Examination of suspicious objects by virus analysts

E V Ananin, I S Ananina, A V Nikishova

Volgograd State University, 100, Universitetskij Av., Volgograd, 400062, Russia

E-mail: nikishova.arina@volsu.ru

Abstract. The paper presents data on virus threats urgency. But in order for antiviruses to work properly, all data on new implementations of viruses should be added to its database. For that to be done, all suspicious objects should be investigated. It is a dangerous process and should be done in the virtual system. However, it is not secure for the main system as well. So the diagram of a secure workplace for a virus analyst is proposed. It contains software for its protection. Also all kinds of setting to ensure security of the process of investigating suspicious objects are proposed. The proposed approach allows minimizing risks caused by the virus.

1. Introduction

Nowadays, virus threats are becoming more relevant for various industries. According to statistic in figure 1, the number of malicious programs grows constantly. Thus regardless of whether one is using a home computer or administer a network of a large company, one might be incurred to a virus attack.

Figure 1. Malware detection volume for the past five years.
To protect devices, people often use antivirus, which can significantly reduce the risk of infection of the system with a virus. However, in order to recognize a threat, it is necessary that antivirus contains a signature of the virus, which is made by disassemble of a suspicious object by virus analysts.

2. Approaches to suspicious files’ investigation

Not all suspicious objects can be a threat. The behavior of some may often be similar to a virus, but it can be a completely legitimate program (for example, archivers can be confused with ransomware). Malware detections by antivirus do not always indicate actual infections, but rather the weaponization of the code and/or attempted delivery to target victims and systems. Therefore, there is a need to carefully disassemble the object before it can be confidently asserted that it is dangerous.

To investigate suspicious files, virus analysts can apply several approaches:

- Monitoring.
- Static analysis.
- Dynamic analysis.
- Automatic analysis.

With the help of the first method, it is often possible to determine the suspicious activity of the program without a detailed analysis of the suspicious object. To do this, monitor programs are configured (for example TCPview, Procmon, Procexp, Fiddler) and a suspicious file is launched like in figure 2.

![Figure 2. Example of using monitoring.](image)

“Attention! You should not run a suspicious file on your work-PC! To do this, you must set up a
protected test-system, which will be described later". After the suspicious object is launched, the monitoring programs will show the activity of the object. What processes it creates, what files it writes, what registry keys were created, and what connections were opened. In addition, one can examine what data are sent to remote servers. This method allows one to learn a lot of useful information about the file, but it has a big drawback - low security and safety. Therefore, before one starts the file and collects information about its activity in the system, it must first be examined at least by static analysis. [1]

Static analysis provides an analysis of the file while it is not executed by the system. The following utilities are used for static analysis.

- HIEW - well-known Hex editor, which has a built-in disassembler for x86, x86-64 and ARM architectures.
- IDA Pro - powerful interactive disassembler, which includes a lot of useful functions, greatly facilitating the analysis of the file.
- md5deep - program that allows calculating md5 hash of the file.
- strings - utility, which allows one to view the ASCII strings contained in the file.
- PEID / DiE / Protection_ID - utilities, which help to understand how the suspicious file is packed, what programming language it is written in, look at the sections of the executable file and show the entry point of the examined program (depending on the utility, the set of functions can be different).
- Dependency Walker-utility that allows one to view a list of functions used by the program from imported libraries.
- PEView - utility that allows examining sections of executable files.
- Resource Hacker - utility, which allows viewing the contents of the resource section of the executable file. It often contains elements of the graphical user interface.

Static analysis is safer than monitoring and allows one to gather a huge amount of useful information about the file being researched like in figure 3.

![Figure 3. Example of using static analysis.](image)

However, this approach has a drawback. If the file one is examining is packed, obfuscated or compressed, static analysis will not give much information that can help one to determine if the file is
dangerous. In addition, the program may not have some modules that dynamically loaded from the server to perform various tasks. To detect suspicious activity, to unpack and to analyze carefully suspicious file activity, dynamic analysis is used. [2]

Dynamic analysis allows one to explore the program in the process of its execution by the system. “Attention! Dynamic analysis should be done after a thorough static analysis and only on a prepared test or virtual machine! For dynamic analysis, debuggers are usually used. The most common debuggers are IDA Pro (with debugger), OllyDBG, x64dbg, gdb, radare2 (can be used for static analysis). This approach can be combined with monitor programs. This will allow one to gather more complete information and helps one not to miss the suspicious activity of the investigated file. The main advantages of dynamic analysis of a suspicious file are the ability to step through every program's actions step by step, unpack it, view the contents of the memory at any time, cut off unused portions of the code, allow the program to load the missing modules and get an idea of the protection or cryptographic mechanisms of the program as shown in figure 4.

![Figure 4. Example of using dynamic analysis.](image)

The main drawback of this method is low safety. Uncontrolled execution of malicious code during dynamic analysis can seriously damage one's virtual or test system. [3]

In addition to static and dynamic, the method of automated analysis of suspicious files deserves special attention. To do this, so-called, "sandboxes" are used. They are configured to collect certain activity information about the executable file. The file is safely executed and all the necessary information is gathered from its activity and a report is made. Besides protected sandboxes, there are software packages, for example SysAnalyzer, which run the file for execution and automatically monitors program activity and prepares a report using built-in modules. The main drawback of the second type of automatic analyzers is low security. The file, which is being analyzed, directly runs in the system, and this can lead to serious consequences. Therefore, this type of program should be used on test or virtual systems. [4]

3. Threats of suspicious files’ investigating
By investigating suspicious files, the virus analyst endangers not only his test or virtual system, but also other systems in both the local and the global networks. For example, accidentally running a suspicious file for execution. An inexperienced analyst can accidentally run a dangerous file in a
debugger without setting breakpoints, or simply from the file explorer. In most cases, if such uncontrolled execution occurs on tests or a virtual system is not connected to the network, only the system itself will be affected and the damage will be minimal.

In another case, if there is an Internet connection, the investigated program can start sending spam or join a bot-net and start a DoS attack on a remote host. In addition, the debugger programs themselves are vulnerable to exploitation by the disassembled suspicious file. This can also be used by an intruder against a test or virtual system of a virus analyst.

Also, it is possible that the attacker envisaged a possible virus launch in the virtual system. Unfortunately, virtual systems are also vulnerable. Therefore, an attacker can try to exploit such vulnerability and execute an arbitrary code no longer in a virtual environment, but on a host system (for example, vulnerability CVE-2016-7461).

Currently, there are many methods for determining the execution of a file on a virtual system, the main ones of which are:

- Search for files, processes, registry keys related to virtual machines and their guest additions.
- Analysis of the system hardware. It can be a processor, PCI device list, BIOS, MAC-address, drivers.
- Attempt to execute high priority instructions such as IN, SIDT, SGDT, SLDT.
- Combined methods.

Even when running on a virtual system, virus can cause damage:

- If one examines ransomware, one can have all one’s files encrypted, even VM machine images.
- Licenses/license files could be stolen from the virtual system.
- BIOS can be spoiled.
- Hardware of the virtual system, which is commonly shared with the real system, can be attacked, e.g. overclocking, decreasing fan speed.
- Users’ passwords of the virtual system can be stolen.

To ensure safety of suspicious files’ investigating process, it is proposed to organise a secure workplace for the virus analyst.

4. Secure workplace for virus analyst
Proceeding from the above-mentioned information, it is very important to prepare a separate system, whether it is test, working or virtual, for the work related to the analysis of a suspicious file. Figure 5 shows an exemplary diagram of a secure workplace for a virus analyst.

![Exemplary diagram of secure workplace for virus analyst](image)

**Figure 5.** Exemplary diagram of secure workplace for virus analyst.
On the test machine, those programs for monitoring the activity of the file are installed: TCPview, Procmon, Procexp, Fiddler. In addition, the system should have tools for dynamic analysis of the programs, at least OllyDBG. To protect the system from the consequences of executing a suspicious application, the Shadow Defender utility is used, which allows in an automatic mode to restore the “clean” state of the hard disk every time the system is rebooted. So, all changes that occur in a user session are not written to the hard disk, but are reset to the initial state which set by the analyst. However, it is still recommended to make a backup of the test system in case Shadow Defender does not cope with its task.

In addition, it is necessary to provide limited access for the test system to the Internet, so that the investigated file can load its modules, receive commands from a remote server and access remote addresses, but could not infect nearby hosts. In order to provide adequate protection against virus activity and secure the local network, the test computer connected to firewall, must be configured for safe operation.

A virtualization tool is installed on the working computer (for example, Vmware, VirtualBox), static analysis tools (IDA). The virtual system is installed with dynamic analysis tools; a snapshot of the “clean” system state is taken.

Between all systems, one should configure shared folders for quick file sharing. On the working machine, the shared folder is added to the antivirus exceptions, the Group Policy must be set up not allowing execution of the executable files from the shared folder.

The proposed set of programs and settings allows one to minimize risks caused by the virus.

5. Conclusion
For modern antivirus software to work properly, all suspicious files should be carefully investigated. There are four approaches to analyzing suspicious files. But all of them have drawbacks. To ensure safety of the virus analyst’s system, virtualization is used.

However, investigating suspicious files can be dangerous for main system, even when it is made in the virtual system. In order to minimize risks, a diagram of a secure workplace for a virus analyst is proposed. Utility Shadow Defender and settings of virtual system’s network connection and file system’s permissions allow making more secure investigations of suspicious files.

6. Acknowledgments
The reported study was funded by the Council for grants of the Russian Federation President according to research project No. MK-6404.2018.9.

References
[1] Perdisci R, Lanzi A and Lee W 2008 Classification of packed executables for accurate computer virus detection J. Pat. Rec. Let. 29 1941-1946
[2] Díaz G and Ramón J Bermejo 2013 Static analysis of source code security: Assessment of tools against SAMATE tests J. Inf. and Software Tech. 55 1462-1476
[3] Shijo P and Salim A 2015 Integrated Static and Dynamic Analysis for Malware Detection J. Procedia Comp. Sci. 46 804-811
[4] Noor M, Abbas H and Bin Shahid W 2018 Countering cyber threats for industrial applications: An automated approach for malware evasion detection and analysis J. Jour. of Net. and Comp. App. 103 249-261
[5] Mao W, Cai Z, Yang Y, Shi X and Guan X 2018 From big data to knowledge: A spatio-temporal approach to malware detection J.Computers and SecurityVolume 74 167-183
[6] Stiborek J, Pevný T and Rehák M 2018 Probabilistic analysis of dynamic malware traces J. Computers and SecurityVolume 74 221-239
[7] Al-rimy B.A.S., Maarof M.A. and Shaid S.Z.M. 2018 Ransomware threat success factors, taxonomy, and countermeasures: A survey and research directions J. Computers and SecurityVolume 74 144-166
[8] Singh B and Singh U 2018 Program execution analysis in Windows: A study of data sources, their format and comparison of forensic capability *J. Computers and Security* Volume 74 94-114

[9] Sheneamer A, Roy S and Kalita J 2018 A detection framework for semantic code clones and obfuscated code *J. Expert Systems with Applications* Volume 97 405-420

[10] Manès V.J.M., Jang D., Ryu C. and Kang B.B. 2018 Domain Isolated Kernel: A lightweight sandbox for untrusted kernel extensions *J. Computers and Security* Volume 74 130-143