B.Tech Project Report

QuickSMA

BACHELOR OF TECHNOLOGY IN

INFORMATION TECHNOLOGY

Submitted By:

Nitish Kumar (2K19/IT/093)

Naveen Kumar (2K19/IT/087)

Under the Guidance Of:

Dr. Seba Susan

Professor DTU

DEPARTMENT OF INFORMATION TECHNOLOGY

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering), Bawana Road,

Delhi-110042

November-2021

GitHub: https://github.com/Nitish9711/QuickSMA

CANDIDATE'S DECLARATION

We, hereby declare that the project work entitled "QuickSMA" submitted by **Nitish Kumar(2k19/IT/93)** and **Naveen Kumar(2K19/IT/87)** to the Department of Information Technology, Delhi Technological University is a record of bonafide.

Project work carried out by us under the guidance of Dr. Seba Susan and this project is submitted in the partial fulfillment of the requirements for the awards of the degree of Bachelor of Technology in Information Technology. The results embodied in this thesis have not been submitted to any other University or Institute for the award of any diploma or degree.

Delhi Naveen Kumar

Date: November,19,2021 Nitish Kumar

CERTIFICATE

This is to certify that the project entitled, “QuickSMA" submitted by Nitish Kumar and Naveen Kumar in partial fulfillment of the requirements for the award of Bachelor of Technology in Information Technology under the guidance of Dr. Seba Susan is an authentic work carried out by them under my supervision and guidance. To the best of my knowledge all the work has been done on their own and has not been copied from elsewhere without proper citation.

Delhi   Dr. Seba Susan

Date: November,19,2021   Professor D.T.U

ACKNOWLEDGEMENT

We have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. We would like to extend our sincere thanks to all of them.

We are highly indebted to our professor Dr. Seba Susan for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project. We would like to express our gratitude towards our parents & members of Delhi Technological University for their kind cooperation and encouragement which helped us in completion of this project.

We would like to express our special gratitude and thanks to industry persons for giving us such attention and time.

Our thanks and appreciation also go to our colleague in developing the project and people who have willingly helped us out with their abilities.

TABLE OF CONTENTS

INTRODUCTION

Executables are the programs packed into an executable package as one whole package for the purpose of Installation on the host device.
Many times these come from an untrusted source, and it is unknown whether they are genuine executables or are tampered with or are themselves some sort of malware. Static Malware Analysis is a technique where malware can be analyzed without running it.

Basic Static analysis involves techniques such as Analyzing raw strings in the executables that might include some malicious IP addresses, pieces of malicious codes, definite patterns of code, DLL's imports, API calls etc.

Some DLL imports are specific to some specific type of malware.
eg: "winsock.DLL" is a DLL which provides functions which help a program(malware) to connect to any remote host via socket. Therefore, presence of these types of DLLs in strings can be quite handy in detecting or analyzing malicious executables.

Many-a-time executables are in packed format so very few strings are produced for these types of executables and number of DLL imports for such files are less which might indicate their Packed nature.

2. Problem Description

When we analyse an executable and generate strings for it, the number of strings produced are very large.

Also a large proportion of the string generated for an executable involves strings which represent memory addresses or other portions of code that are interpreted by the ‘string module’ as raw string which actually are not useful in any way for us to perform the static analysis of the executable.

So we aim to design an approach to efficiently extract all the DLL imports and malicious IP addresses from the large number of strings so that we can easily analyse large text files for useful information.

3. Proposed Approach

DLLs contain executable code. That's their whole purpose. The only difference between an executable file and a DLL is that a DLL is loaded into memory dynamically, as needed by an application, and a single DLL might be shared by multiple applications.

There is nothing magical about DLLs to make them somehow immune from infection. They are just as vulnerable as EXE files. Executable code is executable code. Distribution of malware via an infected DLL is actually very common.

We Aim to use 'REGEX module' to solve this problem since REGEX modules help in efficient searching of strings and patterns from any given text with Cross Platform GUI support. Also to assist client in analysing DLL's information extracted from executables we have performed Web scraping which provides a brief information regarding each DLL imported by the executable and also provides appropriate URLs for further investigation.

3.1 Assumptions

This Program might not work efficiently in case of compressed/packed executables and also it might not work adequately in case of executables that involve Runtime Linking (because here DLL imports are runtime) and can't be analysed statically.

3.2 Proposed Algorithm

1. Ask the client to upload the executable at the GUI provided.

2. Extract all strings from executable and store them in temp.txt file.

3. Extract all DLL imports and IP addresses using the REGEX module and temp.txt file is deleted from client' system.

4. Then using web-scraping brief information about DLL imports and url links for further reading are provided on GUI.

4. Implementation Details

We have developed an application named QuickSMA. QuickSMA is an implementation of static malware analysis.

Basic Static analysis involves techniques such as Analysing raw strings in the executables that might include some malicious IP addresses , pieces of malicious codes, definite patterns of code, DLL's imports, API calls etc.

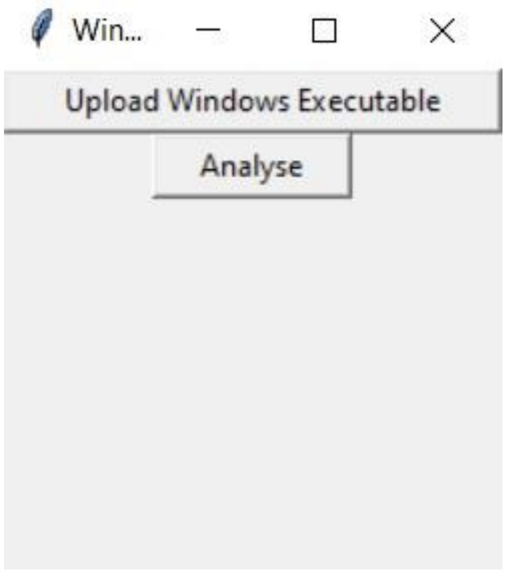
Some DLL imports are specific to some specific type of malware. eg : "winsock.DLL" is a DLL which provides functions which help a program(malware) to connect to any remote host via socket. Therefore, presence of these types of DLL's in strings can be quite handy in detecting or analysing malicious executables.

TOOLS USED:

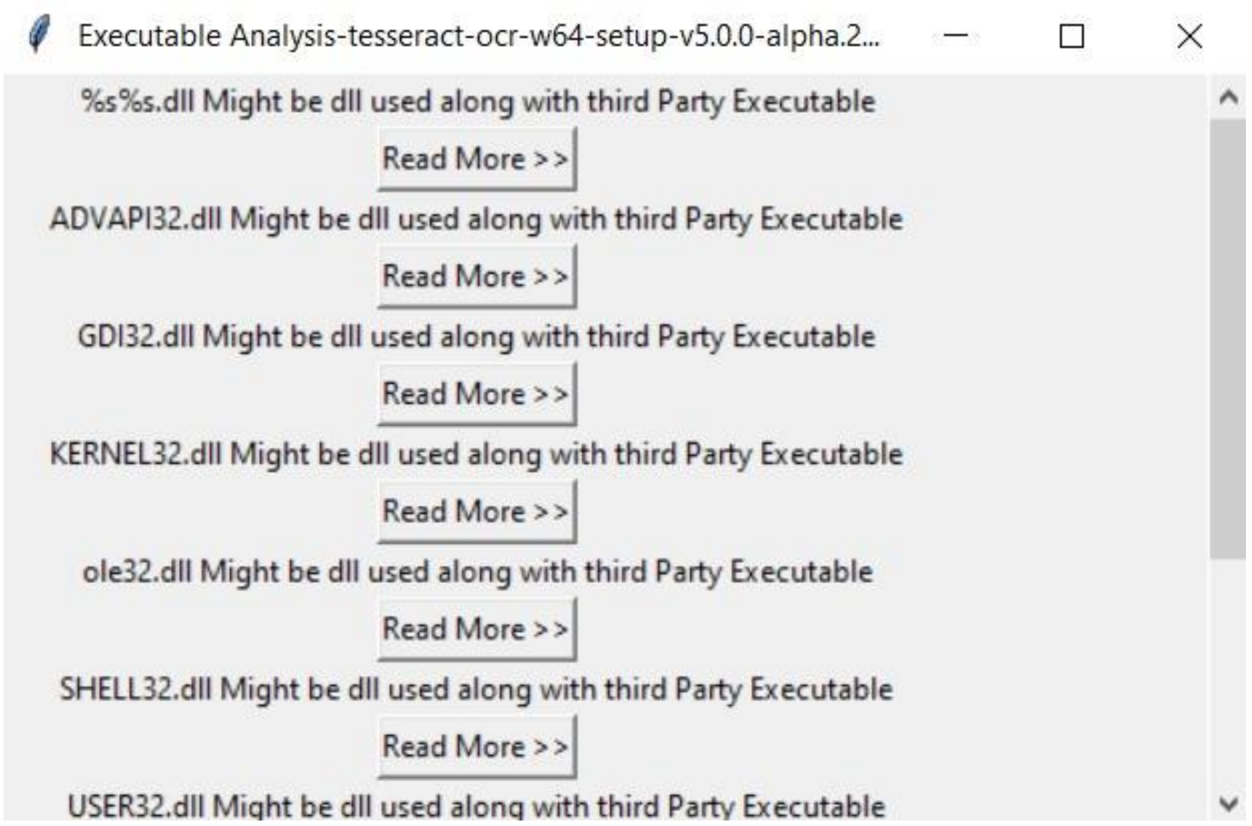
1. Programming language : python
2. LIbraries/Packages :
   2.1 tkinter
   2.2 BeautifulSoup
   2.3 urllib

3. APIs Used : ip-api

1. First it asks the user to upload the executable.

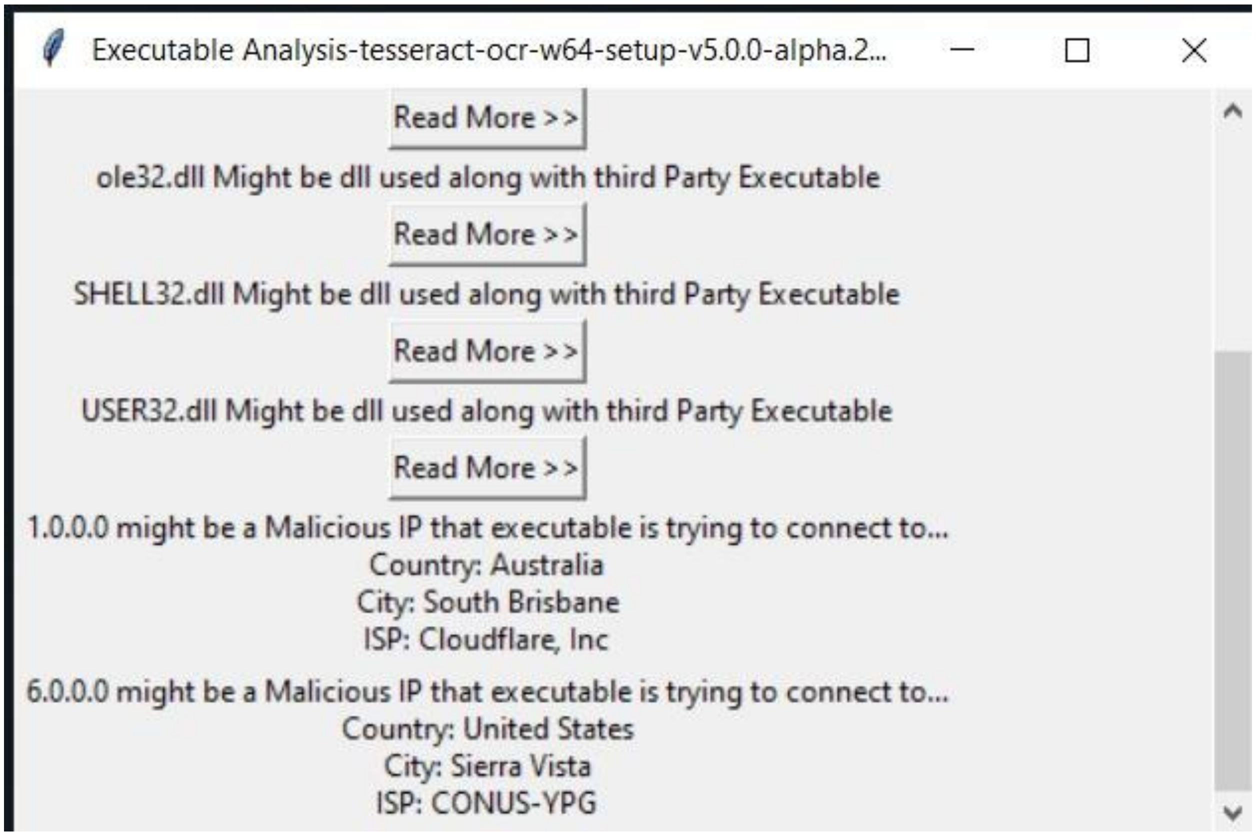


2. Then it extracts all DLL imports and IP addresses and shows them to the user.



Executable Analysis-tesseract-ocr-w64-setup-v5.0.0-alpha.2...

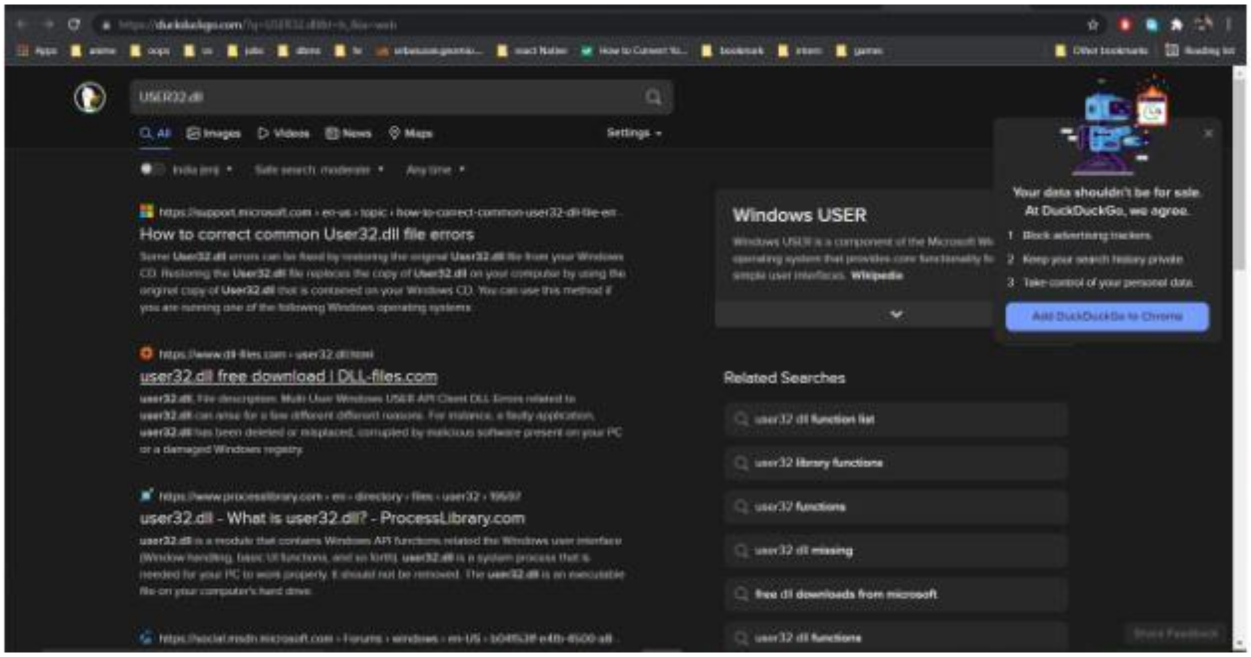
```
%s%s.dll Might be dll used along with third Party Executable
Read More >>
ADVAPI32.dll Might be dll used along with third Party Executable
Read More >>
GDI32.dll Might be dll used along with third Party Executable
Read More >>
KERNEL32.dll Might be dll used along with third Party Executable
Read More >>
ole32.dll Might be dll used along with third Party Executable
Read More >>
SHELL32.dll Might be dll used along with third Party Executable
Read More >>
USER32.dll Might be dll used along with third Party Executable
```



Executable Analysis-tesseract-ocr-w64-setup-v5.0.0-alpha.2...

```
Read More >>
ole32.dll Might be dll used along with third Party Executable
Read More >>
SHELL32.dll Might be dll used along with third Party Executable
Read More >>
USER32.dll Might be dll used along with third Party Executable
Read More >>
1.0.0.0 might be a Malicious IP that executable is trying to connect to...
Country: Australia
City: South Brisbane
ISP: Cloudflare, Inc
6.0.0.0 might be a Malicious IP that executable is trying to connect to...
Country: United States
City: Sierra Vista
ISP: CONUS-YPG
```

3. You can click on 'read more' to read more about the respective dll import.



USER32.dll

All Images Videos News Maps Settings

India (en) Safe search: moderate Any time

https://support.microsoft.com › en-us › topic › how-to-correct-common-user32-dll-file-err...
How to correct common User32.dll file errors
Some User32.dll errors can be fixed by restoring the original User32.dll file from your Windows CD. Restoring the User32.dll file replaces the copy of User32.dll on your computer by using the original copy of User32.dll that is contained on your Windows CD. You can use this method if you are running one of the following Windows operating systems

https://www.dll-files.com › user32.dll.html
user32.dll free download | DLL-files.com
user32.dll, File description: Multi-User Windows USER API Client DLL Errors related to user32.dll can arise for a few different different reasons. For instance, a faulty application, user32.dll has been deleted or misplaced, corrupted by malicious software present on your PC or a damaged Windows registry.

https://www.processlibrary.com › en › directory › files › user32 › 19697
user32.dll - What is user32.dll? - ProcessLibrary.com
user32.dll is a module that contains Windows API functions related the Windows user interface (Window handling, basic UI functions, and so forth). user32.dll is a system process that is needed for your PC to work properly. It should not be removed. The user32.dll is an executable file on your computer's hard drive.

https://social.msdn.microsoft.com › Forums › windows › en-US › b04f53ff-e4fb-4500-a8...

Windows USER
Windows USER is a component of the Microsoft Wi... operating system that provides core functionality fo... simple user interfaces. Wikipedia

Your data shouldn't be for sale. At DuckDuckGo, we agree.
1 Block advertising trackers.
2 Keep your search history private.
3 Take control of your personal data.
Add DuckDuckGo to Chrome

Related Searches
user32 dll function list
user32 library functions
user32 functions
user32 dll missing
free dll downloads from microsoft
user32 dll functions

CONCLUSION

Project was successfully implemented, and problem stated was solved efficiently using Regex. However future scope of work in this project includes: -

• We could include an IP tracker program to trace routes for malicious Ips detected.

• We could create a database of function calls imported.

• Certain executables that are packed can be detected by presence of certain API calls such as

REFERENCES

[1]https://techtalk.pcmatic.com/2017/11/30/running-dll-files-malware-analysis/

[2]https://wikidll.com/microsoft?page=1

[3]https://medium.com/mrx-007/basic-static-analysis-of-malware-and-common-dll-ef9455d49968

[4] https://docs.python.org/3/library/tkinter.html