

# Intelligent Malicious Content Checker (I-MCC) for Files, Images and URL

## **Team members**

Mediana—A0231458E

Meng Chenxi—A0231546J

Sarah Elita Shi Yuan Wong—A0231507N

Zhou Xinyi—A0231538H

November 17, 2021

# Contents

<b>1</b>	<b>Business Case</b>	<b>3</b>
<b>2</b>	<b>System Design</b>	<b>4</b>
2.1	Static Analysis	4
2.1.1	BinText	5
2.1.2	PEStudio	6
2.2	Image Analysis	6
2.3	Web Checking	6
<b>3</b>	<b>System Development</b>	<b>7</b>
3.1	Frontend and Backend	7
3.2	RPA and IPA	8
3.2.1	Static Analysis	9
3.2.2	Online URL/file hash checking	12
3.2.3	Google APIs	15
<b>4</b>	<b>Conclusion</b>	<b>16</b>
	<b>References</b>	<b>17</b>
<b>A</b>	<b>Project Proposal</b>	<b>18</b>
<b>B</b>	<b>System Functionalities Against Knowledge</b>	<b>20</b>
<b>C</b>	<b>Installation And User Guide</b>	<b>21</b>
C.1	Objectives	21
C.2	Scopes	21
C.3	System Overview	21
C.4	Installation	21
C.4.1	Requirements	21
C.4.2	Backend Server Setup and Configuration	22
C.5	User Guide	24
C.5.1	Use Case Overview	24
<b>D</b>	<b>Individual Report</b>	<b>27</b>
D.1	Mediana - Individual Report	27
D.2	Meng Chenxi - Individual Report	27
D.3	Sarah Elita Shi Yuan Wong - Individual Report	27
D.4	Zhou Xinyi - Individual Report	28

# 1 Business Case

**Executive Summary** According to the AV-TEST Institute[[tre](#)], there are currently over a billion malware programs online and millions of new malware programs are being created each month. Malware is often spread through email attachments, file downloads, or by getting users to click on malicious website links. Furthermore, all kinds of content exist on the internet, and we might unintentionally come across explicit or inappropriate content. Therefore, it is important to always be cautious about what we download and open on the internet. To help keep users safe on the internet, we have developed an automated content checking system using RPA and IPA. The system employs RPA to perform static analysis for zipped files and to check the safety of website links, and IPA to check for the presence of any inappropriate images in a folder. As an MVP, it is currently implemented as a web-application, where users can provide a zipped file or URL that they want to check, and the results of the check will be sent back to the user via email.

**Product Description** The product is a web-based application which aims to help users keep their devices safe and free from malware and inappropriate material. Users can go to our application website and upload the files or website links that they wish to check. The application then performs 3 main functions using RPA and IPA at the backend: <sup>1</sup> static analysis of the file contents to check for harmful code; <sup>2</sup> scans through multiple images and checks if there are any inappropriate material (such as pornographic material); <sup>3</sup> checks the file hash or website on a free online service (Virus Total), which analyses files and URLs for any type of malicious content. After checking, the analysis report is then sent to the email address provided by the user.

**Strategy** We know there are some mature malware detection websites and we actually make use of them partly. For example, we take advantage of Virus Total to increase the comprehensiveness of our product. Also, we develop our own methods to detect improper images using Google APIs. That's how the diversity of our product shows.

**Market Analysis** After I read the article *Top 10 Best Free Anti-Malware Software (2021)*[[top](#)], I find there are some similarities between them and we try to make our product performs like them. First of all, they are all free and this means they have the most support from the average user. Secondly, they are all easy to use and also have clear layout, which means user won't be bothered by dizzying and useless information while using the detector. We make our product have all these features and on top of that, better user experience.

**Customer Analysis** Here we talk about better user experience. Easy to use, no more operational burden, comprehensive, these are users care about most. For users with higher demands, we can add more high-class feature to meet their needs, such as large file or video detection.

**Financial Analysis** Initially, we attract average users, and then after our product becomes more sophisticated, we can attract high-demand users and advertisers, and that's when we accumulate business value.

**Future Plans** This is just a course project, and we did what we could do, given our limited time and resources. If we have the chance to continue our work, we can optimize our product, the malware detector, with the support of other tools and technology, making it more powerful, such as having the ability to detect malicious intrusion while user surfing on the Internet and browsing a document without uploading files, stopping browsing by force if it's necessary, etc.

**Recommendation** In conclusion, what we do is really a useful tool for everyone who use the Internet to make use of. Especially in this prospering information era, we need more powerful protect for our daily use of computer. For companies which need high-quality information security protection, they can choose a promoting edition which provides better services that meet the needs.

## 2 System Design

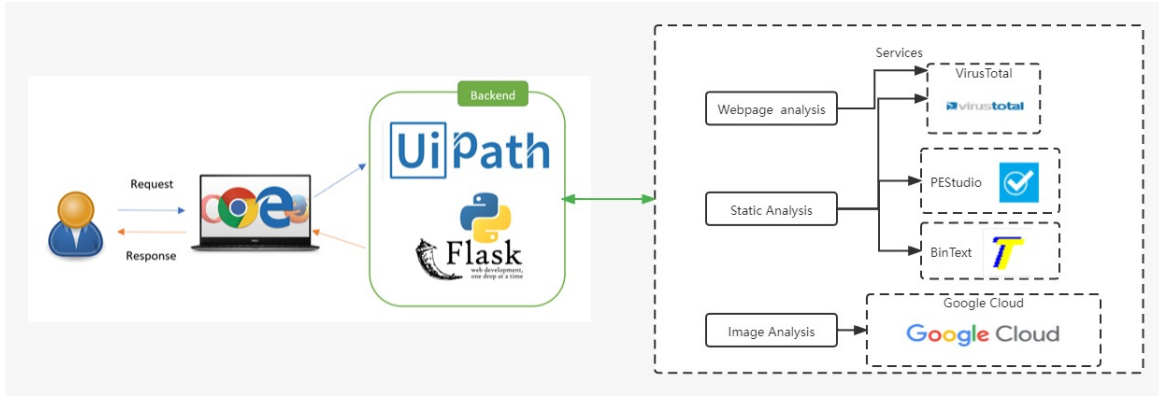


Figure 1: System Architecture

### 2.1 Static Analysis

Malware analysis is a study to determine the behaviour, functionality and potential impact of a suspicious file or program. There are two methods to perform malware analysis: static analysis and dynamic analysis. In static analysis, each component of the binary files is dissected and studied without executing it, while in dynamic analysis, files are executed and run on a host system to observe the behaviour of the suspicious program.

In this project, we automated static malware analysis using two basic tools, PEStudio and BinText. It can be used to develop host-based signatures or indicators to detect malicious code on victim machines. Though, static analysis is slow and require further investigation of the output results, yet it is the safest way to analyse the suspicious file, as executing the code without proper sandboxing or setup could infect the system. Dynamic analysis which requires execution of the programs and safe environment to run the malware can be incorporated in the future release.

## 2.1.1 BinText

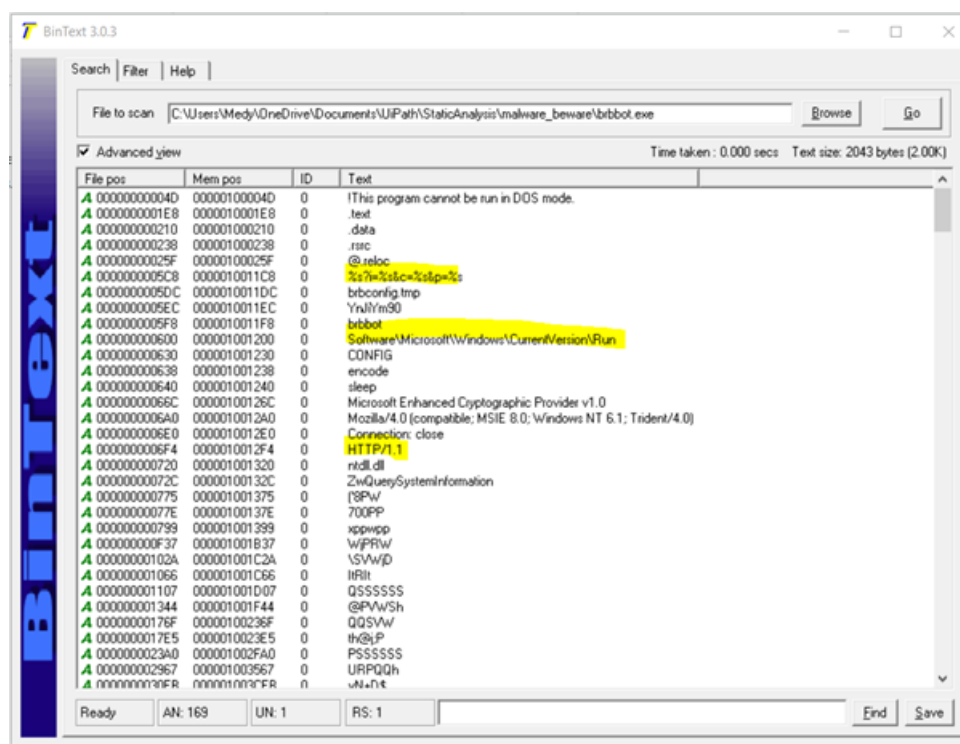


Figure 2: BinText - Interesting Strings

BinText is a small and fast text extractor that can find ASCII, Unicode and Resource strings in a file. Searching through the strings is the simplest way to obtain hints about the functionality of a program. Figure above shows some of the interesting strings extracted using BinText. It could be used as a guide for the next steps in the process of examining the malware and as an Indicator of Compromise for detection. The conclusions derived from the BinText are usually the assumptions made by analyst and need to be further proved or disproved. Table below shows the indicator and remarks for brbbot.exe

Mechanism	Indicator	Remark
File	brbconfig.tmp	Potential temporary configuration file
Persistence	Software\Microsoft\Windows\CurrentVersion\Run	The malware may have persistence mechanism to auto-run even after system reboot
Network	%s?i=%s&c=%s&p=%s	Part of an HTTP request: Parameter=value pair
Network	HTTP/1.1	The malware might communicate over HTTP
User Agent	Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0)	The malware might be checking if the victim's machine has a specific browser (MSIE 8.0 = IE 8) and a specific operating system (Windows NT 6.1 = Windows 7)

Figure 3: Indicator and Remarks for BRBBOT.EXE

### 2.1.2 PESTudio

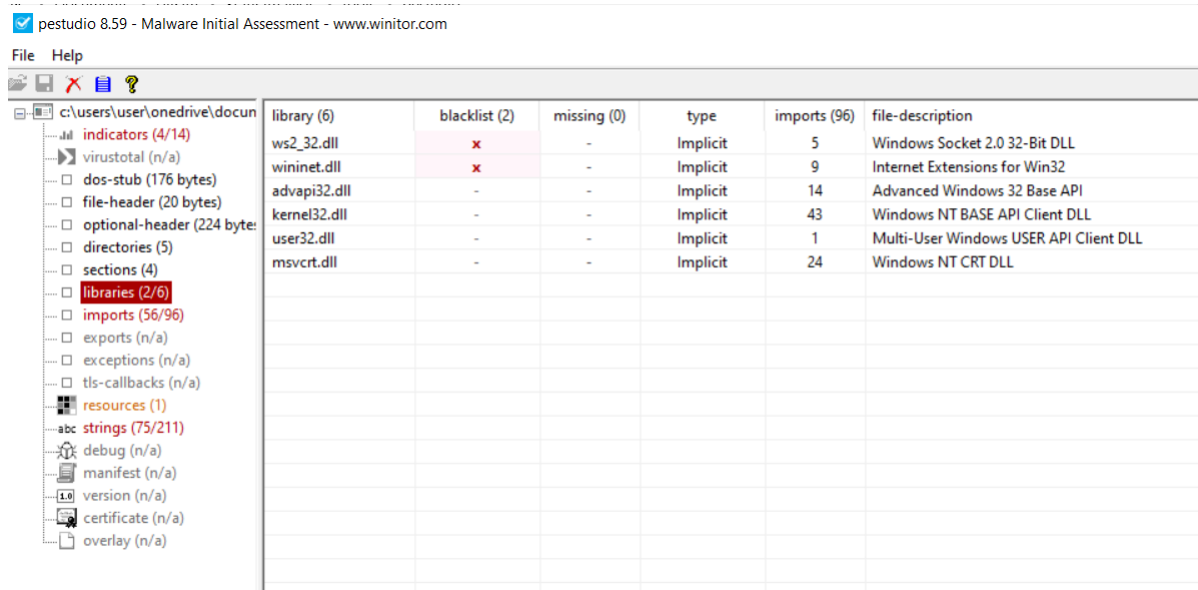


Figure 4: PESTudio

PEStudio scan the suspicious files to spot the artifacts and lists out the file's header information, ie list the libraries and functions referenced in the suspicious files to ease and accelerate malware assessment. Figure above shows the malware initial assessment using PESTudio.

## 2.2 Image Analysis

The image analysis is based on Google Cloud AI. We use *google.cloud.vision\_v1* package to do detect explicit content (SafeSearch), such as adult content or violent content within an image.

There are 3 categories that we will detect for users: adult, violence, and racy. Google Cloud AI returns the likelihood that each is present in a given image.

```
likelihood_name = ('UNKNOWN', 'VERY_UNLIKELY', 'UNLIKELY', 'POSSIBLE',  
                  'LIKELY', 'VERY_LIKELY')
```

Figure 5: Likelihood List

After user upload the zip file of image, our system will unzip and go through every photo in the unzip folder. If the likelihood is likely or beyond, then the system will remove this explicit image from the folder. After all photos processed, the filtered folder will be zipped into a new zip file for user to check.

## 2.3 Web Checking

We have implemented an automated online checking of the given file hash or website link using a free online service (Virus Total). This website performs analysis of files and URLs for any type of malicious content. The web checking component will be triggered under 2 circumstances: (1) when user inputs a website link to be checked, or (2) when static analysis of file has a file hash to be checked.

The web checking software robot accepts a string containing either a file hash or URL and queries the Virus Total website with that string. It then gathers the output on the results page and includes it in the summary report for the user. More details on the technical implementation are explained in the system development section.

## 3 System Development

### 3.1 Frontend and Backend

I-MCC front end is a web application user interface running on HTML, and the backend is using FlaskApp, Python and UiPath. Figure below illustrate the front-end and back-end communication flow.

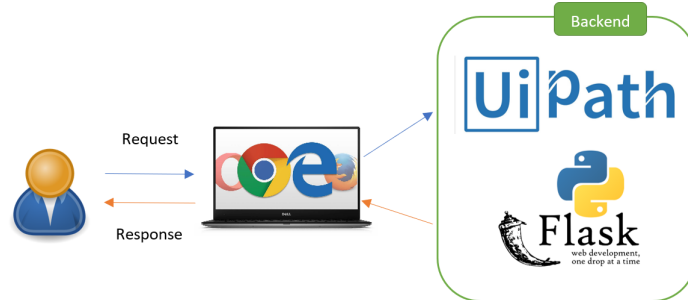


Figure 6: I-MCC FrontEnd and BackEnd

Three tasks that can be performed by users:

**URL Checker** Input URL in the textbox and click Submit URL for checking the URL in VirusTotal dataset whether it is marked as malicious in multiple antivirus engine.

**Images Content Checker** Upload images in zip and click “Send the file” for checking whether the zip file contains any inappropriate images like violence, and abuse. It will delete all the inappropriate images. A zip file with all the inappropriate images is deleted will be returned to users for download.

**Files Checker** Upload binary files in zip and encrypted with password “malware” for performing static malware analysis. It will perform analysis as stated in Section XX Static Analysis then output PEStudio results, BinText results and VirusTotal Hash Check results for users to download. Figure below shows the web-based GUI for users.

← → ↻ ⚠ Not secure | 192.168.0.121:5000

## URL

URL:

## Upload Images

Choose a zipped file

No file chosen

## Upload Zipped Binary Files

Choose a zipped file encrypted with password "malware"

No file chosen

Figure 7: I-MCC Web GUI (Upload)

← → ↻ ⚠ Not secure | 192.168.0.121:5000/images

## Download Results

[File](#)

Figure 8: I-MCC Web GUI (Download)

### 3.2 RPA and IPA

UiPath is an end-to-end automation platform for building and deploying software robots. It provides tools for each step of RPA/IPA implementation, such as process discovery, building the robot, management of the automated process, running the bots, and a user-friendly interface for end-users.

In this project we have used UiPath Studio, which is the free community version of the build platform, to implement RPA for static analysis and online URL/file hash checking.



### 3.2.1 Static Analysis

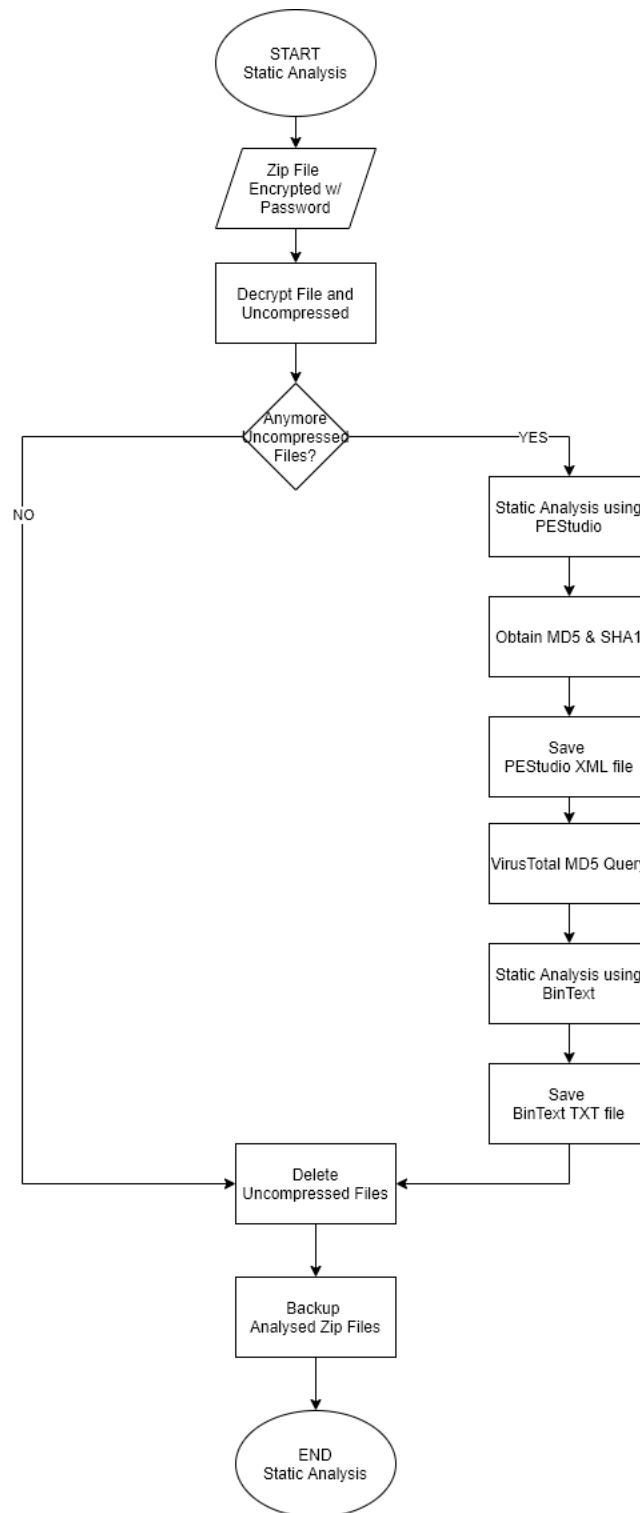


Figure 9: Static Analysis Activities Flow

**MainStaticAnalysis (MainStaticA.xml)** MainStaticA.xml contains the main flow of the static analysis. It invokes FOUR (4) workflows, including DecryptFile.xml, StaticAnalysis.xml, "QueryVirus-totalHash.xml", and lastly "CleanUpFile.xml".

**File Decryption (DecryptFile.xaml)** To properly handle the uploaded files due to the potential malicious file is being uploaded, we have requested users to encrypt the folder with password “malware”. DecryptFile.xaml workflow will be called in the MainStaticA.xaml to perform file decryption. It uses the UPath.Python.Activities to load python script and performs following actions:

- 1) Decrypt all the zip file in the MAL\_WORK\_FOLDER which contains all the uploaded zip files to be analysed,
- 2) Output all the program name or filenames to MainStaticAnalysis for further processing in StaticAnalysis.xaml

**Static Malware Analysis using PEStudio and BinText (StaticAnalysis.xaml)** Filename retrieves from Decrypt.xaml is passed as an argument to StaticAnalysis.xaml. In this workflow, if the filename exists, then BinText.xaml and PEStudio.xaml are further invoked.

BinText.xaml: Perform static malware analysis using BinText and output the results as a TEXT file.

File pos	Mem pos	ID	Text
=====	=====	==	====
0000000004D	0000010004D	0	!This program cannot be run in DOS mode.
0000000001E8	0000010001E8	0	.text
000000000210	000001000210	0	.data
000000000238	000001000238	0	.rsrc
00000000025F	00000100025F	0	@.reloc
0000000005C8	0000010011C8	0	%s?i=%s&c=%s&p=%s
0000000005DC	0000010011DC	0	brbconfig.tmp
0000000005EC	0000010011EC	0	YnJiYm90
0000000005F8	0000010011F8	0	brbbot
000000000600	000001001200	0	Software\Microsoft\Windows\CurrentVersion\Run
000000000630	000001001230	0	CONFIG
000000000638	000001001238	0	encode
000000000640	000001001240	0	sleep
00000000066C	00000100126C	0	Microsoft Enhanced Cryptographic Provider v1.0
0000000006A0	0000010012A0	0	Mozilla/4.0 (compatible; <u>MSIE</u> 8.0; Windows NT 6.1; Trident/4.0)
0000000006E0	0000010012E0	0	Connection: close
0000000006F4	0000010012F4	0	HTTP/1.1
000000000720	000001001320	0	ntdll.dll
00000000072C	00000100132C	0	ZwQuerySystemInformation
000000000775	000001001375	0	('8PW
00000000077E	00000100137E	0	700PP
000000000799	000001001399	0	xppwpp
000000000F37	000001001B37	0	WjPRW
00000000102A	000001001C2A	0	\SVWjD
000000001066	000001001C66	0	ItRIIt
000000001107	000001001D07	0	QSSSSSS

Figure 10: Output Results (BinText)

PEStudio.xaml: Perform static malware analysis using PEStudio and output the results as an XML document. The MD5 of the uploaded file is extracted and return to MainStaticA.xaml to be used by QueryVirusTotalHash.xaml

```

C:\Users\User\AppData\Loca... x
<?xml version="1.0" encoding="UTF-8"?>
<!-- pestudio 8.59 - Malware Initial Assessment - www.winitor.com-->
- <image name="c:\users\user\pycharmprojects\isa-ipa-2021-11-17-is03ft-grp1-
intelligentmaliciouscontentchecker_i-mcc\systemcodes\uiopath\staticanalysis\malware_beware\unzip\brbbot.exe">
  - <overview>
    <file-description>n/a</file-description>
    <file-version>n/a</file-version>
    <created>00:00:0000 - 00:00:00</created>
    <cpu>32</cpu>
    <size>18944</size>
    <type>executable</type>
    <subsystem>GUI</subsystem>
    <signature>Microsoft Visual C++ 8</signature>
    <entropy>6.032</entropy>
    <md5>11DD7DA7FAA0130DAC2560930E90C8B1</md5>
    <sha1>2C9E509DE4B3EC03589B5C95BABA06A9387195E6</sha1>
    <imphash>n/a</imphash>
  </overview>
  - <indicators count="15">
    <indicator severity="2">The file references the Windows Native API</indicator>
    <indicator severity="1">The file modifies the Registry</indicator>
    <indicator severity="2">The file references the Windows Socket (winsock) library</indicator>
    <indicator severity="2">The file references other process(es)</indicator>
    <indicator severity="2">The file references the Event Log</indicator>
    <indicator severity="2">The file references the Windows Internet (WinINet) library</indicator>
    <indicator severity="2">The file installs an Exception Handler</indicator>
    <indicator severity="1">The file is scored (58/69) by virustotal</indicator>
    <indicator severity="1">The file references blacklisted string(s)</indicator>
    <indicator severity="8">The file ignores Address Space Layout Randomization (ASLR)</indicator>
    <indicator severity="5">The file opts for cookies on the stack (GS)</indicator>
  </indicators>
</image>

```

Figure 11: Output Results (PEStudio)

**VirusTotal – Hash Check (QueryVirustotalHash.xml)** The MD5 extracted from the Pestudio is further search in VirusTotal dataset that contains multiple antivirus engines. It provides the total number of engines that marked the file as malicious, malware category, and if available, additional information about the malware. QueryVirusTotalHash output the results in TXT file and XLS file as show in figure below.

Virus Total Summary		
1. 58 / 69 security vendors flagged this file as malicious		
	A	B
1	Vendor	Category
2	Ad-Aware	Dropped:Generic.Malware.SF.B75165A5
3	AhnLab-V3	Malware/Win32.RL_Generic.R355032
4	Alibaba	Trojan:Win32/Generic.08498d37
5	ALYac	Dropped:Generic.Malware.SF.B75165A5
6	Antiy-AVL	Trojan/Generic.ASMalwS.FFF164
7	Avast	Win32:Agent-BCKN [Trj]
8	AVG	Win32:Agent-BCKN [Trj]
9	Avira (no cloud)	TR/Agent.sqjbn
10	BitDefender	Dropped:Generic.Malware.SF.B75165A5
11	BitDefenderTheta	AI:Packer.78BF02F91F
12	Bkav Pro	W32.AIDetect.malware2
13	CAT-QuickHeal	Trojan.Mauvaise.SL1

Figure 12: Output Results (VirusTotal - Hash Check)

**House-Keeping Files (CleanUpFile.xml)** After completed the static malware analysis, CleanUp-File workflow is called in MainStaticA.xml to housekeep the files.

- All files in MAL\_WORK\_FOLDER (potential malicious zip files) are move to DONE\_ANALYSED\_FOLDER in password-protected mode.
- Delete all files in MAL\_UNZIP\_FOLDER where contains all the unzip files (not password-protected).

### 3.2.2 Online URL/file hash checking

We have 2 separate but similar web checking workflows to handle the 2 different scenarios of hash and URL checking. File hash checking is implemented within static analysis, while URL checking is an individual workflow.

The figure below is an overview of the workflow implemented in UiPath for Website URL checking.

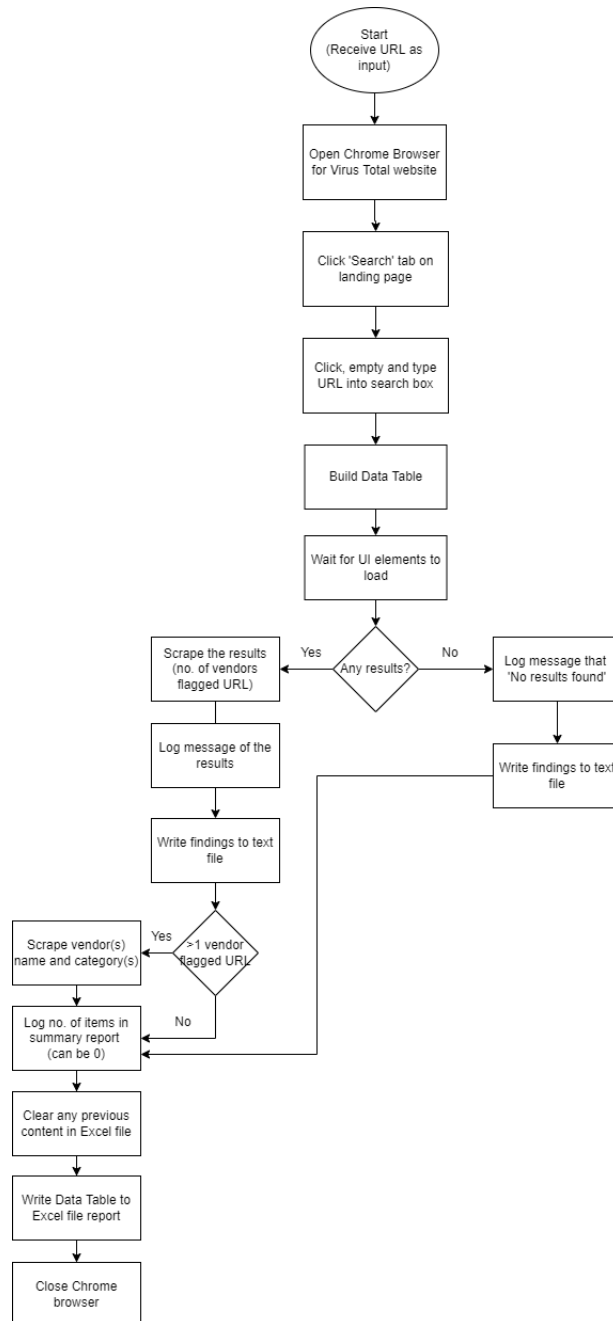


Figure 13: Workflow in UiPath for checking Website URL

Within the UiPath workflow, when the web checking workflow is activated, UiPath first opens a

Chrome browser and navigates to the Virus Total page (search tab) as shown in the Figure 14.

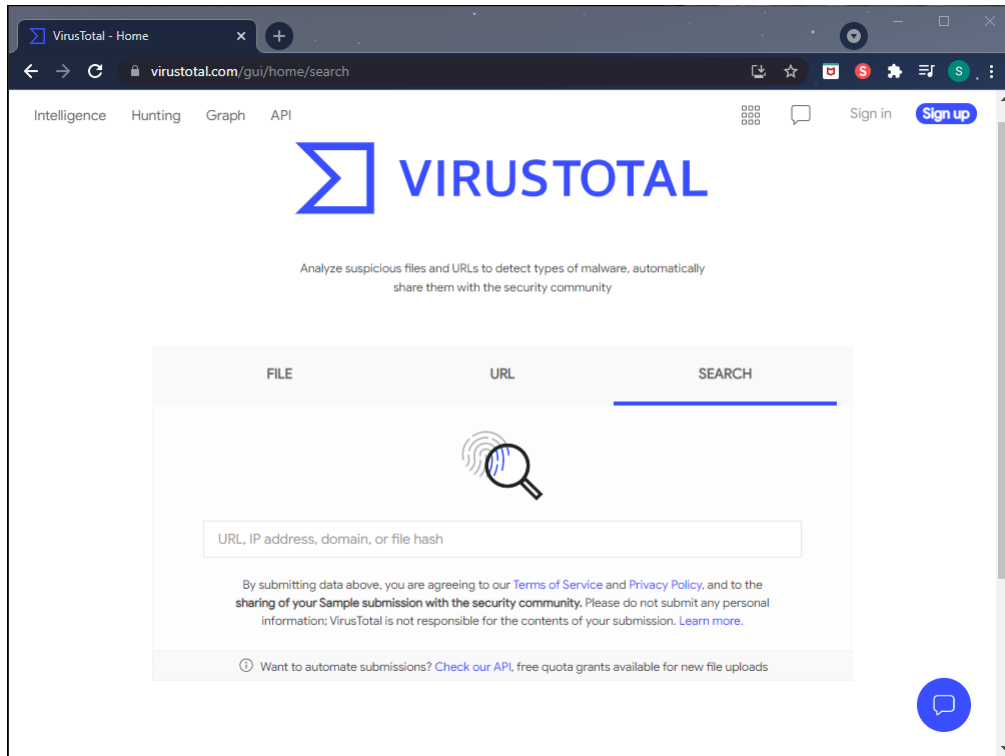


Figure 14: UiPath opens Chrome browser and lands at Virus Total landing page

The Element Finder in UiPath then locates the search box using pre-defined selectors. It performs a click and clear text before typing in the string to be searched and pressing the enter key. The search string can be a file hash or an unknown website link, depending on what the user had uploaded for checking. If the user had uploaded a file, then the file hash will be passed into the web checking workflow for file hashes, from the static analysis workflow. If the user has uploaded a URL then it will be directly passed into the web checking workflow for URLs. A sample search string is shown in the Figure 15.

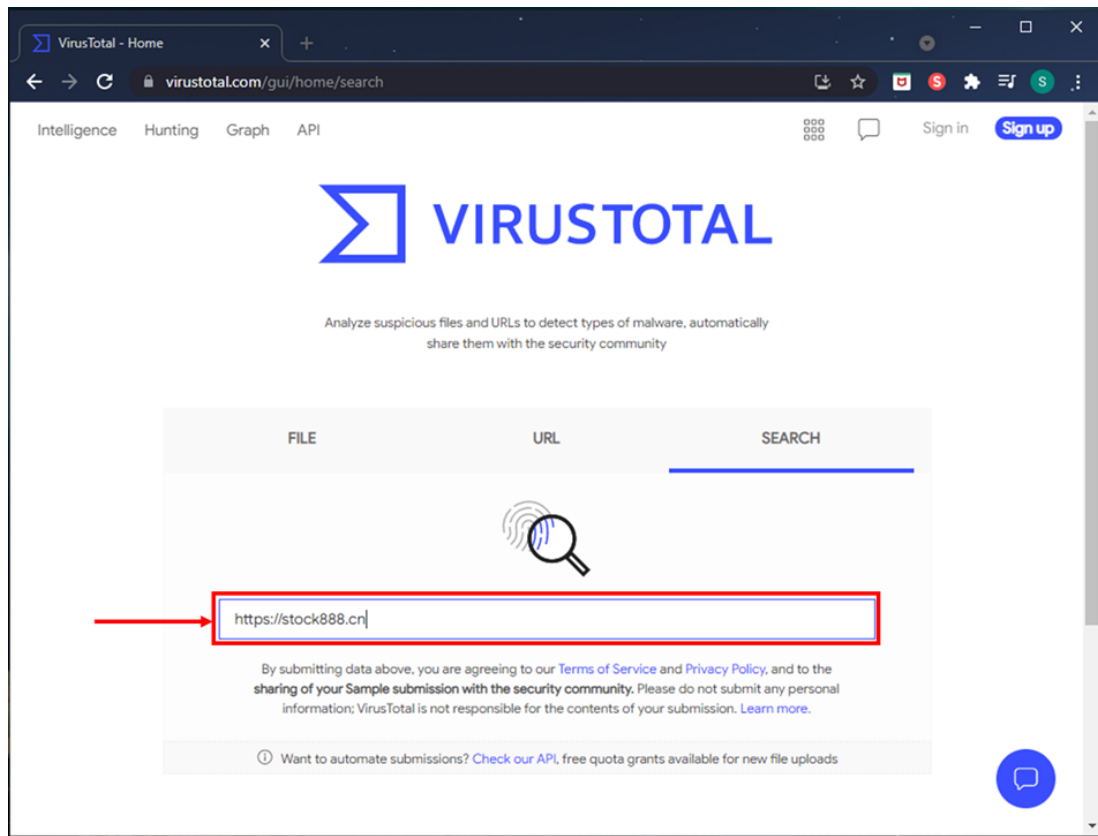


Figure 15: UiPath types into search box

Figure 16 shows the results of the search on Virus Total. It shows us the number of security vendors which have flagged the sample URL/file as malicious. The text highlighted by red boxes on the top left are scraped by UiPath as numerator and denominator values to be included in the analysis report. The results page also shows us a table of details containing vendor names and the respective categories that the searched URL/file has been categorized as. We only scrape the data items (vendor name, category) for the vendors which have flagged the URL/file as malicious.

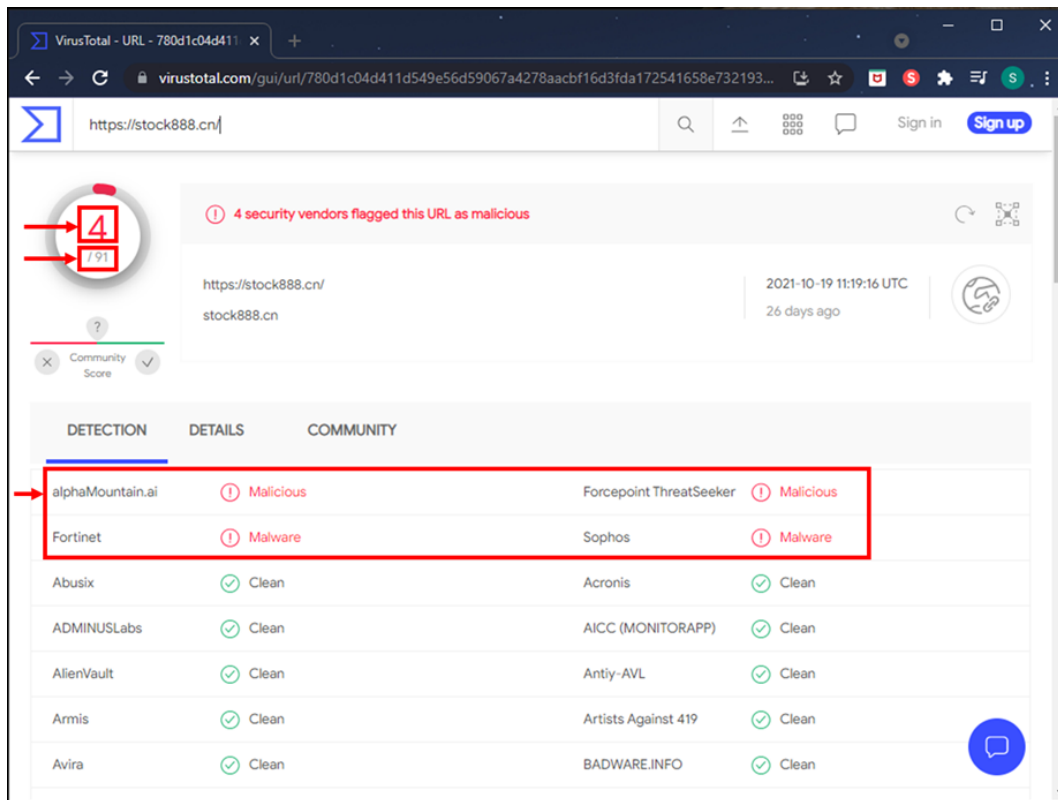


Figure 16: UiPath gets text from UI elements

### 3.2.3 Google APIs

Google Cloud platform provide huge amount of machine learning services such as speech-text transform, item detect, natural language processing, etc. It is easy to use and have mature community to discuss the usage.

First, we initialize a project on Google Cloud platform. Then add the vision API.

To get Google Cloud access in Python script, we need to generate and initialize the client with this json file to get authenticate.

```
client = vision_v1.ImageAnnotatorClient.from_service_account_json("sigma-freedom-326000-91cabb*****.json")
```

The following is the sample of how to get one picture's safe degree. The returned "safe" contains the three categories of likelihood that we need.

```
response = client.safe_search_detection(image=image)
safe = response.safe_search_annotation
```

## 4 Conclusion

RPA and IPA has many potential use cases. In this project, we have explored the use of automation in helping users check the safety of unknown files, website links, and images. Automation is useful in this case as it is able to perform multiple checks quickly and return the analysis report, which saves a lot of time and effort for the user. We have used both RPA and IPA in our solution. RPA was mainly implemented with UiPath, while IPA was implemented using Google Cloud Platform API for image analysis.



## References

- [top] <https://www.antivirussoftwareguide.com/free-malware-protection>.
- [tre] <https://www.av-test.org/en/statistics/malware/>.

## A Project Proposal

## Project Proposal

**Date of proposal:**

17/11/2021

**Project Title:**

Intelligent Malicious Content Checker (I-MCC) for Files, Images and URL

**Group Members:**

Mediana

Meng Chenxi

Sarah Elita Shi Yuan Wong

Zhou Xinyi

**Project Descriptions:**

Malware is often spread through email attachments, file downloads, or by getting users to click on malicious website links. Furthermore, all kinds of content exist on the internet, and we might unintentionally come across explicit or inappropriate content. Therefore, it is important to always be cautious about what we download and open on the internet. To help keep users safe on the internet, we have developed an automated content checking system using RPA and IPA. The system employs RPA to perform static analysis for zipped files and to check the safety of website links, and IPA to check for the presence of any inappropriate images in a folder. As an MVP, it is currently implemented as a web-application, where users can provide a zipped file or URL that they want to check, and the results of the check will be sent back to the user via email.

## B System Functionalities Against Knowledge

Course Module	Techniques and Skills
<b>Intelligent Process Automation</b>	Google Cloud API <ul style="list-style-type: none"><li>• Google Cloud Vision API is used to detect inappropriate content in images and filter it out.</li></ul>
<b>Software Robot Best Practises</b> <b>Strategy Management</b>	UIPath <ul style="list-style-type: none"><li>• UIPath is used to build and deploy the software robot. We automate the URL checking process, Image Content Filtering process and Static Malware Analysis process</li></ul>

## C Installation And User Guide

### C.1 Objectives

The objective of this document is to provide an overview of Intelligent Malicious Content Checker (I-MCC) application and the necessary information to use the application. The manual assumes that the reader has sufficient understanding on system implementation, Robotic Process Automation (RPA), programming language (Python), Google Cloud API and FlaskApp.

### C.2 Scopes

The high-level scope of the user guide will encompass THREE (3) sections:

1. System Overview
2. Installation and Configuration
3. User Manual (Use Case)

### C.3 System Overview

Figure 17 shows the system overview for I-MCC. Following are the components used in I-MCC.

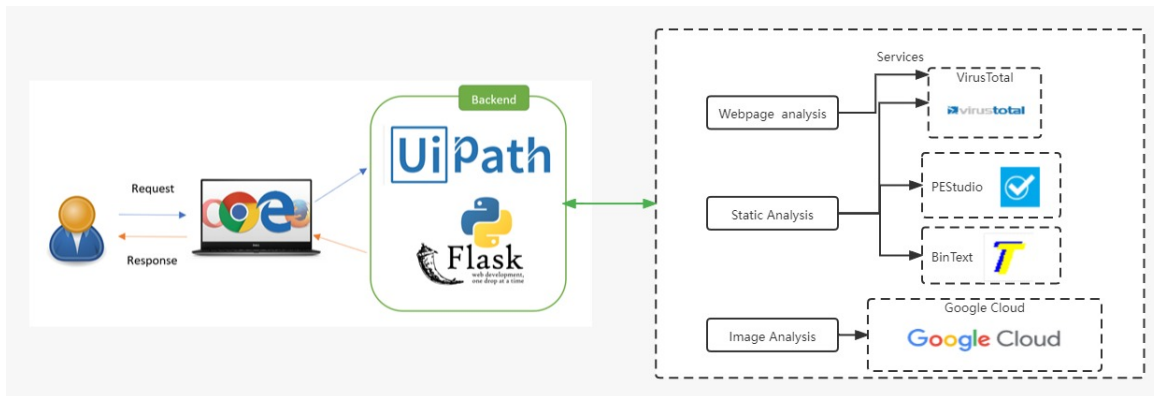


Figure 17: System Overview

1. **Web browser** – User interface with I-MCC websites
2. **UIPath** – Robotic Process Automation tools to automate the process
3. **FlaskApp** – tools, libraries and technologies that are used in this project to build a web application
4. **Google Cloud API** – APIs that can call services to detect explicit content in image
5. **VirusTotal, PESTudio and BinText** – Tools and websites to perform static malware analysis

### C.4 Installation

#### C.4.1 Requirements

Description	Specification
Software	Python 3.8 UIPath Community Edition
Packages	All required python packages are defined in the requirements.txt, please use 'pip install' command to install them.

### C.4.2 Backend Server Setup and Configuration

Below shows the step to setup the backend server. It only requires minimum setup and configuration as all the necessary codes are included in the github folder – SystemCodes.

1. Install UIPath Community Edition(<https://docs.uipath.com/installation-and-upgrade/docs/studio-install-studio>)
2. Use 'git clone' command to download the project from the following URL: [https://github.com/mediana-medy/ISA-IPA-2021-11-17-IS03FT-GRP1-IntelligentMaliciousContentChecker\\_I-MCC](https://github.com/mediana-medy/ISA-IPA-2021-11-17-IS03FT-GRP1-IntelligentMaliciousContentChecker_I-MCC)
3. All the required codes for UIPath, FlaskApp, Google Cloud API, and tools used in the I-MCC are resided in SystemCodes folder
  - a) **UIPath/StaticAnalysis** – contains all the required RPA XAML files, static analysis tools like BinText, and PESTudio. Please ensure the downloaded files containing the following directory and RPA files.

Names	Directory/Files
back_up_malware-zip_file done_analyse malware_beware pythonscript tools virustotal_result resultfile	Directory
MainStaticA.xaml BinText.xaml CleanUpFile.xaml DecryptFile.xaml PEStudio.xaml project.json QueryVirusotalHash.xaml QueryVirusotalURL.xaml StaticAnalysis.xaml	RPA Files

- b) **Download\_folder** – contains the results of analysis for users to download
- c) **Images\_module** – contains the python files for performing the images content checker using Google Cloud API.
- d) **Templates** – contains two html pages, ie upload.html and download.html
- e) **Testfile** – contains two zipped file that users can use for testing the images content checker module and static analysis checker.
- f) **Upload\_folder** – contains the zipped file that are uploaded by users for performing analysis
- g) **App2.py** – main python files for receiving the request from users and calling relevant functions to perform the three modules, ie URL Checker, Images Content Checker and Files Checker (Static Analysis)

master ISA-IPA-2021-11-17-IS03FT-GRP1-IntelligentMaliciousContentChecker\_I-MCC / SystemCodes /

mediana-medu UIPath	
..	
UIPath/StaticAnalysis	UIPath
download_folder	init
images_module	Initial commit
templates	Initial commit
testfile	init
upload_folder	init
app2.py	Initial commit
requirements.txt	Initial commit

Figure 18: System Codes (Folder)

- After installation of UIPath, ensure that UIRobot.exe are also installed and resided in this default folder: "C:/Users/User/AppData/Local/Programs/UiPath/Studio/UiRobot.exe". Please update the ROBOT\_EXE in app2.py if the UIRobot.exe is not in the default folder.

```

app2.py x static_utils.py x
15 BINARY_RESULT =UIPATH_FOLDER + "resultfile/"
16 ROBOT_EXE = "C:/Users/User/AppData/Local/Programs/UiPath/Studio/UiRobot.exe"

```

Figure 19: Define ROBOT\_EXE in app2.py

- Please update the UIPATH\_FOLDER in app2.py where the GitHub downloaded files reside according to your environment.

```

app2.py x static_utils.py x
12 UIPATH_FOLDER = "C:/Users/User/PycharmProjects/ISA-IPA-2021-11-17-IS03FT-GRP1-IntelligentMaliciousContentChecker_I-MCC/SystemCodes/UIPath/StaticAnalysis/"

```

- Run the app2.py and click browser <http://localhost:5000>

The screenshot shows a web browser at localhost:5000. The page has three main sections:

- URL**: A section with a label "URL:" followed by a text input field and a "Submit URL" button.
- Upload Images**: A section with the text "Choose a zipped file", a "Choose File" button (which shows "No file chosen"), and a "Send the file" button.
- Upload Zipped Binary Files**: A section with the text "Choose a zipped file encrypted with password 'malware'", a "Choose File" button (which shows "No file chosen"), and a "Send the file" button.

Figure 20: I-MCC Webpage

## C.5 User Guide

### C.5.1 Use Case Overview

I-MCC provides 3 modes for users, URL Checker, Images Content Checker and File Checker (Static Malware Analysis).

**URL Checker** Input URL in the textbox and click Submit URL for checking the URL in VirusTotal dataset whether it is marked as malicious in multiple antivirus engine.

This is a close-up of the "URL" section from the previous figure. It shows the heading "URL" in a large, bold, serif font. Below it is the label "URL:" followed by a text input field. At the bottom of this section is a button labeled "Submit URL".

Figure 21: URL Input



C > Desktop > urlresult.zip



Name	Type
 VirusTotalReport.txt	TXT File
 VirusTotalReport.xlsx	Microsoft Excel Worksh

Figure 22: URL Results - VirusTotal Report

**Images Content Checker** Upload images in zip and click “Send the file” for checking whether the zip file contains any inappropriate images like violence, and abuse. It will delete all the inappropriate images. A zip file with all the inappropriate images is deleted will be returned to users for download.

## Upload Images

Choose a zipped file

Choose File No file chosen

Send the file

Figure 23: Images Content Checker (Input Images)

C > Desktop > imagesresult.zip



Name	Type
 camp.jpeg	JPEG File
 Graves-Island-camping-stars.jpg	JPG File

Figure 24: URL Results - VirusTotal Report

**File Checker** Upload binary files in zip and encrypted with password “malware” for performing static malware analysis. It will perform analysis as stated in Section XX Static Analysis then output PEStudio results, BinText results and VirusTotal Hash Check results for users to download.

## Upload Zipped Binary Files

Choose a zipped file encrypted with password "malware"

Choose File brbconfig.zip

Send the file

Figure 25: Files Checker - Static Malware Analysis (Input Password-Protected Zip Files)

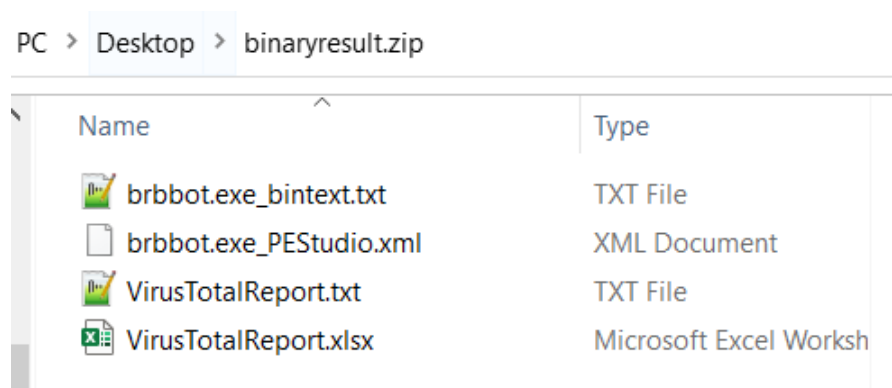


Figure 26: Files Checker - Static Malware Analysis containing BinText, PEStudio and VirusTotal results

## D Individual Report

### D.1 Mediana - Individual Report

**Personal contribution to group project** I initiated the idea of creating an application that allow users to check the characteristics of URLs, Images and Files whether it is safe, appropriate and benign. For technical component, I was responsible for the following task: 1) Static Malware Analysis Process Automation (using application BinText and PESTudio), and 2) FrontEnd and BackEnd Development. I also merged the three modules (URL Checker, Images Content Checker and Files Checker – Static Malware Analysis) into a runnable system. Using the RPA UIPath tools, I automated the file checking modules using basic tools for static malware analysis, ie PESTudio and BinText. While for frontend and backend, I utilized HTML page and FlaskApp to develop a simple web application for users to upload and download the results. For report, I wrote the section mentioned above including frontend and backend, Static Analysis, Installation and User Guide.

**What learnt is most useful for me** Through this project, I become more familiar in automating a task using UIPath for enterprise application (opensource tools) automation and web automation. Besides, I have also learnt to develop a simple web application. There are a lot of limitation in using UIPath for enterprise application or opensource tools, yet I managed to use workaround and overcome the limitation.

**How I can apply the knowledge and skills in other situations or your workplaces** The knowledge and skills that I gained in this project will be very useful in my future workplaces, because many companies are looking to streamline and automate their process to increase their work efficiency and turn-around time. I can be more confidence in handling automation either web automation, Enterprise Application (Open-Source Tools) automation, and Email Automation, knowing the potential challenges and workaround to resolve it.

### D.2 Meng Chenxi - Individual Report

**Personal contribution to group project** My original assignment was to connect UIPath system and User Interface, which I thought can be done using UIPath Assistant, but after trying UIPath Assistant, I can't connect orchestrator so I give up this method. After that, I did a User Interface using React(driven by JavaScript), which can let user to input name, email and files to be tested, while, due to the time limitation and I totally don't know how to build a flask backend, so my part is not been accepted to our project, which is mean what I did is in vain. Actually, I did entire business part of our project. I wrote business case which is included in the report, and also I did the business video using animaker. It's quite a nice try although it is a bit tedious to me because I haven't use animaker before. I am very familiar with LaTeX, so I did the report part which is integrating everyone's paragraph together and make a nice-looking report.

**What learnt is most useful for me** I learn about how to make a frontend using JavaScript and it means a lot to me because I totally don't know any of it before. I learn some algorithms and how to use it, although I learn so slow that I am much later than my teammates. But it's quite a good way to push me. I am not major in Computer Science as an undergraduate so it's quite difficult for me to follow up other classmates, but I will try to.

**How I can apply the knowledge and skills in other situations or your workplaces** I will apply video making and JavaScript, also UIPath techniques in my future workplace.

### D.3 Sarah Elita Shi Yuan Wong - Individual Report

**Personal contribution to group project** I have worked on the online checking for URLs and file hashes. My part mainly uses UiPath web automation functions, Data Table, writing to Excel and text file.

For the report writing, we each wrote the sections that we did the work on and combined our writing.

**What learnt is most useful for me** Through this project, I have strengthened my RPA skills using UiPath. It was not easy to obtain the correct selectors for the table data in the HTML but it was a good practice. I have also learnt to implement an RPA solution in practice and the many various applications that RPA can have.

Furthermore, on the non-technical side I have worked with classmates who are from different background from me. This has also let me improve my communication and teamwork skills when collaborating with groupmates on a project.

**How I can apply the knowledge and skills in other situations or your workplaces** If my future work requires any automation, I would be more confident in looking into possible ways that RPA and IPA can be implemented.

## D.4 Zhou Xinyi - Individual Report

**Personal contribution to group project** Throughout the project, I participated in the image detection. And I also contributed to the project report, which includes project structure figure, elaborating implements. At first we decide to let users send their request through email, and use UiPath to extract the attached file for later analysis, then send the email back with malware report. To make use of cloud AI or local AI, I came up with the idea that to detect the picture or video data. Then I found that Google AI just provided the API “vision” to detect explicit content in picture. Based on what I learnt in class, I use the project that I built for workshop on Google Cloud Platform to enable easy access of the cloud AI. All above is written in python script, so I have to merge it with other UiPath functions. I add python package in UiPath to load the script and invoke the method to get the python object to run. It will pass the path of the image folder to the script and output the filtered images to a zip file at assigned directory.

**What learnt is most useful for me** I became more familiar with the Google Cloud Platform and UiPath. And also learnt the management of minimal valuable project, how to carry out these kind of project in a short limited time with limited source.

**How I can apply the knowledge and skills in other situations or your workplaces** As mentioned above, I gained more experience in building practical uipath agent to meet the requirements and how to use Google Cloud APIs. Google Cloud is really a easy to use tools not only in AI area but also data warehouse and computing. Making good use of it can relief my work burden and hard ware requirements. In addition, getting more familiar with Python and the use of Pandas will also be of great help to me. They are indispensable tools today to processing data and analyze data.