KKH ANTIVIRUS

Project Report

(19CA3581)

Submitted By

HEMANTH REDDY	**(ENG19CA0011)**
KAREEM KHAN	**(ENG19CA0013)**
KRISHNA PRASAD	**(ENG19CA0016)**

In partial fulfillment for the award of degree of

BACHELOR OF COMPUTER APPLICATION

DEPARTMENT OF COMPUTER APPLICATIONS

SCHOOL OF ENGINEERING

DAYANANDA SAGAR UNIVERSITY

Hosur Rd, Kudlu Gate, Srinivasa Nagar, Hal Layout, Singasandra,

Bengaluru, Karnataka 560068

JAN 2022

DAYANANDA SAGAR UNIVERSITY

Hosur Rd, Kudlu Gate, Srinivasa Nagar, Hal Layout, Singasandra,

Bengaluru, Karnataka 56006.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BONAFIDE CERTIFICATE

Certified that the major project work entitled "*KKH ANTIVIRUS*" carried out by ***"HEMANTH REDDY (ENG19CA0011)", KAREEM KHAN (ENG19CA0013), KRISHNA PRASAD M (ENG19CA0016)"*** bonafide students at Dayananda Sagar University in fulfillment for 5th semester of Bachelor of Computer Applications, during the year 2020-21. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of mini project work prescribed for the said degree.

GUIDE	**HOD**
Prof. Vasanthi Kumari	**Prof. Vasanthi Kumari**
Assoc. Professor & Head	**Assoc. Professor & Head**

Project Viva-Voice Held on-____________

Internal Examiner: **External Examiner:**

DECLARATION

““HEMANTH REDDY (ENG19CA0011)”, KAREEM KHAN (ENG19CA0013), KRISHNA PRASAD M (ENG19CA0016)” *fifth* semester students declare that the major project entitled **“KKH ANTIVIRUS”** has been carried out and submitted by us in fulfillment of the 5th semester of Bachelor of Computer Applications during the academic year 2020-21. I also declare that, to the best of my knowledge and belief, the work reported here is accepted and satisfied.

HEMANTH REDDY (ENG19CA0011)
KAREEM KHAN (ENG19CA0013)
KRISHNA PRASAD M (ENG19CA0016)

ACKNOWLEDGEMENT

The project report on **“KKH ANTIVIRUS”** is the outcome of guidance, moral support and knowledge imparted on me, throughout my work. For this I acknowledge and express immense gratitude to all those who have guided and supported me during the preparation of this mini project.

I take this opportunity to express my gratitude to everyone who has extended their support for helping me in the mini project completion.

I would like to show my greatest appreciation to Prof. **Vasanthi Kumari**, HOD, Dept of CSE, Major Project Guide, Dept. of CSE and Guide for constantly guiding us throughout the project.

I would also like to thank all teaching and non-teaching staff of the Computer Science and Engineering Department for directly or indirectly helping us in completion of the Major Project.

Lastly and most importantly I convey my gratitude to our parents who have been the source of inspiration and for instrumental help in successful completion of the project.

HEMANTH REDDY (ENG19CA0011)
KAREEM KHAN (ENG19CA0013)
KRISHNA PRASAD M (ENG19CA0016)

ABSTRACT

Computer viruses are executable code programs that have a unique ability to replicate themselves in computer systems and spread rapidly from one computer to another affecting file, documents and programs to alter their normal running. Viruses are represented as patterns of computer instructional codes that exist over time in computer systems. Antiviruses on the other hand are programs specially developed to counter challenges brought about by viruses as they protect the computer systems from virus attacks by heavily relying on the controls enhanced in their databases. Antiviruses therefore scan the computer using some specific patterns of bytes indicative of known viruses. To stay current, they must be developers of these antiviruses and update their databases whenever new viral strains arise. This paper reviews the various virus and antivirus patterns and various detection schemes.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1

1.INTRODUCTION

Antivirus (usually written as the abbreviation AV) is software used to prevent, detect and remove malware (of all descriptions), such as: viruses, malicious, ransom ware, key loggers, backdoors, Trojan, worms.
A variety of strategies are typically employed.

Signature-based detection involves searching for known patterns of data within executable code. However, it is possible for a computer to be infected with new malware for which no signature is yet known.
To counter such so-called zero-day threats, heuristics can be used. One type of heuristic approach, generic signatures, can identify new viruses or variants of existing viruses by looking for known malicious code, or slight variations of such code, in files.

<u>1.1.1 How does antivirus work?</u>

Antivirus software begins operating by checking your computer programs and files against a database of known types of malwares.
Since new viruses are constantly created and distributed by hackers, it will also scan computers for the possibility of new or unknown type of malware threats.

Typically, most programs will use three different detection devices: specific detection, which identifies known malware; generic detection, which looks for known parts or types of malware or patterns that are related by a common codebase; and heuristic detection,
which scans for unknown viruses by identifying known suspicious file structures. When the program finds a file that contains a virus, it will usually mark it for deletion, making it inaccessible and removing the risk to your device.

<u>1.1.2 What Does Antivirus Software Do?</u>

What Does Antivirus Software Do in Your Computer?
An antivirus software, or shortened to AV software, is a software program designed to identify, remove, or isolate malicious programs and codes that may get installed or infect your system. Viruses are computer codes that could replicate themselves and attach themselves to the operating system's main code in order to be executed

Once established, they perform their functions which include but not limited to encrypting important data, leaking out information, destroying the operating system, or even worse, locking out the user from accessing their device. Spyware and other malware do similar things but mostly tracking your actions and collecting your personal information.

Antivirus software therefore performs three major functions:

- Identifying malicious codes in the system.
- Removing them by destroying or isolating them.
- Most importantly, protecting your device from getting infected with such malicious codes.

Scan specific files or directories for any malware or known malicious patterns
Allow you to initiate a scan of a particular file or your entire computer, or of a CD or flash drive at any time.
Remove any malicious code detected.

1.1.3 How Does Antivirus Remove Viruses?

The antivirus is designed to identify malicious codes, Trojan horses, spyware and other malware programs that corrupt the system and cause malfunctions. The Trojan horses, for instance, have the ability to disguise themselves as legitimate programs enticing the user to install them. The AV software should be able to detect this and alert the user before installing such programs.

It does this through three different ways.

Specific detection – This is whereby the antivirus scans the device for suspected malware programs using a set of predefined characteristics of code base. It does this by comparing such suspected codes to those predefined characteristics and alienating those that match up.

Generic detection – Sometimes not all predefined characteristics do not define all types of virus codes. The AV software then uses its knowledge of malware codes to identify pieces of codes that might appear malicious and alienates them. The user is then allowed to determine whether they think it's a safe program and if they're unsure, the program is removed from the system.

Heuristic detection – This final one is the most complex of all as the malware or virus infecting the system is not known by the AV software. So the antivirus scans through the system looking for actions that raise red flags or appear abnormal. Once identified, the program is flagged and removed or alienated from the system.

1.2. TABS IN OUR ANTIVIRUS

1.2.1 URL Scan: Uniform Resource Locators

The location of a webpage or file on the Internet.

URL scanners are used to quickly analyze if a link/URL is suspicious or unsafe.
Detect malicious URLs checks whether the URL is safe from viruses, Phishing, Malware, Parked Domains.
After completion of the scan, it shows total number of scans performed and number of positive indications and detections from the URL

1.2.2 IP Scan

IP scanner scans the IP address and checks whether the IP is safe from malwares.
Ip scanner show the owner and country of the IP address, number of detected URLs and number of detected malicious files on the IP address.

1.2.3 File Scan

when a user selects a particular file from the folder and uploads in the file scan.
The file scan scans the complete file which the user has uploaded and detects if the file contains any malicious info/link.

1.2.4 Hash Scan

Hash Scanner is a unique service that allows users to scan files regardless of their size using an MD5, SHA1 or SHA256 hash.
The results returned provide the count of antivirus detections by different AV engines.

1.2.5 Quick Scan

A quick scan checks common areas for computers viruses. Scanned areas can include the computer memory and common areas of the hard drive,
including the temporary Internet files and the operating system directory. A quick scan takes less than 5 minutes to complete.

Quick Scan checks the objects loaded at the operating system startup, boot sectors.

1.2.6 Full Scan

Full Scan performs a thorough check of the whole system.
Because the full scan checks everything, it takes longer to perform the scan.

Depending on how much memory and hard drive space a computer has, it could take 30 minutes to several hours to complete a full scan. By default, the following objects are scanned:

– system memory.
– programs loaded at startup.
– system backup.
– email databases.
– hard drives, removable storage media and network drives and sometimes any external devices connected to the computer, like external hard drives.

1.2.7 VPN

VPN stands for "Virtual Private Network" and describes the opportunity to establish a protected network connection when using public networks.
VPNs encrypt your internet traffic and disguise your online identity. This makes it more difficult for third parties to track your activities online and steal data.
The encryption takes place in real time

Chapter 2

02.LITERATURE SURVEY

2.1 INTEGRATION OF ANTIVIRUS WITH VIRTUAL PRIVATE NETWORK FOR PERSONAL SYSTEMS SECURITY.

Authors: Syed Shuja Ali Naqvi, Joshua Samuel.

Key Terms: Antivirus, VPN, Tunneling, Personal System Security, Encryption.

The aim of this research is to provide an integrated system that will protect the home users from intruders and other viruses and to explore all the technique that is designed to detect and remove software viruses.

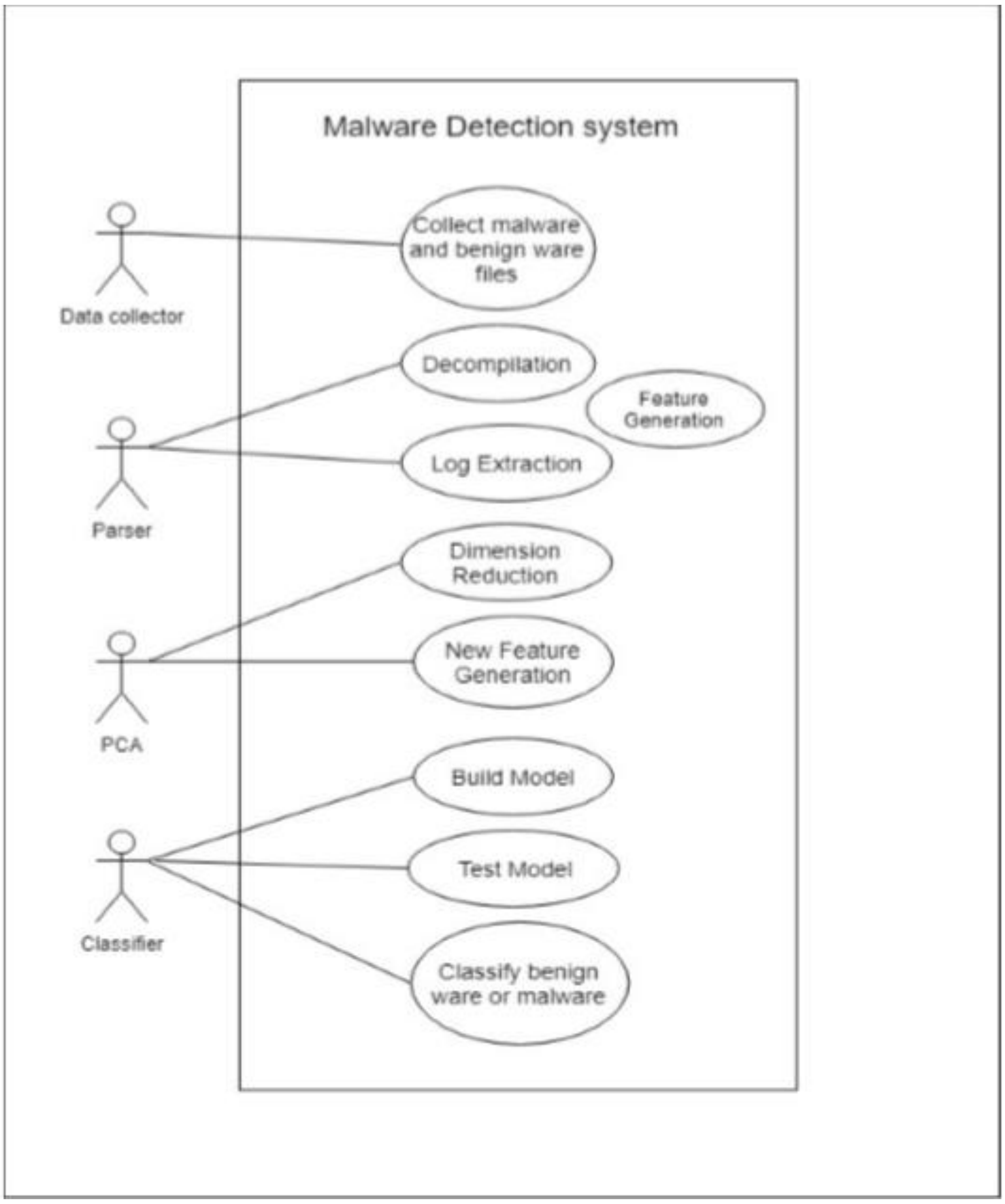
The core idea of this technology is to transmit data packages on the tunnel that can be formed by different tunneling protocols. Such as PPTP, and L2TP/IPsec.

Scope:
The scope of the proposed system is very vast when it comes to the systems security of home users as it can be implemented for any home user as for them security is a major concern.
It provides anti-malware protection, virtual private network with encryption, creates a safe and encrypted connection over a less secure network.

Chapter 3

3. SYSTEM DESGIN

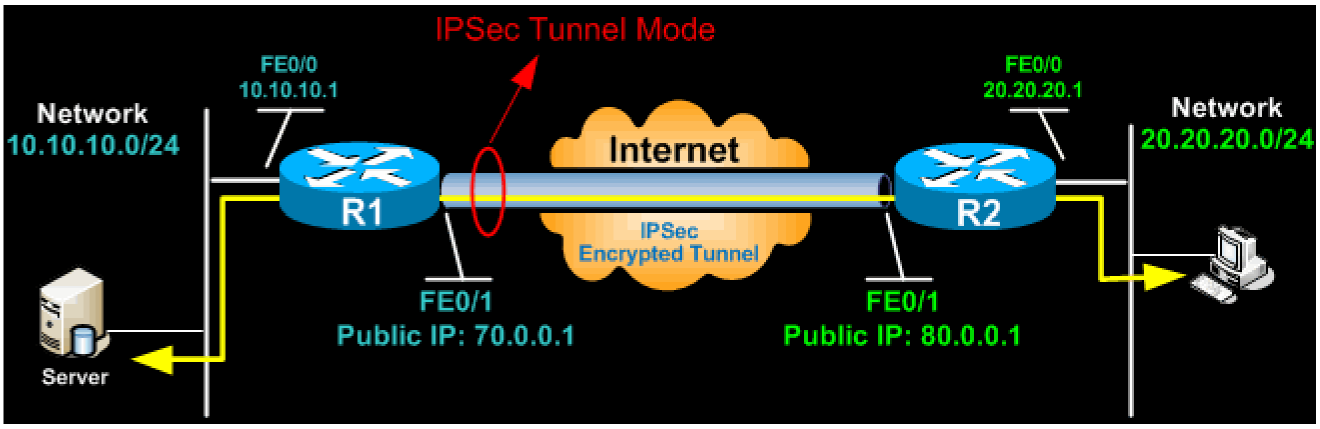
3.1 Machine Learning



(Fig.3.1)

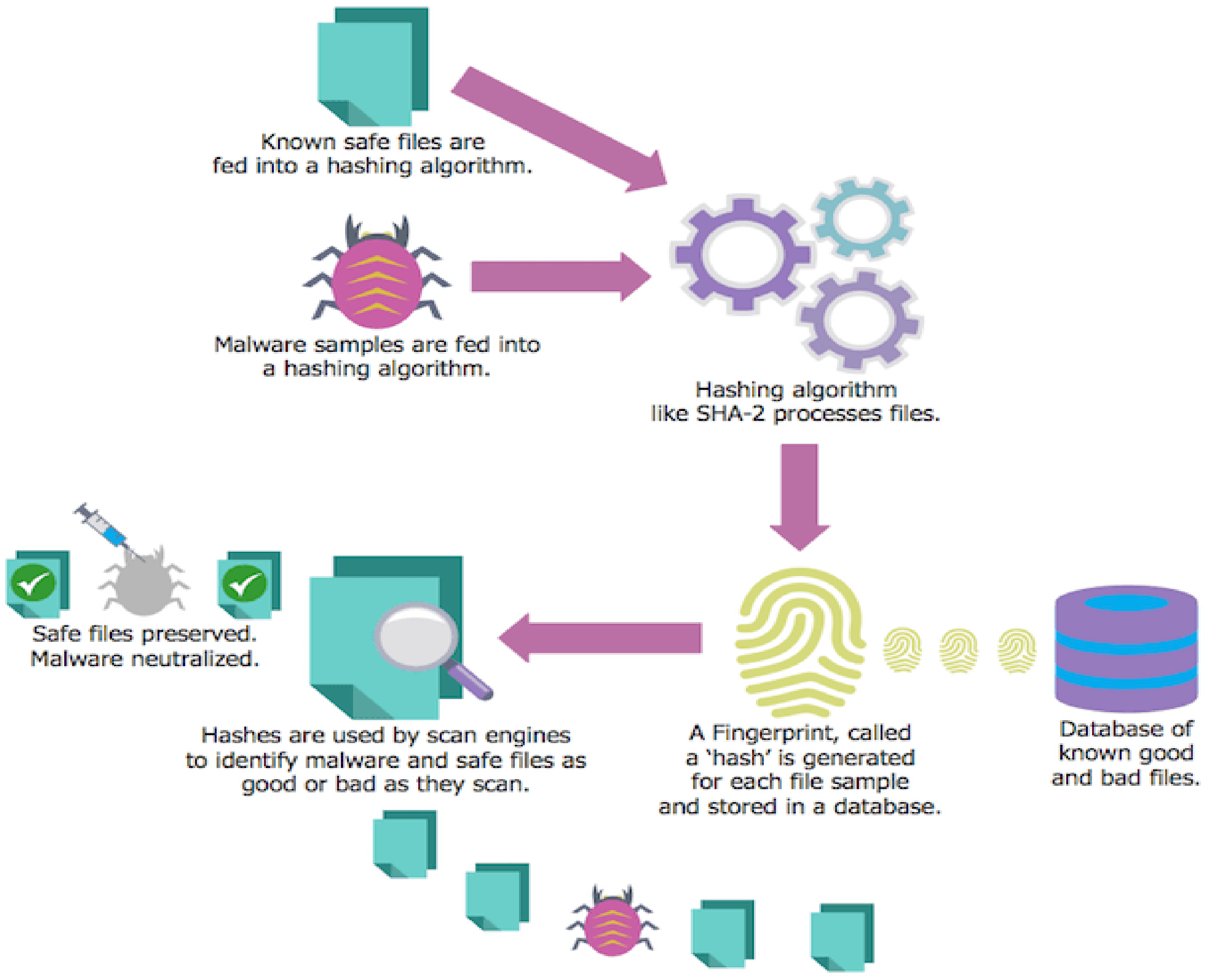
3.2 VPN

(Fig 3.2)

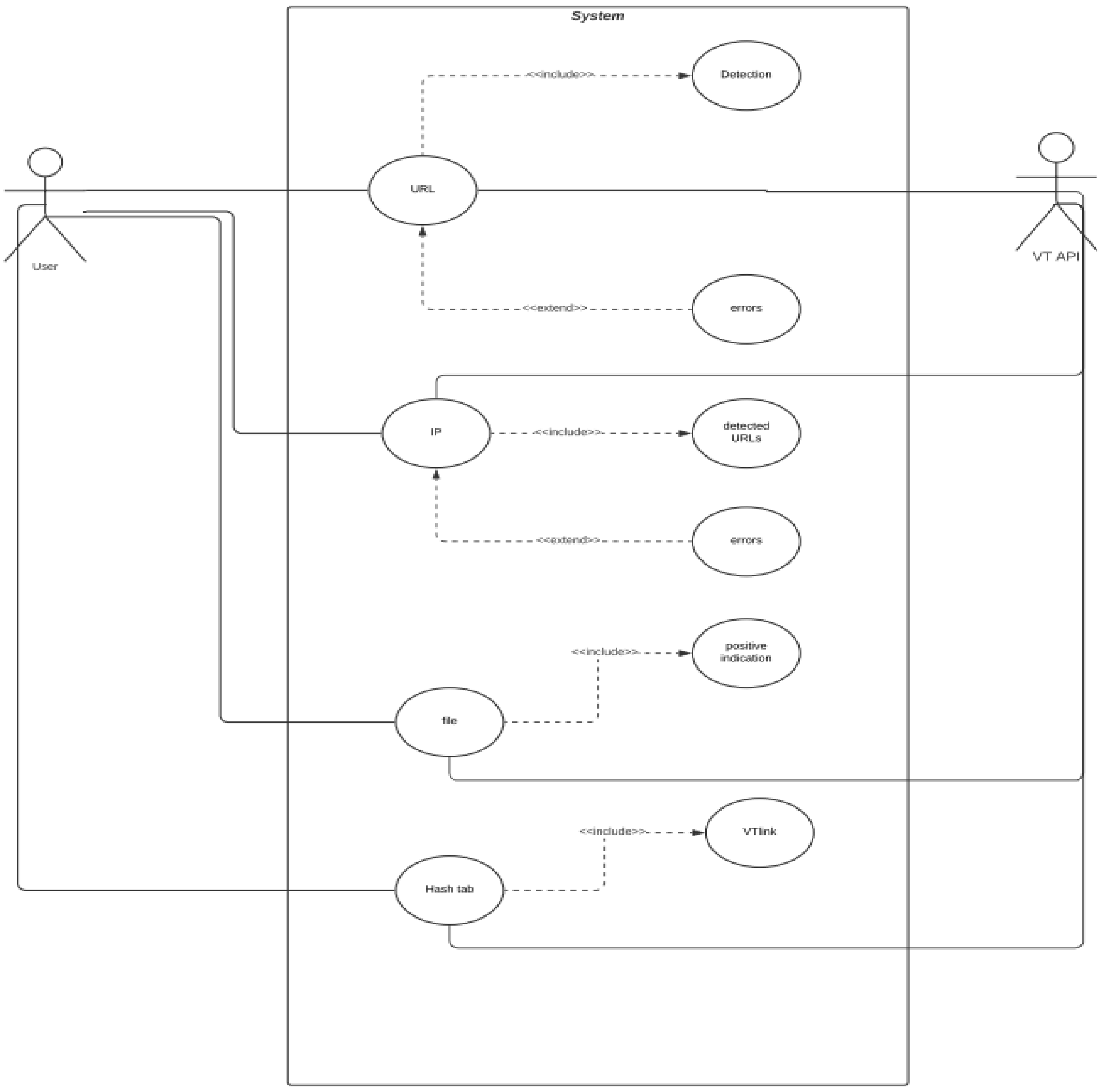


3.3 Signature Based

(Fig 3.3)


Known safe files are
fed into a hashing algorithm.
Malware samples are fed into
a hashing algorithm.
Hashing algorithm
like SHA-2 processes files.
Safe files preserved.
Malware neutralized.
Hashes are used by scan engines
to identify malware and safe files as
good or bad as they scan.
A Fingerprint, called
a 'hash' is generated
for each file sample
and stored in a database.
Database of
known good
and bad files.

3.4 Virus total



(Fig 3.4)

4. IMPLEMENTATION

In this chapter we have discussed about various tools and technologies used for implementing the antivirus as well as the API which are used as a third-party plugin for few modules

4.1 Tool and Environment:

- Programming languages: Python, Tkinter (GUI).

  Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built-in data structures, combined with dynamic typing and dynamic binding,

  Make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance.

  Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms and can be freely distributed.

  Tkinter (GUI) Python provides the standard library Tkinter for creating the graphical user interface for desktop-based applications.

  Developing desktop-based applications with python Tkinter is not a complex task.

- API: Virus-Total

  Application programming interfaces, or APIs, simplify software development and innovation by enabling applications to exchange data and functionality easily and securely.

  Virus Total’s API lets you upload and scan files, submit and scan URLs, access finished scan reports and make automatic comments on URLs and samples without the need of using the HTML website interface.

  In other words, it allows you to build simple scripts to access the information generated by Virus-Total.

- IDE: PyCharm community edition.

PyCharm is an integrated development environment (IDE) used in computer programming, specifically for the Python language.

It is developed by the Czech company JetBrains (formerly known as IntelliJ).It provides code analysis, a graphical debugger, an integrated unit tester, integration with version control systems (VCSes), and supports web development with Django as well as data science with Anaconda.

PyCharm is cross-platform, with Windows, macOS and Linux versions. The Community Edition is released under the Apache License and there is also Professional Edition with extra features – released under a proprietary license

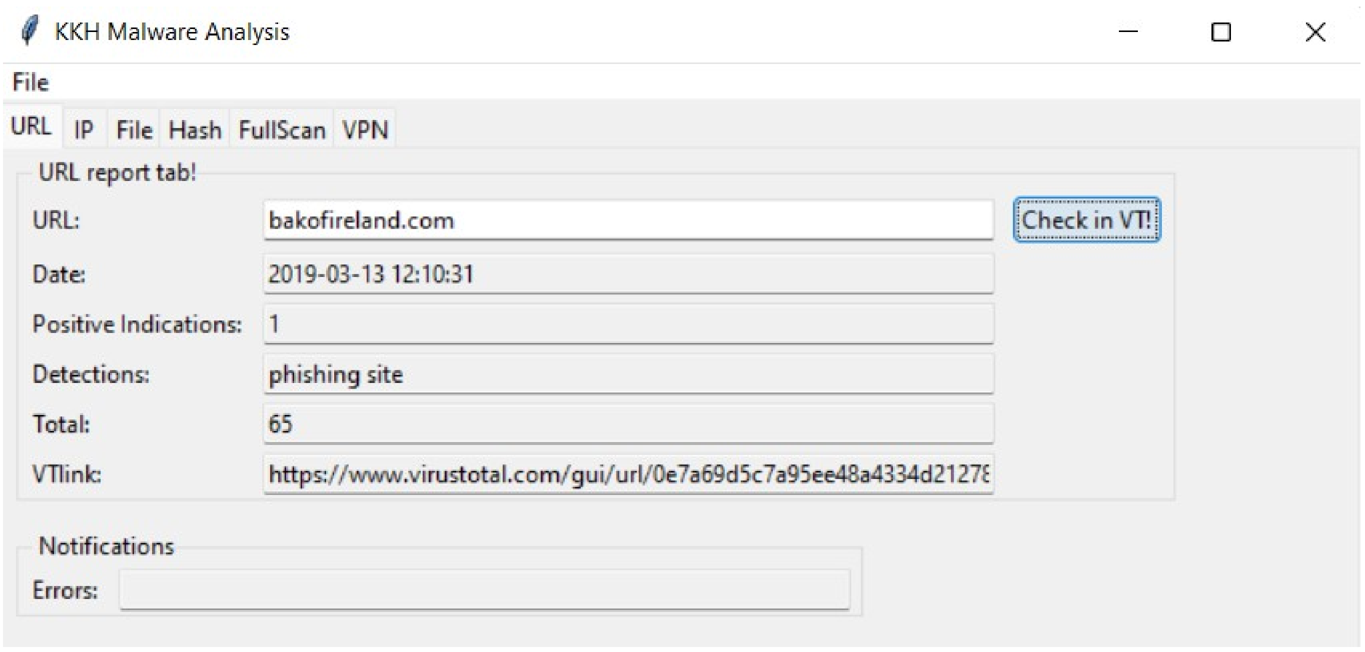
4.2. Modules:

- **import Tkinter:** The significance of "import *" **represents all the functions and built-in modules in the Tkinter library.**
- **import requests:** The requests module allows you to **send HTTP requests** using Python. The HTTP request returns a Response Object with all the response data (content, encoding, status, etc).
- **import configparser:** This module provides the Configparser class which implements a basic configuration language which provides a structure similar to what's found in Microsoft Windows INI files
- **import webbrowser:** The webbrowser module provides a high-level interface to allow displaying web-based documents to users.
- **import glob2:** The glob module is a useful part of the Python standard library. glob (short for global) is **used to return all file paths that match a specific pattern.**
- **import hashlib:** This module implements a common interface to many different **secure hash and message digest** algorithms.
- **import os:** The OS module in Python provides functions for creating and removing a directory (folder), fetching its contents, changing and identifying the current directory, etc. ... So, import it using the import os statement before using its functions.
- **import sys:** import sys in python is **loading the module named sys into the current namespace** so that you can access the functions and anything else defined within the module using the module name.
- **import subprocess:** Subprocess in Python is **a module used to run new codes and applications by creating new processes**.

Chapter 4

05.RESULT

5.1. URL.



URL report tab!

URL: bakofireland.com

Date: 2019-03-13 12:10:31

Positive Indications: 1

Detections: phishing site

Total: 65

VTlink: https://www.virustotal.com/gui/url/0e7a69d5c7a95ee48a4334d21278

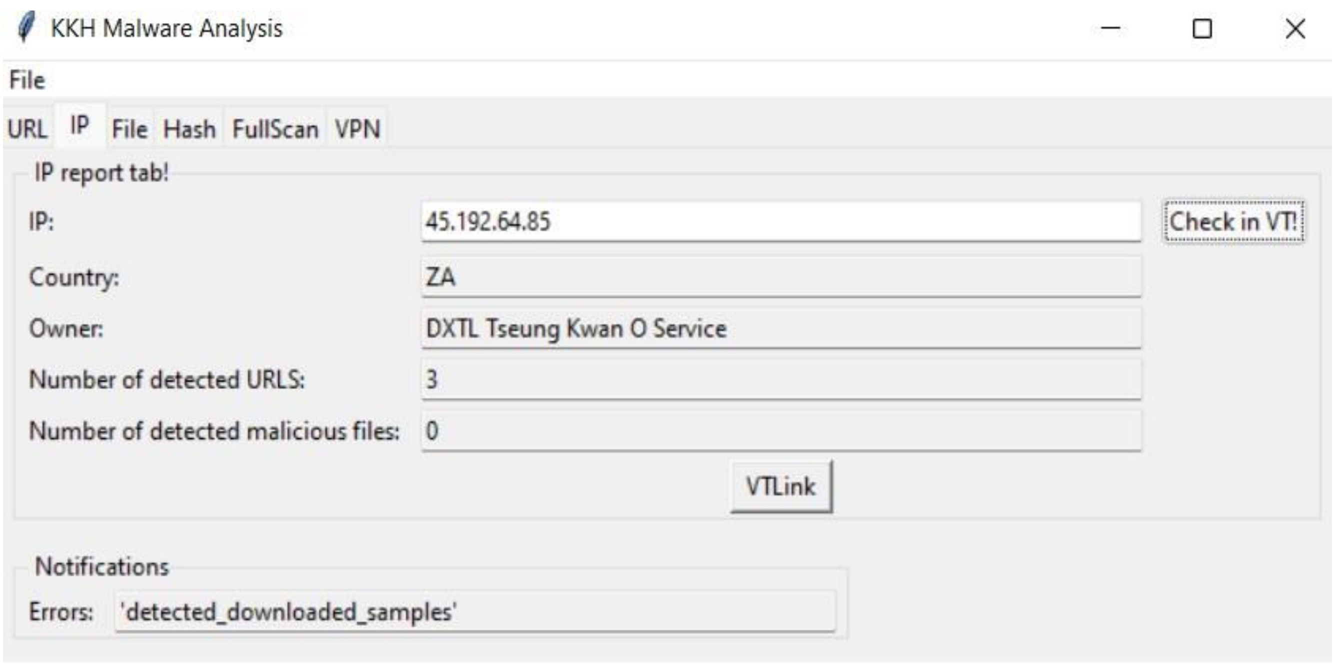
Notifications

Errors:

(Fig 5.1)

URL Scanner is used to quickly analyze if a link/URL is suspicious or unsafe. Detect malicious URLs checks whether the URL is safe from viruses, Phishing, Malware, Parked Domains.
After completion of the scan, it shows total number of scans performed and number of positive indications and detections from the URL.

5.2 IP



IP report tab!

IP: 45.192.64.85

Country: ZA

Owner: DXTL Tseung Kwan O Service

Number of detected URLS: 3

Number of detected malicious files: 0

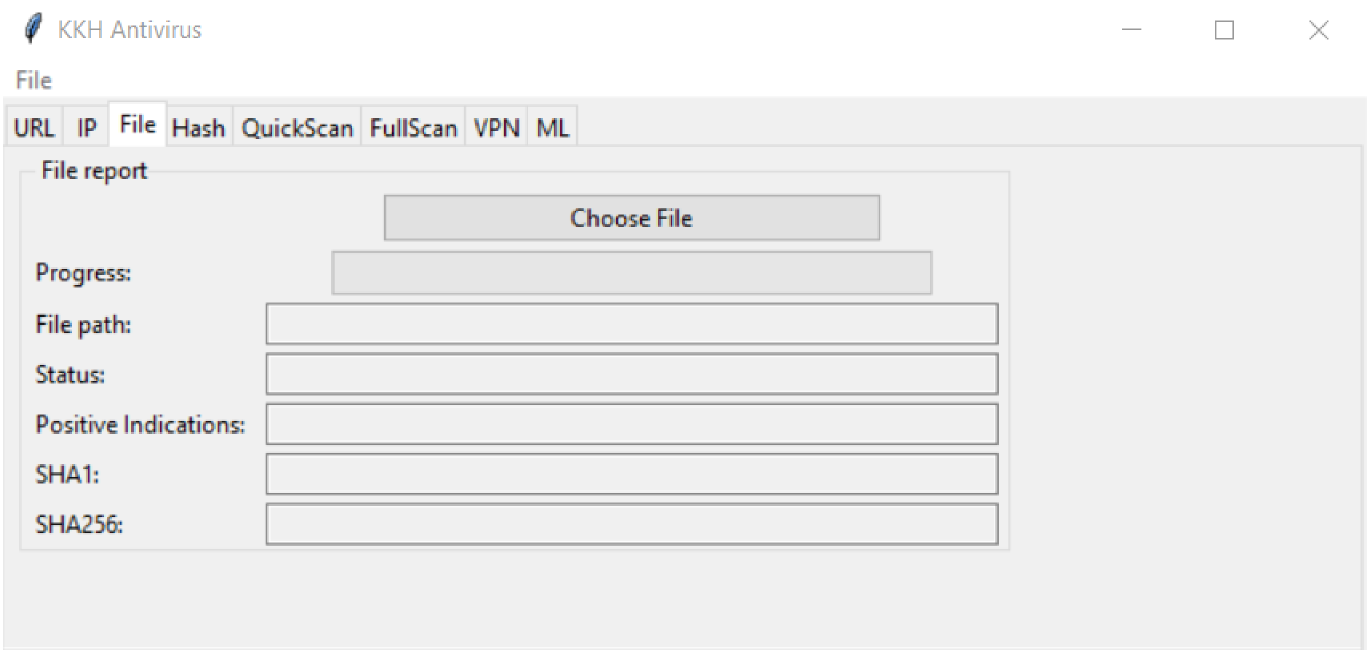
Notifications

Errors: 'detected_downloaded_samples'

(Fig 5.2)

IP scanner scans the IP address and checks whether the IP is safe from malwares. Ip scanner show the owner and country of the IP address, number of detected URLs and number of detected malicious files on the IP address.

5.3 FILE SCAN.



KKH Antivirus

File

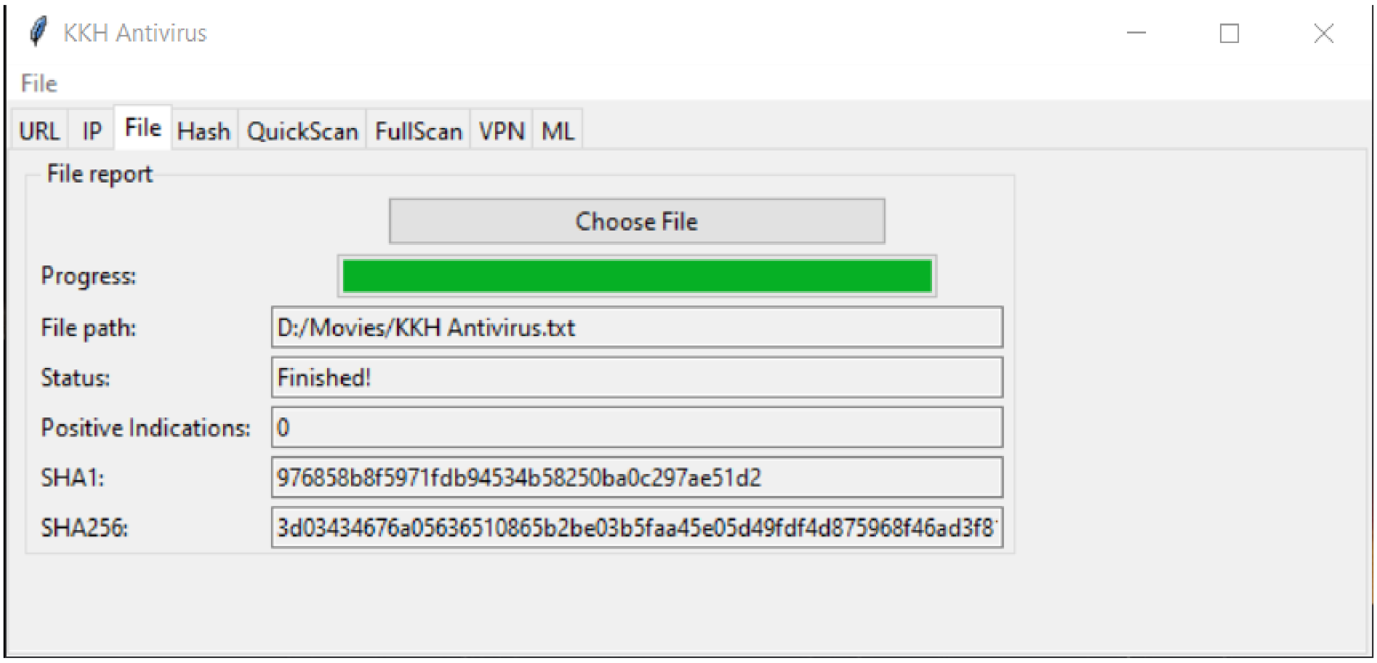
URL | IP | File | Hash | QuickScan | FullScan | VPN | ML

File report

Choose File

Progress:	
File path:	
Status:	
Positive Indications:	
SHA1:	
SHA256:	

(Fig 5.3.1)



KKH Antivirus

File

URL | IP | File | Hash | QuickScan | FullScan | VPN | ML

File report

Choose File

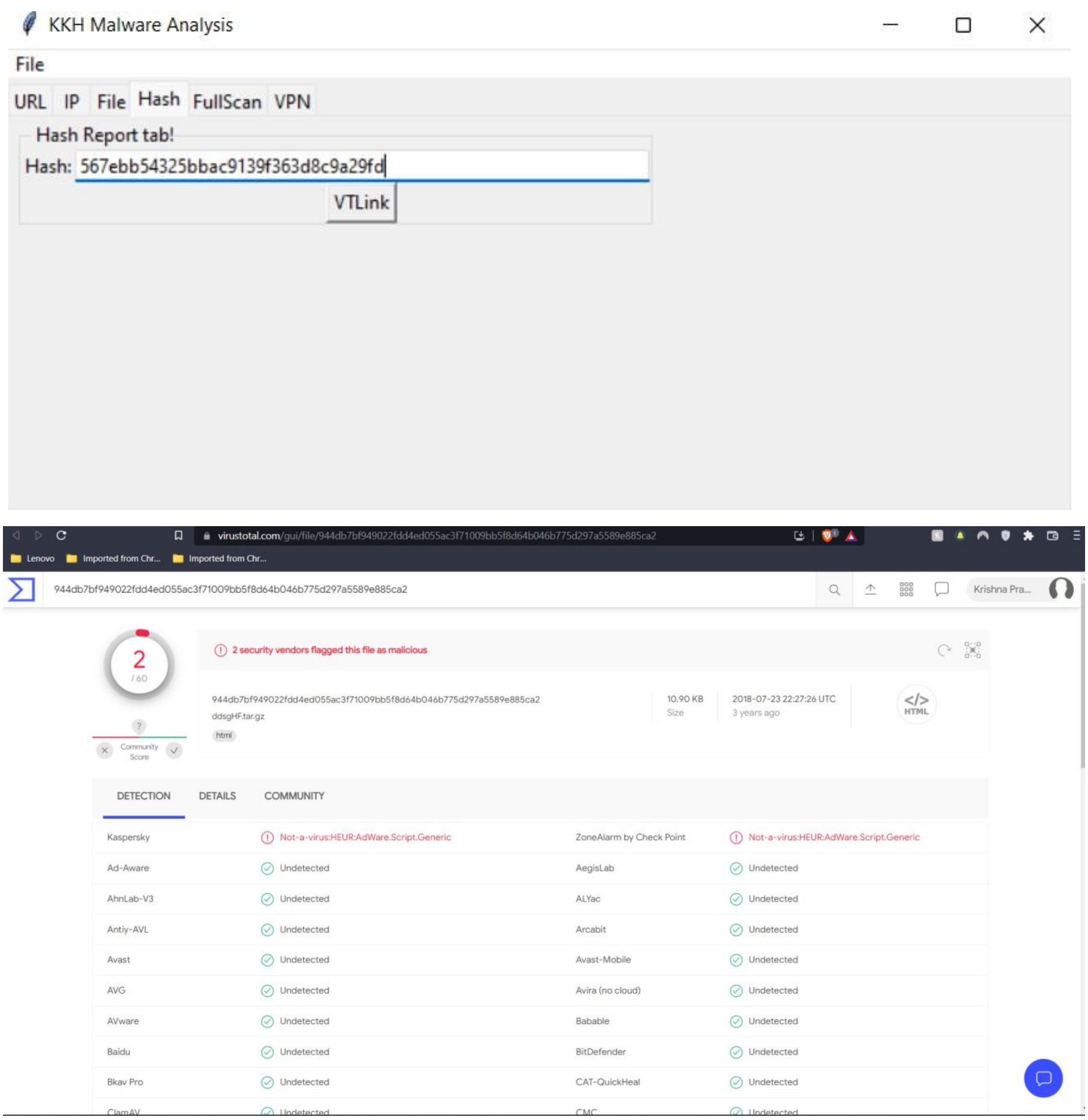
Progress:	🟩
File path:	D:/Movies/KKH Antivirus.txt
Status:	Finished!
Positive Indications:	0
SHA1:	976858b8f5971fdb94534b58250ba0c297ae51d2
SHA256:	3d03434676a05636510865b2be03b5faa45e05d49fdf4d875968f46ad3f8

(Fig 5.3.2)

when a user selects a particular file from the folder and uploads in the file scan.

The file scan scans the complete file which the user has uploaded and detects if the file contains any malicious info/link.

5.4 HASH



Hash: 567ebb54325bbac9139f363d8c9a29fd

virustotal.com/gui/file/944db7bf949022fdd4ed055ac3f71009bb5f8d64b046b775d297a5589e885ca2

944db7bf949022fdd4ed055ac3f71009bb5f8d64b046b775d297a5589e885ca2

2 / 60

2 security vendors flagged this file as malicious

944db7bf949022fdd4ed055ac3f71009bb5f8d64b046b775d297a5589e885ca2
ddsgHF.tar.gz
html

10.90 KB Size

2018-07-23 22:27:26 UTC 3 years ago

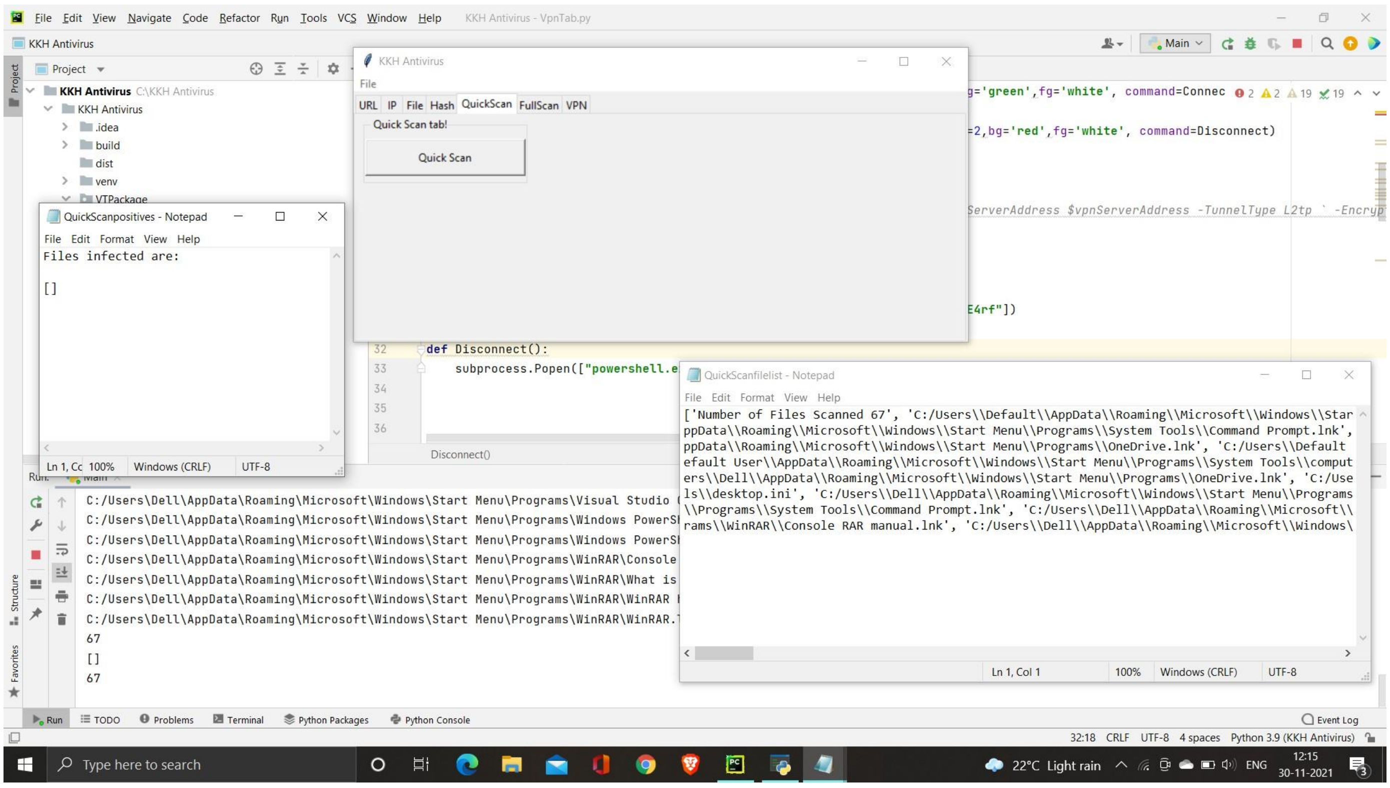
Kaspersky	Not-a-virus:HEUR:AdWare.Script.Generic	ZoneAlarm by Check Point	Not-a-virus:HEUR:AdWare.Script.Generic
Ad-Aware	Undetected	AegisLab	Undetected
AhnLab-V3	Undetected	ALYac	Undetected
Antiy-AVL	Undetected	Arcabit	Undetected
Avast	Undetected	Avast-Mobile	Undetected
AVG	Undetected	Avira (no cloud)	Undetected
AVware	Undetected	Babable	Undetected
Baidu	Undetected	BitDefender	Undetected
Bkav Pro	Undetected	CAT-QuickHeal	Undetected
ClamAV	Undetected	CMC	Undetected

(Fig5.4)

Hash Scanner is a unique service that allows users to scan files regardless of their size using an MD5, SHA1 or SHA256 hash.

The results returned provide the count of antivirus detections by different AV engines.

5.5 QUICK SCAN



```
Files infected are:

[]
```

```
['Number of Files Scanned 67', 'C:/Users\\Default\\AppData\\Roaming\\Microsoft\\Windows\\Star
ppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\System Tools\\Command Prompt.lnk',
ppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\OneDrive.lnk', 'C:/Users\\Default
efault User\\AppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\System Tools\\comput
ers\\Dell\\AppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs\\OneDrive.lnk', 'C:/Use
ls\\desktop.ini', 'C:/Users\\Dell\\AppData\\Roaming\\Microsoft\\Windows\\Start Menu\\Programs
\\Programs\\System Tools\\Command Prompt.lnk', 'C:/Users\\Dell\\AppData\\Roaming\\Microsoft\\
rams\\WinRAR\\Console RAR manual.lnk', 'C:/Users\\Dell\\AppData\\Roaming\\Microsoft\\Windows\
```

```
def Disconnect():
    subprocess.Popen(["powershell.e
```

```
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Visual Studio (
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Windows PowerSh
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Windows PowerSh
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\Console
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\What is
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\WinRAR
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\WinRAR.
67
[]
67
```

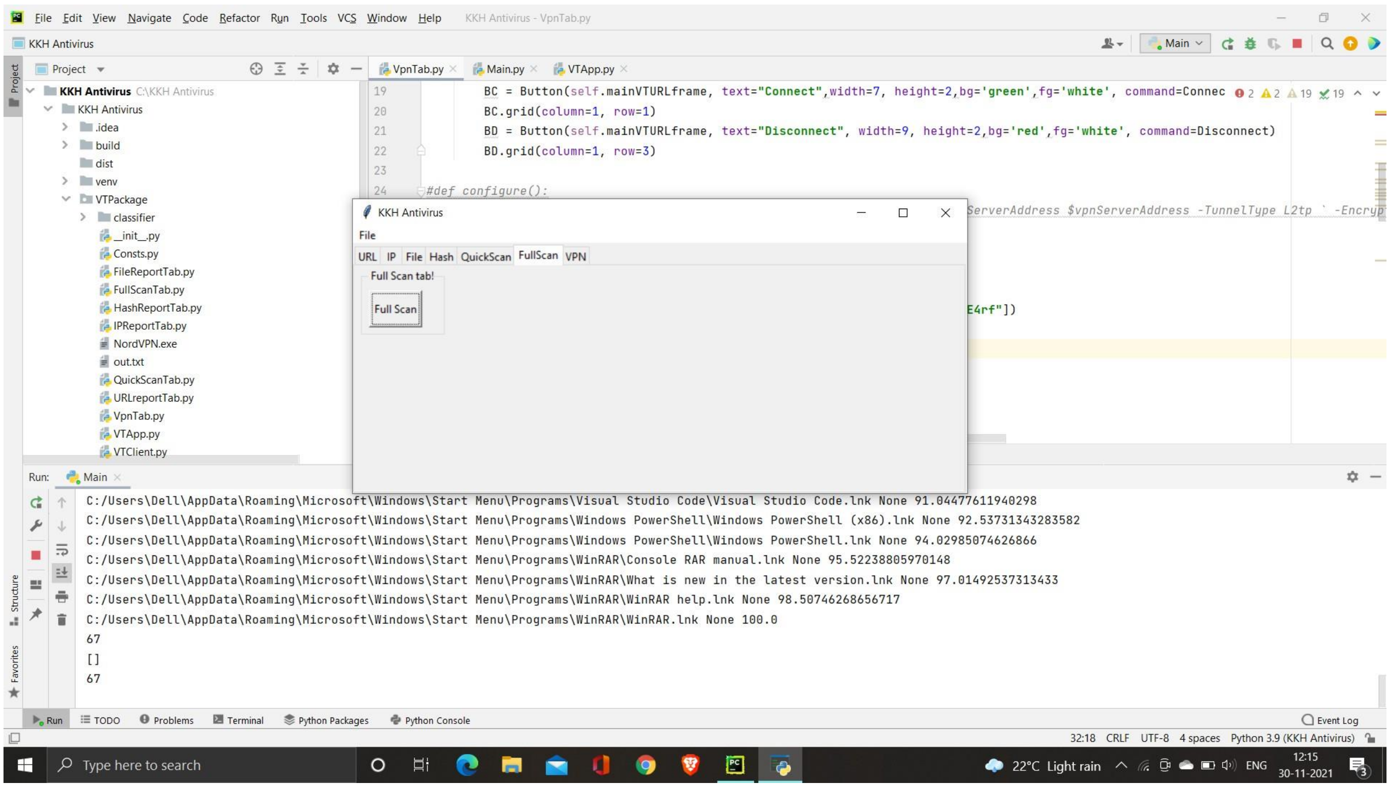
(Fig 5.5)

A quick scan checks common areas for computers viruses.

Scanned areas can include the computer memory and common areas of the hard drive, including the temporary Internet files and the operating system directory.

A quick scan takes less than 5 minutes to complete.

5.6 FULL SCAN



```
        BC = Button(self.mainVTURLframe, text="Connect",width=7, height=2,bg='green',fg='white', command=Connec
        BC.grid(column=1, row=1)
        BD = Button(self.mainVTURLframe, text="Disconnect", width=9, height=2,bg='red',fg='white', command=Disconnect)
        BD.grid(column=1, row=3)

#def configure():
```

```
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Visual Studio Code\Visual Studio Code.lnk None 91.04477611940298
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Windows PowerShell\Windows PowerShell (x86).lnk None 92.53731343283582
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Windows PowerShell\Windows PowerShell.lnk None 94.02985074626866
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\Console RAR manual.lnk None 95.52238805970148
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\What is new in the latest version.lnk None 97.01492537313433
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\WinRAR help.lnk None 98.50746268656717
C:/Users\Dell\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\WinRAR\WinRAR.lnk None 100.0
67
[]
67
```

(Fig 5.6)

Full Scan performs a thorough check of the whole system.
Because the full scan checks everything, it takes longer to perform the scan.

Depending on how much memory and hard drive space a computer has, it could take 30 minutes to several hours to complete a full scan. By default, the following objects are scanned:

– system memory.

– programs loaded at startup.

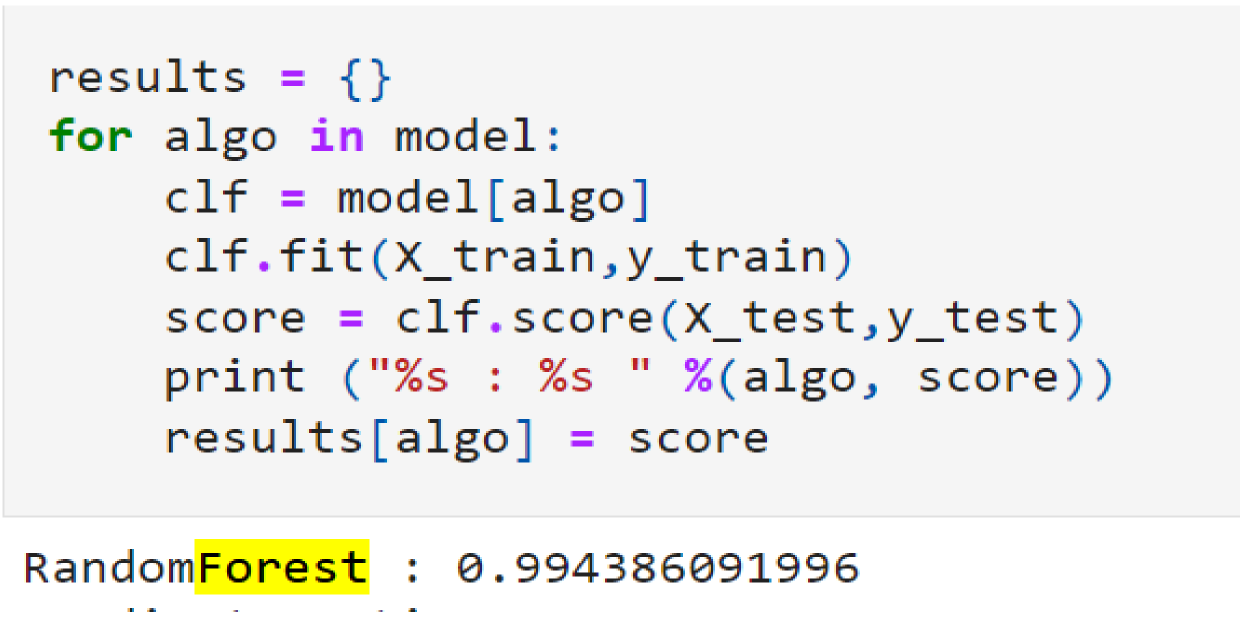
– system backup.

– email databases.

– hard drives, removable storage media and network drives and sometimes any external devices connected to the computer, like external hard drives.

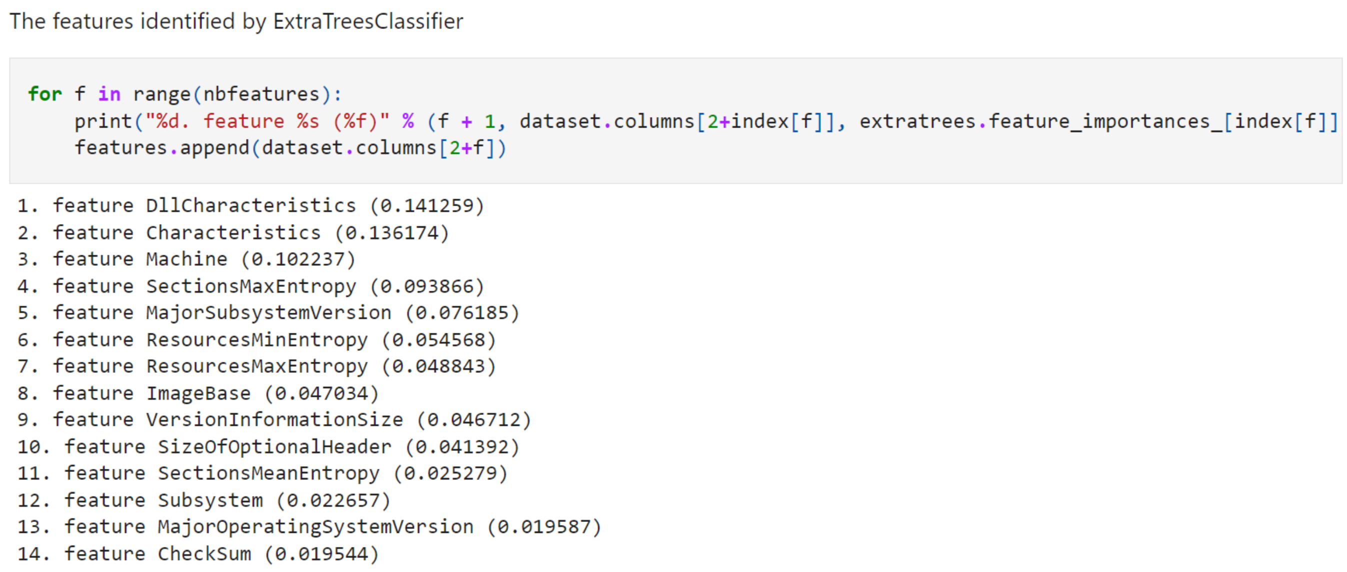
5.7 Machine Learning

(Fig 5.7.1)



```
results = {}
for algo in model:
    clf = model[algo]
    clf.fit(X_train,y_train)
    score = clf.score(X_test,y_test)
    print ("%s : %s " %(algo, score))
    results[algo] = score
```

```
RandomForest : 0.994386091996
```



The features identified by ExtraTreesClassifier

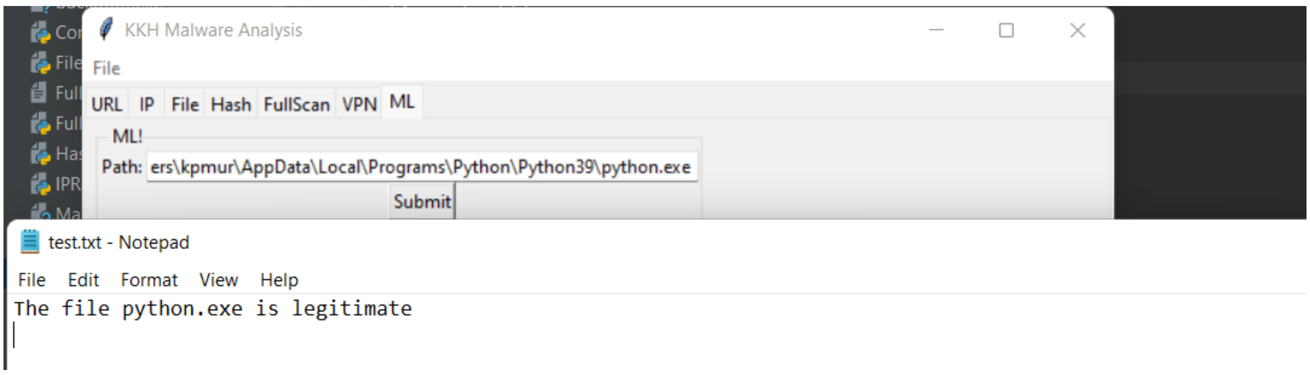
```
for f in range(nbfeatures):
    print("%d. feature %s (%f)" % (f + 1, dataset.columns[2+index[f]], extratrees.feature_importances_[index[f]]
    features.append(dataset.columns[2+f])
```

```
1. feature DllCharacteristics (0.141259)
2. feature Characteristics (0.136174)
3. feature Machine (0.102237)
4. feature SectionsMaxEntropy (0.093866)
5. feature MajorSubsystemVersion (0.076185)
6. feature ResourcesMinEntropy (0.054568)
7. feature ResourcesMaxEntropy (0.048843)
8. feature ImageBase (0.047034)
9. feature VersionInformationSize (0.046712)
10. feature SizeOfOptionalHeader (0.041392)
11. feature SectionsMeanEntropy (0.025279)
12. feature Subsystem (0.022657)
13. feature MajorOperatingSystemVersion (0.019587)
14. feature CheckSum (0.019544)
```

(Fig 5.7.2)

A Machine Learning approach for classifying a file as Malicious or Legitimate.
This approach tries out 6 different classification algorithms before deciding which one to use for prediction by comparing their results RandomForest.

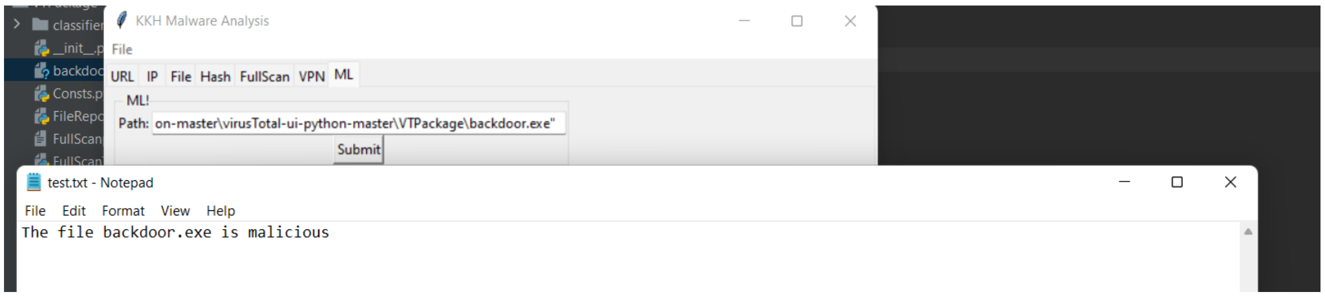
In order to test the model on an unseen file, it's required to extract the characteristics of the given file. Python's pefile.PE library is used to construct and build the feature vector and a ML model is used to predict the class for the given file based on the already trained model.



Path: ers\kpmur\AppData\Local\Programs\Python\Python39\python.exe

```
The file python.exe is legitimate
```

(Fig 5.7.3)



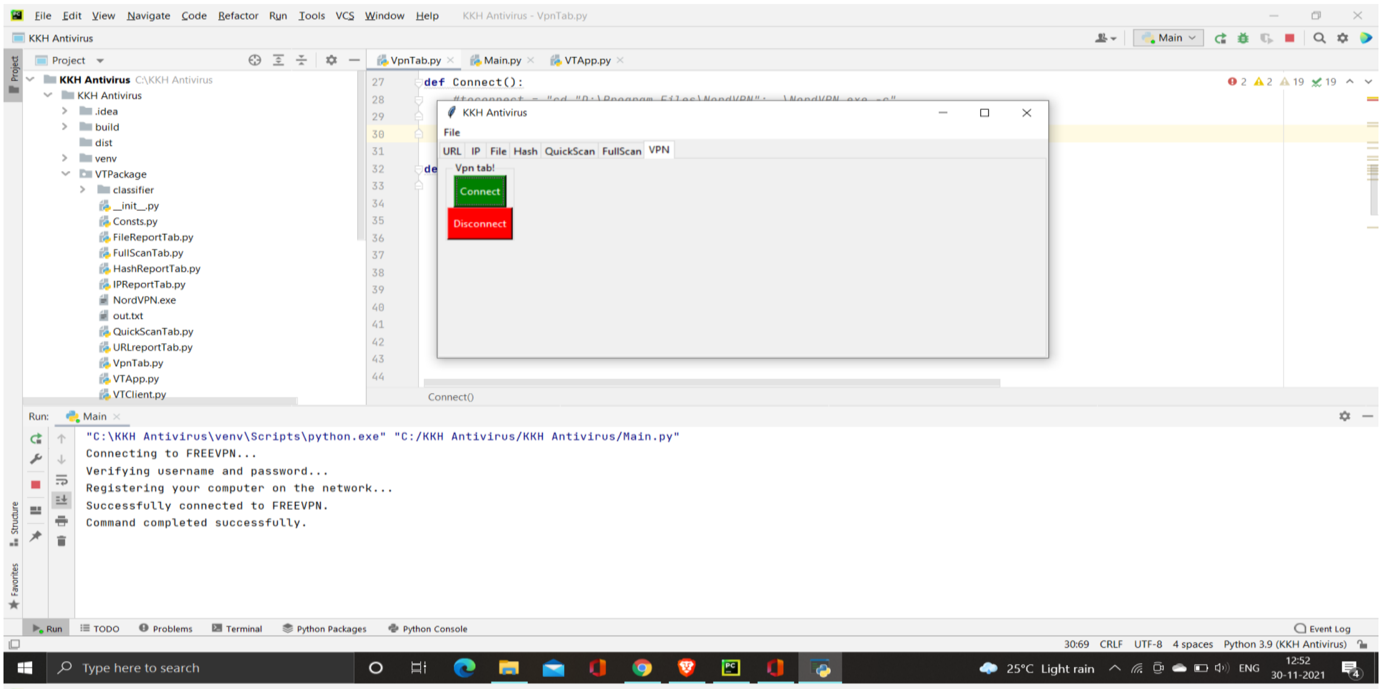
Path: on-master\virusTotal-ui-python-master\VTPackage\backdoor.exe"

```
The file backdoor.exe is malicious
```

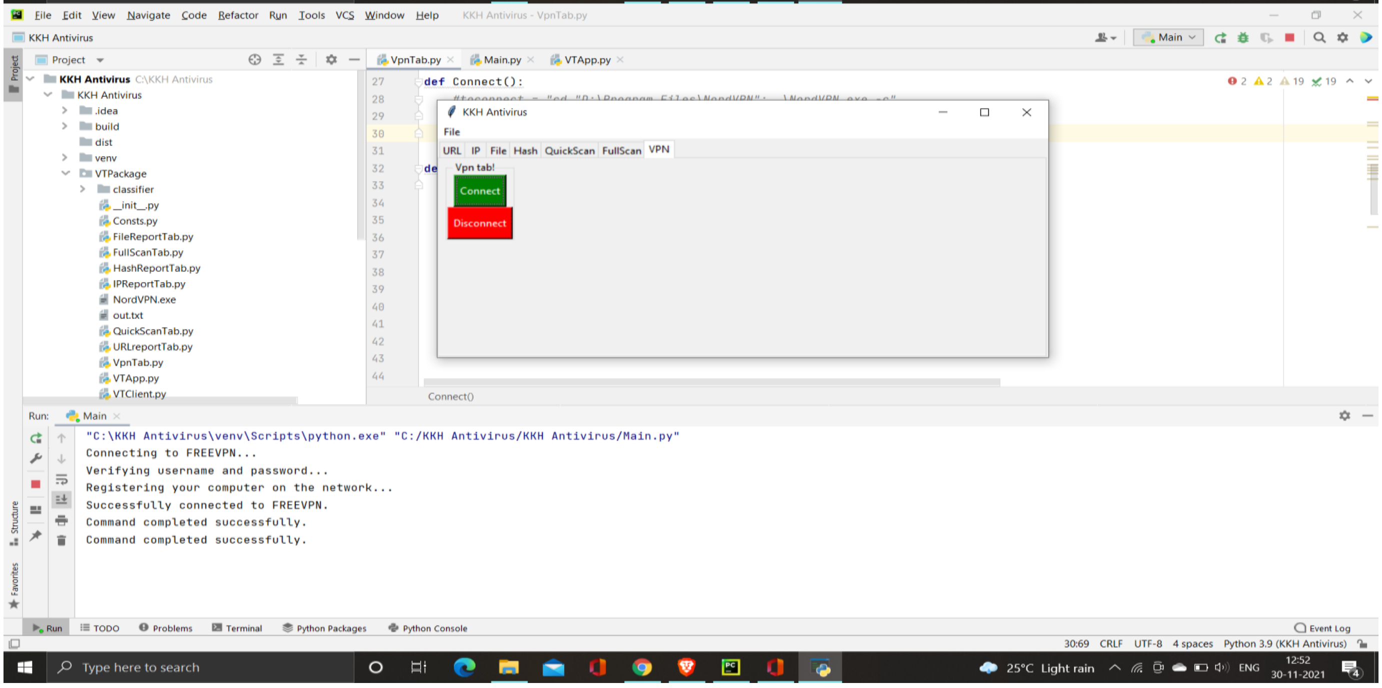
(Fig 5.7.4)

5.8 VPN

(Fig 5.8)



```
"C:\KKH Antivirus\venv\Scripts\python.exe" "C:/KKH Antivirus/KKH Antivirus/Main.py"
Connecting to FREEVPN...
Verifying username and password...
Registering your computer on the network...
Successfully connected to FREEVPN.
Command completed successfully.
```



```
"C:\KKH Antivirus\venv\Scripts\python.exe" "C:/KKH Antivirus/KKH Antivirus/Main.py"
Connecting to FREEVPN...
Verifying username and password...
Registering your computer on the network...
Successfully connected to FREEVPN.
Command completed successfully.
Command completed successfully.
```

VPN stands for "Virtual Private Network" and describes the opportunity to establish a protected network connection when using public networks.

VPNs encrypt your internet traffic and disguise your online identity. This makes it more difficult for third parties to track your activities online and steal data. The encryption takes place in real time.

CHAPTER-4

06.CONCLUSION: -

Antivirus software acts as the final line of defense for your PC(S) and other devices. Which means it can protect – or at least mitigate threats – your devices when every other security software fails.

Therefore, do not underestimate its potential and never leave your devices lying around without antivirus in them. Install antivirus, stay protected against the modern-day malware!

07.REFERENCES

[1] http://www.us-cert.gov/control_systems/pdf/undirected_attack0905.pdf
[2] "Defining Malware: FAQ". http://technet.microsoft.com. Retrieved 2009-09-10.
[3] F-Secure Corporation (December 4, 2007). "F-Secure Reports Amount of Malware Grew by 100% during 2007". Press release. Retrieved 2007-12-11.
[4] History of Viruses. http://csrc.nist.gov/publications/nistir/threats/subsubsection3_3_1_1.html.
[5] Landesman, Mary (2009). "What is a Virus Signature?" Retrieved 2009-06-18.
[6] Christodorescu,M., Jha, S., 2003. Static analysis of executables to detect malicious patterns. In: Proceedings of the 12th USENIX Security Symposium. Washington .pp. 105-120.
[7] Filiol, E.,2005. Computer Viruses: from Theory to Applications. New York, Springer, ISBN 10: 2-287- 23939- 1.
[8] Filiol, E., Jacob, G., Liard, M.L., 2007: Evaluation methodology and theoretical model for antiviral behavioral detection strategies. J. Comput. 3, pp 27–37.
[9] H. Witten and E. Frank. 2005. Data mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, ISBN-10: 0120884070.
[10] J. Kolter and M. Maloof, 2004. Learning to detect malicious executables in the wild. In Proceedings of KDD'04, pp 470-478.
[11] J. Wang, P. Deng, Y. Fan, L. Jaw, and Y. Liu, 2003.Virus detection using data mining techniques. In Proceedings of IEEE International Conference on Data Mining.
[12] Kephart, J., Arnold, W., 1994. Automatic extraction of computer virus signatures. In: Proceedings of 4th Virus Bulletin International Conference, pp. 178–184.
[13] L. Adleman, 1990. An abstract theory of computer viruses (invited talk). CRYPTO '88: Proceedings on Advances in Cryptology, New York, USA. Springer, pp: 354–374.
[14] Lee, T., Mody, J., 2006.Behavioral classification. In: Proceedings of European Institute for Computer
[15] Machine Learning for Cybersecurity Cookbook Nov 2019

08.APPENDIX

8.1 CODE:

- **Load classifier**

```
clf =
joblib.load(os.path.join(os.path.dirname(os.path.realpath(_file_)),'classifier/classifier.pkl'))
 features =

pickle.loads(open(os.path.join(os.path.dirname(os.path.realpath(_file_)),'classifier/features.p
kl'), 'rb').read())

    data = extract_infos(args.FILE)


    pe_features = list(map(lambda x: data[x], features))


    res = clf.predict([pe_features])[0]
```

• Learning using datasets

```
# Importing the libraries
import pandas as pd
import numpy as np
import pickle
import sklearn.ensemble as ske
from sklearn import tree, linear_model
from sklearn.feature_selection import SelectFromModel
import joblib
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split


# Importing the dataset
data = pd.read_csv('data.csv', sep='|')
X = data.drop(['Name', 'md5', 'legitimate'], axis=1).values
y = data['legitimate'].values


print('Researching important feature based on %i total features\n' % X.shape[1])


# Feature selection using Trees Classifier
fsel = ske.ExtraTreesClassifier().fit(X, y)
model = SelectFromModel(fsel, prefit=True)
X_new = model.transform(X)
nb_features = X_new.shape[1]


# Splitting the dataset into the Training set and Test set
X_train, X_test, y_train, y_test = train_test_split(X_new, y ,test_size=0.2)


features = []


print('%i features identified as important:' % nb_features)
```

```
indices = np.argsort(fsel.feature_importances_)[::-1][:nb_features]
for f in range(nb_features):
    print("%d. feature %s (%f)" % (f + 1, data.columns[2+indices[f]],
fsel.feature_importances_[indices[f]]))


# Take care of the feature order
for f in sorted(np.argsort(fsel.feature_importances_)[::-1][:nb_features]):
    features.append(data.columns[2+f])


#Algorithm comparison
algorithms = {
        "DecisionTree": tree.DecisionTreeClassifier(max_depth=10),
        "RandomForest": ske.RandomForestClassifier(n_estimators=50),
    }


results = {}
print("\nNow testing algorithms")


# Fitting Classification algorithms to the Training set
for algo in algorithms:
    clf = algorithms[algo]
    clf.fit(X_train, y_train)
    score = clf.score(X_test, y_test)
    print("%s : %f %%" % (algo, score*100))
    results[algo] = score


winner = max(results, key=results.get)
print('\nWinner algorithm is %s with a %f %% success' % (winner, results[winner]*100))


# Save the algorithm and the feature list for later predictions
print('Saving algorithm and feature list in classifier directory...')
joblib.dump(algorithms[winner], 'classifier/classifier.pkl')
open('classifier/features.pkl', 'w').write(pickle.dumps(features))
print('Saved')

# Predicting the Test set results
y_pred = clf.predict(X_test)
```

- Testing and importing file metadata

```
def extract_infos(fpath):
    res = {}
    pe = pefile.PE(fpath)
    res['Machine'] = pe.FILE_HEADER.Machine
    res['SizeOfOptionalHeader'] = pe.FILE_HEADER.SizeOfOptionalHeader
    res['Characteristics'] = pe.FILE_HEADER.Characteristics
    res['MajorLinkerVersion'] = pe.OPTIONAL_HEADER.MajorLinkerVersion
    res['MinorLinkerVersion'] = pe.OPTIONAL_HEADER.MinorLinkerVersion
    res['SizeOfCode'] = pe.OPTIONAL_HEADER.SizeOfCode
    res['SizeOfInitializedData'] = pe.OPTIONAL_HEADER.SizeOfInitializedData
    res['SizeOfUninitializedData'] = pe.OPTIONAL_HEADER.SizeOfUninitializedData
    res['AddressOfEntryPoint'] = pe.OPTIONAL_HEADER.AddressOfEntryPoint
    res['BaseOfCode'] = pe.OPTIONAL_HEADER.BaseOfCode

parser = argparse.ArgumentParser(description='Detect malicious files')
    parser.add_argument('FILE', help='File to be tested')
    args = parser.parse_args()
    # Load classifier
    clf = joblib.load(os.path.join(
        os.path.dirname(os.path.realpath(_file_)),
        'classifier/classifier.pkl'
    ))
    features = pickle.loads(open(os.path.join(os.path.dirname(os.path.realpath(_file_)),
'classifier/features.pkl'), 'rb').read())


    data = extract_infos(sys.argv[1])


    pe_features = map(lambda x: data[x], features)


    res = clf.predict([pe_features])[0]
```

- Full scan and Quick scan

```
with ThreadPoolExecutor() as threads:
    filenames = glob2.glob('C:/*//.*', recursive=True)


  final_list = []
  count = 0


  def md5_search(fname):
    hash_md5 = hashlib.md5()
    if os.path.isfile(fname):
      assert os.path.isfile(fname)
      if fname != "*.tmp":
        try:
          with open(fname, "rb") as f:
            for chunk in iter(lambda: f.read(2 ** 20), b""):
              hash_md5.update(chunk)
            search_word = hash_md5.hexdigest()
            f.close
            with open('C:/Users\kpmur\Downloads\VirusShare_Copy.txt') as file1:
              contents = file1.read()
              if search_word in contents:
                os.remove(fname)
                final_list.append(fname + "\n" + "->" + search_word)
                return ("not safe " + search_word)
              else:
                return ("safe")
        except Exception:
          pass
```