STATIC MALWARE ANALYSIS
BEGINNING BASICS

ISSA – March 14, 2017
PURPOSE AND OBJECTIVE

• The purpose of Malware analysis is to provide information needed to respond to a network intrusion. Your objective is to find out what happened and ensure that all of the machines and files affected have been located.
TYPES OF ANALYSIS

• Static – Looking at the malware without actually running it
• Dynamic – Running the malware and analyzing its behavior etc.
FURTHER BREAKDOWN OF TYPES

• Basic Static Analysis – Analysis without actually reviewing the program code. Can provide insight on whether a program is, in fact, malware. Can give clues of what behavior to expect. It is, however largely ineffective for more sophisticated malware variants.

• Basic dynamic Analysis - Running the malware, noting it’s behavior. Requires an environment to run it in. This can be done without extensive programming knowledge but like basic static it can miss key details.

• Advanced Static Analysis – Reverse engineering code, use a disassembler, review instructions for purpose …

• Advanced dynamic analysis – Use of a debugger, monitor internal state, variables …
TYPES OF MALWARE

- Malware can span multiple categories - Types are for reference
- Backdoor
- Botnet
- Downloader
- Information Stealing
- Launcher
- Rootkit
- Scareware
- Spam-sending malware
- Worm or Virus
BASIC STATIC ANALYSIS:

• A helpful first step is to test against multiple Anti-virus programs. Differences in AV programs can reveal different signatures and heuristic behavior.

• www.virustotal.com

• Allows you to upload the file that you can scan against numerous antivirus engines and provides a report
HASHING

• Hashing is another good way to identify malware
• Fciv from Microsoft
• md5deep
• http://md5deep.sourceforge.net/
• WinMD5 – GUI based good to view multiple files at once
• http://www.winmd5.com/
HASHING AS A LABEL

• Share with analysts
• See if the file has been identified
THE STRINGS TOOL

• A good set of tools to have is the Sysinternals tool set from Microsoft

• String types:
  • Ascii Strings
  • Unicode strings
  • Microsoft’s Unicode referred to as Wide characters

• Look for strings like registry keys, IP addresses, file names, system messages.
PACKED AND OBFUSCATED MALWARE:

• Basically using techniques to hide the main program and routines:

• PEiD: Doc and links for download at: https://www.aldeid.com/wiki/PEiD

• Very common: UPX packer can be downloaded from http://upx.sourceforge.net

• Some of the PEiD plugins will actually run the malware so be careful.
PORTABLE EXECUTION FILE FORMAT - PE

• The Portable Executable (PE) file format is used by Windows executables and dll’s
• Data structure that contains information needed by the loader
• Nearly every Windows executable uses the PE format
• Yes, there are some legacy Non-PE formats that still show up in Malware
PE FILE HEADERS

• Information about the code
• The type of application
• Library functions
• Space requirements
• All important to malware analyst
LINKED LIBRARIES AND FUNCTIONS

• Imports – Functions used in one program that are stored in different programs, such as code libraries
• Connected to the main executable by linking
TYPES OF LINKING

• Static linking – Entire code library is copied to the executable. Increases program size, Makes analysis harder. Not as common in Windows though used often in *nix

• Runtime linking – Popular for malware (especially packed), Connects to the program only when needed not at compile time or at program start

• Dynamic linking - Occurs at program start. The OS searches as the program is loaded.
OTHER WAYS TO LOAD MODULES

- GetProcAddress
- LdrGetProcAddress
- LdrLoadDll
- LoadLibrary
- Very difficult to detect
DEPENDENCY WALKER

• Dependency Walker is helpful to review Dynamic links
• http://www.dependencywalker.com/
COMMON DLL’S

• Kernel32.dll – Core functionality for manipulation of memory, files and hardware

• Advapi32.dll – Advanced Windows core processes such as Service Manager and the registry

• User32.dll – Contains user-interface components such as buttons, scroll bars ... Indicates a GUI

• Gdi32.dll – Display and manipulate graphics

• Ntdll.dll – Interface to the windows kernel. Generally not imported though indirectly imported by Kernel32.dll. If directly imported it’s likely the author is using functionality is not common in Windows programs such as manipulating processes and hiding functionality.
COMMON DLL’S - CONTINUED

• WSocket32.dll, Ws2_32.dll - For networking
• Winninet.dll – High level networking functions that implement protocols such as FTP, Http, Ntp
• Shell32.dll, shlwapi.dll – Responsible for handling shell API calls, which affect a large amount of the items you interact with in Windows (for example opening files)
PE FILE HEADERS

• Header will give info about imported functions
• Remember a program can export functions for other executables
• Each imported dll will have a list of functions.
• Packed programs won’t yield a lot of information for basic static analysis
• Analyze the sections and headers of a PE file
• Gives the analyst more insight as to what the malware is doing
PE FILE HEADERS - IMPORTANT SECTIONS

- .text – Contains the executable instructions.
- .rdata – Contains the import and export information similar to dependencywalker
- .data – the programs global data
- .rsr – icons, images, menus, strings
- .idata-for import functions
- .edata-for export functions
- .pdata-64 bit only stores exception handling info
SOURCES OF MALWARE

• **Contagio Malware Dump**: Free; password required
• **Das Malwerk**: Free
• **FreeTrojanBotnet**: Free; registration required
• **KernelMode.info**: Free; registration required
• **MalShare**: Free; registration required
• **Malware.lu’s AVCaesar**: Free; registration required
• **MalwareBlacklist**: Free; registration required
SOURCES OF MALWARE

- **Malware DB**: Free
- **Malwr**: Free; registration required
- **Open Malware**: Free
- **theZoo** aka Malware DB: Free
- **Virusign**: Free
- **VirusShare**: Free
ZEUS - GAMEOVER

• **Gameover ZeuS** is a peer-to-peer botnet based on components from the earlier ZeuS trojan. It is believed to have been spread through use of the Cutwail botnet.\(^\text{[1]}\)

• Unlike its predecessor the ZeuS trojan, Gameover ZeuS uses an encrypted peer-to-peer communication system to communicate between its nodes and its command and control servers, greatly reducing its vulnerability to law enforcement operations.\(^\text{[1]}\) The algorithm used appears to be modeled on the Kademlia P2P protocol.\(^\text{[2]}\)

• According to a report by Symantec, Gameover Zeus has largely been used for banking fraud and distribution of the CryptoLocker ransomware.