Hook Analyser Project

Project Home – www.hookanalyser.com

Author – Beenu Arora (www.beenuarora.com)

Email – beenudel1986@gmail.com

Description: Hook Analyser is a freeware project, started in 2011, to analyse an application during the run-time. The project can be potentially useful in analysing malwares (static and run time), and for performing application crash analysis.

The following sections break down the features (and functionality) of the Hook Analyser, and attempts to answer 'How-to' and 'so-what' queries.

Application UI – Significant updates have been performed on the latest release (v 2.2) to make it more verbose.

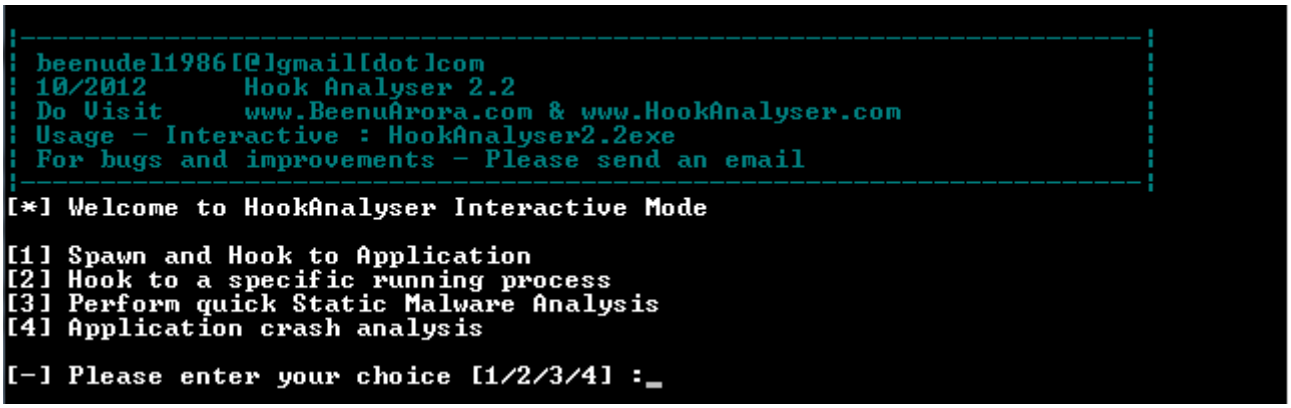

Figure 1

1. Spawn and Hook to Application

This feature allows analyst to spawn an application, and hook into it. Module flow is as following–

a. *PE validation*

b. *Quick static malware analysis.*

c. *Other options (such as pattern search or dump all)*

d. *Type of hooking (Automatic, Smart or manual)*

e. *Spawn and hook*

Currently, there are three types of hooking being supported –

- *Automatic – The tool will parse the application import tables, and based upon that will hook into specified APIs.*
- *Manual – On this, the tool will ask end-user for each API, if it needs to be hooked.*
- *Smart – This is essentially a subset of automatic hooking however, excludes uninteresting APIs.*

Once an application is specified, the tool will perform a quick static analysis to identify any anomalies or malware traces. Refer to the ***section 4*** for detailed information.

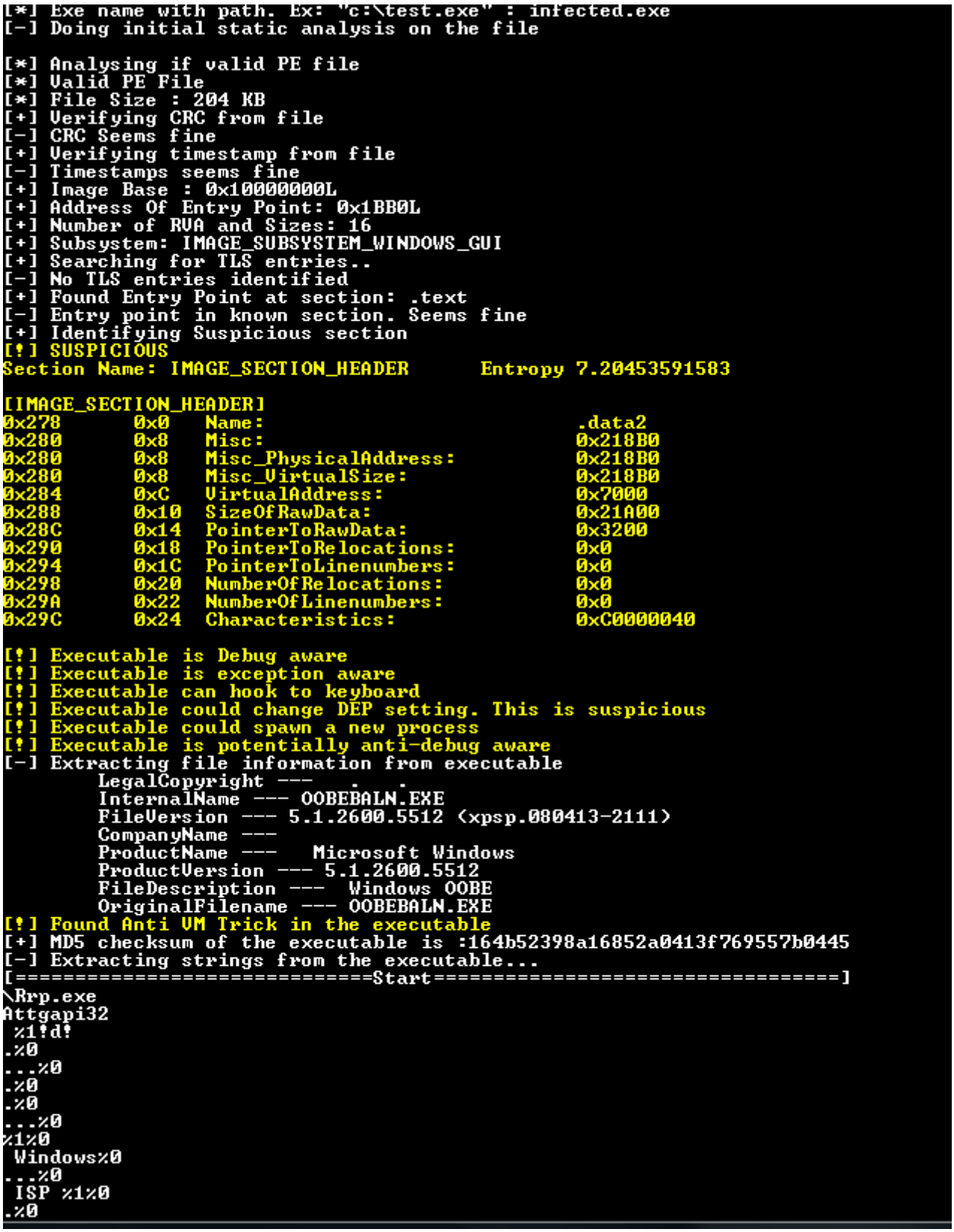
[*] Exe name with path. Ex: "c:\test.exe" : infected.exe
[-] Doing initial static analysis on the file
[*] Analysing if valid PE file
[*] Valid PE File
[*] File Size : 204 KB
[+] Verifying CRC from file
[-] CRC Seems fine
[+] Verifying timestamp from file
[-] Timestamps seems fine
[+] Image Base : 0x10000000L
[+] Address Of Entry Point: 0x1BB0L
[+] Number of RVA and Sizes: 16
[+] Subsystem: IMAGE_SUBSYSTEM_WINDOWS_GUI
[+] Searching for TLS entries..
[-] No TLS entries identified
[+] Found Entry Point at section: .text
[-] Entry point in known section. Seems fine
[+] Identifying Suspicious section
[!] SUSPICIOUS
Section Name: IMAGE_SECTION_HEADER Entropy 7.20453591583
[IMAGE_SECTION_HEADER]
0x278 0x0 Name: .data2
0x280 0x8 Misc: 0x218B0
0x280 0x8 Misc_PhysicalAddress: 0x218B0
0x280 0x8 Misc_VirtualSize: 0x218B0
0x284 0xC VirtualAddress: 0x7000
0x288 0x10 SizeOfRawData: 0x21A00
0x28C 0x14 PointerToRawData: 0x3200
0x290 0x18 PointerToRelocations: 0x0
0x294 0x1C PointerToLinenumbers: 0x0
0x298 0x20 NumberOfRelocations: 0x0
0x29A 0x22 NumberOfLinenumbers: 0x0
0x29C 0x24 Characteristics: 0xC0000040
[!] Executable is Debug aware
[!] Executable is exception aware
[!] Executable can hook to keyboard
[!] Executable could change DEP setting. This is suspicious
[!] Executable could spawn a new process
[!] Executable is potentially anti-debug aware
[-] Extracting file information from executable
LegalCopyright --- .
InternalName --- OOBEBALN.EXE
FileVersion --- 5.1.2600.5512 (xpsp.080413-2111)
CompanyName ---
ProductName --- Microsoft Windows
ProductVersion --- 5.1.2600.5512
FileDescription --- Windows OOBE
OriginalFilename --- OOBEBALN.EXE
[!] Found Anti VM Trick in the executable
[+] MD5 checksum of the executable is :164b52398a16852a0413f769557b0445
[-] Extracting strings from the executable...
[=========Start=========]
\Rrp.exe
Attgapi32
%1!d!
.%0
...%0
.%0
.%0
...%0
%1%0
Windows%0
...%0
ISP %1%0
.%0

Figure 2

Figure 3

Once static analysis is performed, the tool allows analyst to perform one of the following –

- *List APIs – The option is used to list API calls only, and this doesn't display buffer data (on stack or heap).*
- *Pattern search – The option allow searching for patterns or 'strings' in memory after application is spawned and hooked.*
- *Dump all – The option will list API calls and output data from respective buffer (stack or heap).*

Once the above option is selected, analyst can chose type of hooking and proceed (as described above)

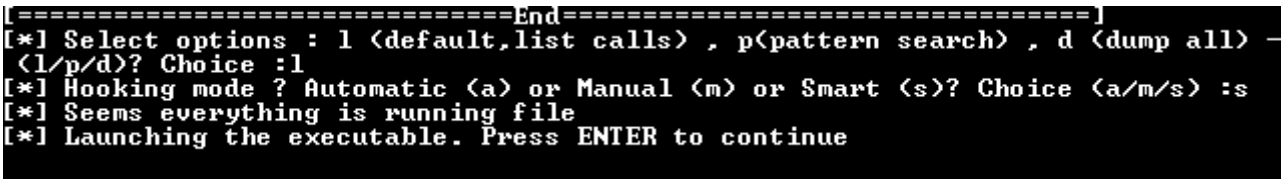

Figure 4

Sample malware analysis demonstration –***http://www.youtube.com/watch?v=sdnRP90weT4***

Note – This video was created using Hook Analyser 1.4 (the old version).

2. Hook to a specific running process

The option allows analyst to hook to a running (active) process. The program flow is –

a. List all running process

b. Identify the running process executable path.

c. Perform quick static malware analysis on executable (fetched from process executable path)

d. Other options (such as pattern search or dump all)

e. Type of hooking (Automatic, Smart or manual)

f. Hook and continue the process

Figure 5

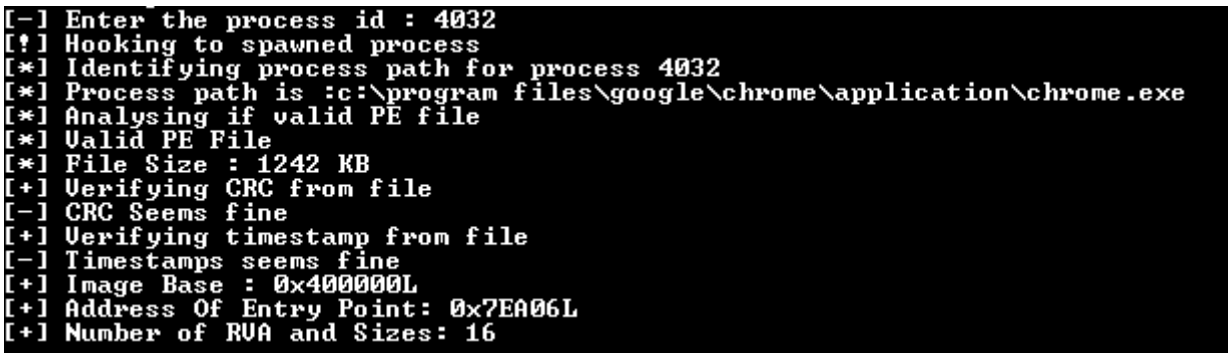

Figure 6

Refer to the ***section 1*** for the rest of the module flow description.

3. Perform quick static malware analysis

This module is one of the most interesting and useful module of Hook Analyser, which performs scanning on PE or Widows executables to identify potential malware traces.

This module has inherited lot of feature from malware analyser project (***www.malwareanalyser.com***).

Currently this module perform the following –

a. PE file validation

b. CRC and timestamps validation

c. PE properties such as Image Base, Entry point, sections, subsystem

d. TLS entry detection.

e. Entry point verification (if falls in suspicious section)

f. Suspicious entry point detection

g. Signature trace (extended from malware analyser project), such as Anti VM aware, debug aware, keyboard hook aware etc. This particular function searches for more than 20 unique malware behaviours (using 100's of signature).

h. Online search of MD5 (of executable) on Threat Expert.

i. String dump (ASCII)

j. Executable file information

k. ...and more.

Refer to the ***figure 2*** for sample screenshot.

4. Application crash analysis

This module enables exploit researcher and/or application developer to analyse memory content when an application crashes.

This module essentially displays data in different memory register (such as EIP).

Application crash analysis video demonstration – ***http://www.youtube.com/watch?v=msYo7pPsu6A***