Malware Analysis – 2

Advanced Static Analysis:

- Basic static and Basic Dynamic malware analysis methods are good for initial triage, but they do not provide enough information to analyze malware completely.
- Advanced static analysis consists of reverse-engineering the malware's internals by loading the executable into a disassembler.
- By looking at the program instructions we can discover what the program does.
- It requires specialized knowledge of disassembly, code constructs, and Windows operating system concepts.
- When we disassemble malware, we take the malware binary as input and generate assembly language code as output, usually with a disassembler (most popular disassembler, IDA Pro)

Tools and Techniques:

IDA Pro:

- The Interactive Disassembler Professional (IDA Pro) is an extremely powerful disassembler distributed by Hex-Rays.
- It supports analysis of various file formats, including the PE/ELF/Macho-O formats.
- IDA is distributed in two other versions: IDA demo version (evaluation version) and IDA Freeware version; both these versions have certain limitations.
- IDA Pro will disassemble an entire program and perform tasks such as function discovery, stack analysis, local variable identification, and much more.
- IDA Pro is meant to be interactive, and all aspects of its disassembly process can be modified, manipulated, rearranged, or redefined.

Disassembling Windows API:

- Malware normally uses Windows API functions (Application Programming Interface) to interact with the operating system (for performing filesystem, process, memory, and network operations).
- Executables import and call these API functions from various DLLs, which provide different functionalities.
- To call the API, the executable process loads the DLL into its memory and then calls the API function.
- Inspecting the DLLs that a malware relies upon and the API functions that it imports can give an idea of the functionality and capability of the malware.
- IDA Pro helps to identify which API or DLL files used and also helps to identify whether the disassembled function is a library or an imported function (a function imported from DLLs).

Advanced Dynamic Analysis

- Advanced dynamic analysis uses a debugger to examine the internal state of a running malicious executable.
- It provides another way to extract detailed information from an executable.
- Most useful when you're trying to obtain information that is difficult to gather with the other techniques.

Debugger:

- Debugging is a technique in which malicious code is executed in a controlled manner.
- A debugger is a program that gives you the ability to inspect malicious code at a more granular level.
- It provides full control over the malware's runtime behavior and allows you to execute a *single instruction, multiple instructions, or select functions* (instead of executing the entire program), while studying the malware's every action.
- Most commonly used debuggers: IDA Pro, x64dbg, DnSpy, radare2, WinDbg, OllyDbg, Immunity Debugger, Hopper, and Binary Ninja.
- Debuggers provide information about a program that would be difficult, or impossible, to get from a disassembler.
- Debuggers provide a dynamic view of a program as it runs. For example, debuggers can show the values of memory addresses as they change throughout the execution of a program.
- There are two ways to debug a program: (a) attach the debugger to a running process, and (b) launch a new process.
- It is always advisable to debug by launching new process, so that you can monitor startup and initialization code that executed.

OllyDbg:

- OllyDbg (x86 debugger) provides the ability to analyze malware while it is running.
- It traces registers, recognizes procedures, API calls, switches, tables, constants and strings, as well as locates routines from object files and libraries.
- As soon as you load a program into OllyDbg, you will see four windows filled with information that you will find useful for malware analysis:
  1) Disassembler window 2) Registers window 3) Stack window 4) Memory dump window

-Hiren Sadhwani