INTRODUCTION TO MALWARE & MALWARE ANALYSIS

by Quick Heal R&D lab

Quick Heal
Security Simplified

INTRODUCTION

Very often people call everything that corrupts their system a virus without being aware about what it actually means or accomplishes. This paper systematically gives an introduction to the different varieties of samples that come under the wide umbrella known as malware, their distinguishing features, prerequisites for malware analysis and an overview of the malware analysis process.

WHAT IS MALWARE?

The genesis of computer viruses started in early 1980 when some researchers came up with self-replicating computer programs. In 1984, Dr. Cohen provided a definition for computer viruses. Here is Cohen's informal definition of a computer virus:

"A virus is a program that is able to infect other programs by modifying them to include a possibly evolved copy of itself."

This definition, based on the behavior of programs of that period, was appropriate. However, over time viruses have evolved into dozens of different categories and are now termed collectively as malware instead of just 'virus'. A virus is now simply considered as one category of malware.

Malware is short for MALicious softWARE. It is software that is specifically designed to harm computer data in some way or the other. Malware has evolved with technology and has taken full advantage of new technological developments.

Wikipedia[1]:
Malware consists of programming (code, scripts, active content, and other software) designed to disrupt or deny operations, gather information that leads to loss of privacy or exploitation, gain unauthorized access to system resources, and other abusive behavior. The expression is a general term used by computer professionals to refer to a variety of forms of hostile, intrusive, or annoying software or program codes.

SYMPTOMS OF INFECTED SYSTEMS

How do you know that your system is possibly infected with malware? Following are some of the symptoms of an infected system:

- System might become unstable and respond slowly as malware might be utilizing system resources
- Unknown new executables found on the system
- Unexpected network traffic to sites that you don't expect to connect with
- Altered system settings like browser homepage without your consent
- Random pop-ups are shown as advertisements
- Recent additions to the set are alerts shown by fake security applications which you never installed. Messages such as "Your computer is infected!" are displayed and it asks the user to register the program to remove detected threats.

Overall, your system will showcase unexpected and unpredictable behavior.

MALWARE CLASSIFICATION

The category of malware is based upon different parameters such as how it affects the system, functionality or the intent of the program, spreading mechanism, and whether the program asks for user's permission or consent before performing certain operations.

A program can be classified as malware if it does one of the following activities:

- Modifies another program
- Replicates itself through a network or a file system without user's consent
- Allows an unauthorized person to take control over a remote system
- Sends personal or confidential information to a remote system without user consent
- Sends data to a system in order to disrupt normal functioning
- Opens a port for listening in on a local machine to accept commands from a control server
- Records keystrokes and sends this information to remote servers
- Connects to suspicious remote servers
- Downloads and executes files from suspicious remote servers
- Copies itself to multiple locations
- Injects code into another program
- Makes unauthorized changes to the system
- Modifies a protected system setting
- Modifies a registry setting used for launching programs upon startup

Now we will look into specific malware categories based on distinguishing malicious features of the sample.

Virus
Virus is the first category of malware to appear on the horizon of computer security. It is self-replicating in nature and is referred to as a parasitic infector. It does not have a separate existence; instead, it inserts its code into existing files on the system. It could be an executable program or script of different programming languages like VBScript, JavaScript, Perl, etc.

Worm
Worms are also self-replicating; however, they are standalone malware strains. They do not modify other files to spread; instead, they make copies of themselves over network shares or on other systems. Worms are further classified based upon the spreading mechanism used such as email, P2P, IRC, etc.

Trojan
A Trojan is always disguised as useful software and tempts a user to install it and it is also bundled with hidden malicious functionality. It is non-replicating in nature, i.e. it does not spread in a similar manner as viruses or worms.

Backdoor
Backdoors allow unauthorized access to a compromised system by opening a port on the system. This creates a pathway for hackers to control the compromised system by sending commands of malicious nature. SubSeven, NetBus and Back Orifice are some of the well-known examples of backdoors which enable unauthorized people to access user systems over the Internet without his/her knowledge.

HackTool
A HackTool is used by a hacker to attack and exploit a system to gain unauthorized access to system resources. It attempts to gain information about the system after bypassing security mechanisms that are inherent to the system. Netcat is an example of HackTool. Sometimes it is even used by network administrators; however, it is mostly used by hackers to gain unauthorized access and to transmit data on a network.

Spyware
Spyware is software that gathers personal or confidential information from user systems without their knowledge. It includes monitoring the systems to collect information such as browsing habits, recently visited sites, passwords, credit card information, and other confidential information. Once spyware is installed, it does not show any visible notifications to indicate that it is monitoring user activities. It instantly sends this information to the configured remote server.

Rootkit
Rootkits use stealth techniques to actively hide their presence by concealing their components such as files, registry keys, running processes and other objects. These techniques are used to hide their behavior from users and to bypass detection from security applications.

Rogue application
These are fake applications which pose as security applications or system tools to mislead users into paying for the removal of non-existent malware or issues with their systems. This category of malware is on a steady rise over the last 4-5 years. They use different social engineering techniques to mislead users into installing them. Their downloader components may come as video codecs to run certain video clips, P2P software or Trojanized shared applications. Malware writers also use SEO poisoning techniques to push malicious URLs based on recent or popular news. When a user visits such malicious URLs, rogue applications get downloaded using drive-by download techniques by exploiting vulnerabilities in web browsers and their plugins.

INFECTION VECTORS

An infection vector refers to the spreading mechanism used by malware.

- Boot Sector: Infecting Master boot record (MBR) of the physical disk
- File Infection: Parasitic infectors
- Email: Email worms
- File Shares: Parasitic infectors, worms
- Network: Network worms, through vulnerabilities
- IRC: Internet Relay Chat
- P2P Networks: IM, Kazaa, etc.
- Removable Media: Floppy, USB drives, optical discs
- Bluetooth: Worms for mobile devices
- Web Apps: Using cross-site scripting vulnerabilities
- Vulnerabilities: Operating system, Web browser and plugins, Adobe Reader vulnerabilities

PREREQUISITES FOR MALWARE ANALYSIS

Now after being equipped with the knowledge about what malware is, you may want to look at it more closely. The natural question that might come to your mind is - "How to analyze malware?" Prerequisites for Malware Analysis include understanding malware classification, essential x86 assembly language concepts[2], file formats like portable executable file format, Windows APIs, expertise in using monitoring tools, disassemblers and debuggers. This section will introduce to you the prerequisites for malware analysis.

CHEAT SHEET OF X86 ASSEMBLY LANGUAGE

Registers are special data locations within the CPU and they are used for data manipulation.	
EAX	Accumulator, Contains the return value
ECX	ECX used as a loop counter, "this" pointer in C++
EBX	General Purpose Register
EDX	General Purpose Register
ESI	Source Index Pointer
EDI	Destination Index Pointer
ESP	Stack Pointer
EBP	Stack Base Pointer
EIP	Instruction Pointer
Flags	ZERO, SIGN, CARRY, OVERFLOW, TRAP

Assembly Instructions	
ARITHMETIC	ADD, ADC, SUB, SBB, MUL, DIV, IMUL, IDIV, INC, DEC, CMP, NEG
LOGICAL	XOR, OR, AND, TEST, NOT
SHIFT AND ROTATE	ROR, ROL, RCR, RCL, SHR, SHL, SAR, SAL
DATA TRANSFER	MOV, PUSH, POP, PUSHA, POPA, XCHG
CONTROL TRANSFER	CALL, RET, JMP, LOOP, JE/JZ, JL, JG, JGE, JLE, JNE/JNZ, conditional JMPs, INT
STRING	CMPS, SCAS, LODS, STOS, MOVS, REP prefix
MISCELLANEOUS	LEA, NOP, XLAT

PORTABLE EXECUTABLE FILE FORMAT

Microsoft uses the Portable Executable (PE) file format[3] for executables and system libraries ever since Windows 95. For reverse engineering, one should be familiar with the Portable Executable file format.

The PE Header contains important information about linker version used, how the executable should be loaded, compatible version of Microsoft Windows, type of executable file, etc.

Some important fields from PE Header are AddressofEntryPoint and Image Base which point to the address of the first instruction to be executed when the executable is loaded and Virtual Address where executable is loaded in virtual memory, respectively.

The PE header is followed by Data Directories including the import table and export table. The import table has information about functions that the program calls from DLL files. The export table, generally present in DLL files, has information of functions that call other programs. It is followed by the Section Table which provides relative virtual addresses and characteristics of sections of the program.

WINDOWS APIS

Microsoft Windows operating system provides an interface to applications through the Windows Application Programing Interface (API). It is implemented as a set of system libraries such as kernel32.dll, user32.dll, etc. A reverse engineer needs to be familiar with file system, memory management, process and thread management, registry management, networking and security related APIs. Understanding of APIs helps during detailed malware analysis. MSDN[4] provides comprehensive documentation of Windows APIs.

MALWARE ANALYSIS

Microsoft Windows operating system is the most popular and widely operating systems thus making it first in the target list of malware authors. Malware appears in different varieties such as executable files, BAT scripts, VBScript, JavaScript, Macros in Microsoft Office files, exploit code in JPG, GIF, SWF, PDF files. More than 80% malware samples received by security vendors are Windows executables.

The purpose of malware analysis is to study a program's behavior and verify if it has malicious functionality or behavior. If the analyzed sample is found to be malicious, then its classification and identification of which malware family it belongs to is necessary.

ENVIRONMENT FOR MALWARE ANALYSIS

One should be very careful when analyzing malware samples. Malware analysis should be done on a system that is separated from production environment and on a network which is isolated from public network. Virtualization software[9] such as VMWare, Virtual Box provides options to create such an environment.

STATIC ANALYSIS

With static analysis, we study a program without actually executing it. Tools of the trade are Hex editors, disassemblers and packer identifiers. We could look for suspicious strings related to file paths, registry keys, URLs or messages intended for users, if any are used in a program. APIs used also give an idea about the functionality of the program.

Samples which are packed or obfuscated provide stern challenges for static analysis. If a sample is packed, then it needs to be unpacked before diving into code analysis.

DYNAMIC ANALYSIS

With dynamic analysis, we study a program as it executes. We need to monitor the changes made to file system, registry, processes and its network communication. SysInternals tools[6] such as Process Monitor, Process Explorer, TCPView, GMER[7] and Wireshark[8] are useful for observing runtime behavior of a program. Debuggers like OllyDbg, IDA Pro and WinDbg are helpful to dig into details about encrypting malware and for detailed analysis.

In case of non-availability of a safe environment to execute suspicious samples, one could use online automated malware analysis systems[5]. Users could submit suspicious samples for analysis and could then see a generated report based on file system modifications, registry modifications, network communications, etc.

REFERENCES

1. **WIKIPEDIA**
   http://en.wikipedia.org/wiki/Malware

2. **Art of Assembly**
   http://www.arl.wustl.edu/~lockwood/class/cs306/books/artofasm/toc.html

3. **PE File Format**
   http://www.microsoft.com/whdc/system/platform/firmware/PECOFF.mspx
   http://www.openrce.org/reference_library/files/reference/PE%20Format.pdf

4. **MSDN**
   http://msdn2.microsoft.com/en-us/library/default.aspx

5. **Online Automated Malware Analysis Systems**
   http://www.threatexpert.com/
   http://www.sunbeltsecurity.com/sandbox

6. **SysInternals Suite**
   http://technet.microsoft.com/en-us/sysinternals/bb842062

7. **GMER**
   http://www.gmer.net/

8. **WIRESHARK**
   http://www.wireshark.org/

9. **Virtualization Software**
   http://www.virtualbox.org/
   http://www.vmware.com/