

MALWARE ANALYSIS II – INTERMEDIATE MALWARE ANALYSIS

The malware author's evil job is to develop software that can collect and return data, run undetected, frustrate reverse-engineering efforts, and make detection almost impossible. This course builds on the material presented in Malware I and focuses on three topic areas that are key for successful malware reverse engineering: disassembly, debugging, and Windows internals. Students will learn to infer the functionality of a program by analyzing disassembly and by watching how it changes a system as it runs. They will learn how to extract investigative leads from host and network-based indicators associated with a malicious program, and they will learn how to modify the program to aid analysis. More specifically, students will learn how to identify specific coding constructs in disassembly, they will be taught the art of dynamic analysis, and they will be taught about several Windows APIs most often used by malware authors. Each section is filled with in class demonstrations, exercises where the students follow along with the instructor, and labs where the students practice what they have learned on their own. The ins and outs of both IDA Pro and OllyDbg will be demonstrated and practiced by the students throughout the class.

Duration	4 days
Who Should Attend	Information technology staff, information security staff, corporate investigators or others requiring an understanding of how malware works and the steps and processes involved in Malware Analysis.
Prerequisites	Excellent knowledge of computer and operating system fundamentals is required. Some exposure to software development is highly recommended. Attendance in MANDIANT Malware I, while not required, is extremely beneficial.
Students Will Learn	<ul style="list-style-type: none">▪ Static Program Analysis Methodology▪ Dynamic Program Analysis Methodology▪ Windows Internals and APIs▪ Use of IDA Pro▪ Debugging Programs▪ Advanced Use of the OllyDbg debugger
Exercises / Labs	<ul style="list-style-type: none">▪ Recognizing Coding Constructs in Disassembly (multipart)▪ Utilizing dynamic analysis to understand functionality of a backdoor (multipart)▪ Understanding Windows internals (multipart): file API, loader, networking, registry usage, threads▪ Final lab: Complete analysis of a fully functioning, real world based backdoor
Course Materials	<ul style="list-style-type: none">▪ Student manual▪ Class handouts▪ MANDIANT gear▪ Free Tools CD with course tools and scripts
Suggested Next Courses	<ul style="list-style-type: none">▪ Malware Analysis III▪ Incident Response Management▪ Network Traffic Analysis
Contact	1.800.647.7020 education@mandiant.com www.mandiant.com/education.htm

Disassembly

Every computer program is a combination of data and machine code instructions carefully arranged to perform useful tasks (such as stealing credit card numbers, harvesting Word documents, or cracking passwords). In this section we dig deep into static analysis of the code that makes up a program using a disassembler to transfer the machine code to a more manageable representation.

IDA Pro is the choice for reverse-engineering and analyzing Windows executables. This section serves as a crash-course in IDA Pro and disassembly, including the following major topics:

- x86 assembly language
- Reversing basics: branches, loops and switches
- Reversing basics: functions
- Is it data, or is it code?
- Cross references
- Enhancing disassembly during manual analysis
- Imports and Exports
- Searching
- Type Libraries (Delphi, NTDDK, NTAPI, etc)
- A String is a String?: strings in ASCII, Unicode, Pascal, C, and Delphi
- Defining Arrays
- Defining Structures
- Standard Library Functions and FLIRT
- IDC Scripts, IDAPython, IDARub
- IDA Plugins

Throughout this section we practice the art of static malware analysis by examining executables and identifying in disassembly the coding constructs and notions listed above. IDA Pro is introduced and used extensively.

Debuggers

Although system monitoring tools provide a simple method for watching a program's behavior as it runs, there are many times when an analyst will need to observe and monitor the internal workings of a running program rather than just watching the external behavior. Debuggers provide a means to observe and change both the code and data of a program as it runs. This capability can be used for many different tasks such as: bypassing password checks, changing the hostname or IP address that malware connects to, or observing a tricky decoding sequence as it runs. Specific topics discussed in this section will include:

- x86 Hardware Debugging Support
- Debugging with OllyDbg
- Debugger Basics
- Viewing/modifying memory, disassembly, registers, stack, call tree
- Labeling, commenting, bookmarks
- Breakpoints, conditional breakpoints, hardware breakpoints
- Controlling execution by stepping
- Running traces, back tracing
- Finding and modifying data of interest
- Patching binaries for temporary or permanent behavior modification
- Debugging a dll

Throughout this section we practice the art of dynamic malware analysis by examining executables via OllyDbg.

Windows Internals

To effectively analyze a program, the analyst must have a very strong understanding of the environment in which the program runs. This course focuses on the analysis of malware on the Microsoft Windows platforms and therefore begins with an in-depth discussion of the Windows “environment”. Topics discussed will include:

- PE file format
- Loader and dynamically linked libraries
- Windows API Overview
- Windows types
- Windows file system internals
- Registry usage
- Processes and threads
- Windows networking functions
- Windows Native API

Throughout this section concepts and practices specific to Windows are discussed. Multiple labs will reinforce and demonstrate how these concepts are used in malware.

Final Lab

Prepare to be challenged. We have taken malware collected by MANDIANT via their Incident Response team, crafted a new backdoor based on what we observed, and are turning the resultant executable over to the students for complete analysis. This lab ties in many of the concepts learned over the previous three days and puts them all into a fully functioning piece of malware that is very typical of what we are observing being used to exploit systems again and again. The lab has structured components, but the students are given free reign to uncover the mysteries of the malware.