropper file

4.6 Discussion

The objective of developing The Dropper Remover Tool has been reached. However, there are several obstacles that need to be repaired from time to time to enhance the tool itself.

There are several advantages of this system that are identified after completion of development and testing carried out on the site. The advantages are first, the user can choose any file that they want to scan and check the security status of the file. Second, the user can see the md5 of the selected file, so user do not have to find it on other online tools. Finally, the system is easy to use and very comfortable for a first-time user.

The constraint of the system can be identified because of the development of this system are first, the system only scans the selected file only, not all file in the devices. Second, the removed items can be found at recycle bin. Finally, the system only works in windows environment and not for Linux.

Apart from the advantages and constraints of the system, several suggestions for improvement were also obtained from the testing process. The suggestion is to improve the tool where it can scan all the files in the devices. Second, the remove function will remove permanently from devices. Finally, the tool can work in Linux.

5. Conclusion

The objective and requirement to develop A Dropper Remover Tool has been reached. Every function that has been listed works as expected. The main reason of this tool developed is to create a system for user to scan the specific file and see whether the file is infected or clean. User can remove the file immediately if the file infects by dropper. In future, I hope the tool can improve the scan task where the user can scan entire file in the devices. I hope the tool could continually improve by doing some more research about detection or remover tools.

Acknowledgment

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

References

[1] Michael L, Steven A, Blake H, Matthew R (2010). Malware Analyst's Cookbook and DVD. John Wiley & Sons, Incorporated. Retrieved from ebookcentral.proquest.com.

[2] Murali, R., Ravi, A., & Agarwal, H. (2020). A malware variant resistant to traditional analysis techniques. In 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE) (pp. 1-7). IEEE.

[3] Kwon, B. J., Mondal, J., Jang, J., Bilge, L., & Dumitraş, T. (2015). The dropper effect: Insights into malware distribution with downloader graph analytics. In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security. pp. 1118-1129.

[4] Li, F., Lai, A., & Ddl, D. (2011). Evidence of advanced persistent threat: A case study of malware for political espionage. In 2011 6th International Conference on Malicious and Unwanted Software (pp. 102-109). IEEE.

[5] J. Caballero, C. Grier, C. Kreibich, and V. Paxson (2011). Measuring pay-per-install: The commoditization of malware distribution. In USENIX Security Symposium.

[6] Pandey, S. K., Singh, G. P., & Kansal, V. (2010). An Alternative approach to Temporary Memory Management in Databases using Object Oriented Systems. IJCSNS, 10(10), 158.

[7] Mukherjee, M. (2016). Object-Oriented Analysis and Design. International Journal of Advanced Engineering and Management, 1(1), 1-11.

[8] Gasparinatos S. (2017). Malware Development with the Use of Known Techniques. Master's Thesis of Security of Digital System Programme.