ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

A Literature Review On Anti Virus And Its Analysis

Author[1]: Mr. Vishal Patel
Assistant Professor
St. Stephen Institute of Business Management & Technology, Anand
E Mail: vpp3188@gmail.com & Mobile: 9033472227

Author[2]: Dr. Pravin H. Bhathawala
Professor
Calorx Teachers University, Ahmedababd
E Mail: pcb1010@yahoo.com & Mobile: 9825148680

ABSTRACT

Anti Virus are nasty software's. It is designed to damage computer systems without the knowledge of the owner using the system and technique advancements are posing big challenges for researchers in both academia and the industry. The purpose of this study is to examine the available literatures on Anti Virus analysis and to determine how research has evolved and advanced in terms of quantity, content and publication outlets. Most Anti Virus programs are large and complex and one can't possibly understand every detail. Educating the internet users about Anti Virus attack, as well as the implementation and proper application of anti-Anti Virus tools, are critical steps in protecting the identities of online consumers against Anti Virus attacks.

Keywords: Anti Virus, Internet, Anti Virus attack, forensic analysis, anti-Anti Virus tools

INTRODUCTION

Anti Virus is a general term that encompasses viruses, Trojans, spywares and other invasive code is widespread today. Anti Virus analysis is a multi-step process providing insight into Anti Virus structure and functionality, facilitating the expansion of remedy.

According to researcher

1) (Christodorescu et al., 2005) described a Anti Virus instance as a program whose objective is malevolent.

2) (McGraw and Morrisett,2000) defined malicious code as "any code added, changed, or removed from a software system in order to intentionally cause harm or subvert the intended function of the system."

3) The description given by (Vasudevan and Yerraballi, 2006) which described Anti Virus as "a generic term that encompasses viruses, trojans, spywares and other intrusive code."

4) (Aycock, 2006) defined Anti Virus as "software whose intent is malicious, or whose effect is malicious".

5) The term "Anti Virus" here is being used as the generic name for the class of code that is malicious, including viruses, trojans, worms, and spyware. Anti Virus authors use generators, incorporate libraries, and borrow code from others—there exists a robust network for exchange, and some Anti Virus authors take time to read and understand prior approaches by (Arief & Besnard ,2003.)

6) (Fred Cohen's) original definition of a computer virus as of 1983 was: "a program that can 'infect' other programs by modifying them to include a possibly evolved copy of itself." He updated this definition a year later in 1984 in his paper entitled: "Computer Viruses – Theories and Experiments".

7) According to BBC News online, 2004 Anti Virus is a general term for a piece of software inserted into an information system to cause harm to that

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

system or other systems, or to subvert them for use other than that intended by their owners.

8) (Skoudis and Zeltser, 2003) Anti Virus is a set of instructions that run on your computer and make your system do something that an attacker wants it to do.

The term computer virus was first used in a science fiction novel by (Gerrold, 1972), which includes a description of a fictional computer program called virus and was able to self-replicate. The first academic use of the term was claimed by (Cohen, 1983). The first published account of the term can be found a year later by (Cohen, 1984) in his paper Experiments with Computer Viruses. Though Cohen first used the term, some early accounts of viruses can be found. According to (Ferbrache, 1992), the first reported incidents of true viruses were in 1981 and 1982 on the Apple II computer. Elk Cloner is considered to be the first documented example of a virus in mid-1981. The first PC virus was a boot sector virus called Brain in 1986, (Hoffman, 1990). Worm also owes their existence to science fiction literature. (Brunner's, 1975) Shockwave Rider introduced us to worm, a program that propagates itself on a computer network. (Shoch, 1982) claimed the first use of the term in academic circles.

Much has been written about viruses, worms, trojans and other Anti Viruss since then, but now we shift our focus, from fiction to the real world where both Anti Virus and anti-Anti Virus are big commercial industries now (Gutmann, 2007).

ANTI VIRUS ANALYSIS

Anti Virus analysis is a multi-step process providing insight into Anti Virus structure and functionality. Behavior monitoring, an important step in the analysis process, is used to observe Anti Virus relations with respect to the system and is achieved by employing dynamic coarse-grained binary-instrumentation on the target system. Initial examination of collected Anti Virus is called profiling, (Aquilina et al., 2008).Dataflow analysis examines the way data is moved and changed throughout the execution of a program (Chess et al., 2007).

(Skoudis, 2004) outlined a model where analysis tools are distributed on a local victim machine and on an external machine, to capture behavioral aspects of the Anti Virus on the local machine and its interaction with external services over a network. External services as outlined by (Arnold et al., 2000) can be setup on the external monitoring segment.

(Rieck et al.) experimented with different heterogeneous test data collected over several months using honeypots demonstrated the effectiveness of the method, especially in detecting novel instances of Anti Virus families previously not recognized by commercial anti-virus software.

A number of analysis tools are utilized by Anti Virus forensic analysts, with static and dynamic analysis representing two significant methodologies that can be used to analyse Anti Virus (Aquilina et al., 2008). Software disassemblers and debuggers such as IDA Pro (Hex-Rays, 2008) and OllyDBg (Yuschusk, 2008) can be used to perform a detailed analysis of the Anti Virus code and provide an internal view of the Anti Viruss functionality (Valli & Brand, 2008). This is referred to as static analysis.

In contrast, dynamic analysis runs the Anti Virus and observes the interaction of the running Anti Virus with the computer from a behavioural point of view. A number of plug-ins that extend the functionality of IDA Pro and OllyDBg include IDA Stealth (Newger, 2008) and Olly Advanced (MaRKuS, 2006) respectively to work with malicious code that employ anti-analysis techniques. The intention of such plug-ins is to provide functionality to hide their associated tools. Extensive literature exists on static analysis of malicious binaries, e.g. (Christodorescu et al., 2005; Kirda et.al, 2006; Kruegel et.al, 2004). Moreover, recent work of (Moser et al., 2007)

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

presents obfuscation techniques that are provably NP-hard for static analysis.

Dynamic Anti Virus

analysis techniques have previously focused on obtaining reliable and accurate information on execution of malicious programs (Bayer et al., 2006; Moser et al., 2007; Willems et al., 2007).

Two techniques for behavior-based Anti Virus analysis using clustering of behavior reports have been recently proposed (Lee et al., 2006; Bailey et al., 2007). (Moser et al., 2007) proposed a system that dynamically monitors a suspicious program to identify the execution points where the application makes control flow decisions based on input-dependent values.

Static Anomaly Detection

(Wagner, 2011) proposed a technique that created a control flow graph (CFG) for a program representing its system call trace. At execution time this CFG was compared with the system call sequences to check for any violation.

Hybrid Anomaly Detection

(Rabek, 2003) proposed an anomaly based technique where static analysis was assisted by dynamic analysis to detect injected, dynamically generated and obfuscated code. Within the program static analysis was used to identify the location of system calls. The programs can be dynamically monitored later to verify that each realistic system call is made from the same location well-known using the static analysis.

Static Misuse Detection

(Bergeron et al., 1999) used a static misuse detection scheme where they used program slicing to extract program regions that are critical from a security point of view.

In a related work (Bergeron et al., 2001) extracted an API call graph instead of the program slices to test against the security policy. (Lo et al., 1995) proposed the idea of tell-tale signs which were heuristic signatures of malicious program.

Dynamic Anomaly Detection

(Hofmeyr et al., 1998**)** proposed anomaly detection based upon sequence of system calls. A normal profile was composed of short sequence of system calls.

In a similar approach (Sekar et al. 2001) used Finite State Automata (FSA) to represent system call sequences. Similarly (Ko et al., 1997) proposed an idea of trace policy which was essentially a sequence of system calls in time (Masri et al,. 2005) presented a tool called Dynamic Information Flow Analysis (DIFA) to monitor method calls at runtime for Java applications (Sekar et al., 1999) created a system call detection engine that compares system calls modeled previously with the system calls made at runtime.

Hybrid Misuse Detection

(Mori, 2004) presented an approach to detect encrypted and polymorphic viruses using static analysis and code emulation.

Dynamic Misuse Detection

(Debbabi, 2001) proposed a dynamic monitoring system that enforces a security policy. The approach was implemented in a system called DaMon. Schneider 1998 presented enforceable security policies in the form of Finite State Automata.

(Vasudevan et al.) have developed a new dynamic coarse-grained binary-instrumentation framework codenamed SPiKE, that aids in the construction of powerful Anti Virus analysis tools to combat Anti Virus that are becoming increasingly hard to analyze. Goal is to present a binary-instrumentation framework that is unremarkable, moveable, capable, easy-to-use and reusable, supporting multithreading and SM-SC code, both in user- and kernel-mode.

(Valli et al.) laid an establishment for a Anti Virus Analysis Body of Knowledge (MABOK) which is required to analyse the Anti Virus forensically. This body of knowledge has been the outcome of several years of study into Anti Virus categorization. Debuggers such as OllyDbg (Yuschuk, 2008) and IDA Pro (Hex-Rays, 2008) are commonly used for the analysis of Anti Virus.

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

Plugins such as Olly Advanced (MaRKuS, 2006) for OllyDbg and IDA Stealth (Newger, 2008) for IDA Pro focus on hiding the presence of the tool from the software under investigation, in an effort to avoid detection. (Christodorescu et al.) presented a unique viewpoint on malicious code detection. Attacker who writes the malicious code tries to conceal the malicious code to threaten the malicious code detectors such as Anti- virus software.

(Bayer et al.) presented TT Analyze, a tool for dynamically analyzing the behavior of Windows executables. Binary is run in an emulated operating system environment and the actions are monitored. It does not modify the program it executes which one of the most important aspects of the system is making it more difficult for the malicious code to be detected. This tool runs the binaries in unchanged Windows environment making it accurate. Therefore these factors make TT Analyze a perfect tool for quickly getting an accepting of the behavior of an unknown Anti Virus. A summarized list of the analysis tools recommended by (Skoudis, 2004) as well as their purpose and analysis type, is shown in Table 1 in the following page.

Table 1- Summary of Anti Virus analysis tools showing analysis type, purpose and name of commonly used tool name.

Analysis Type	Purpose	Tools
Static	Use as many antivirus detection engines as possible to assist classification.	Virus Total (Virus Total, 2008)
	Search the body of the Anti Virus for strings.	Strings (Microsoft, 2008c)
Dynamic	File integrity check to record baseline configuration.	Winalysis (Winalysis.com, 2008)
	File monitoring. Find which tools are opening, reading and writing files.	Filemon (Microsoft, 2008c)
	Process monitoring. Determine resources that are being used such as DLL's and registry keys.	Process explorer (Microsoft, 2008c)
	Network monitoring. Uncover which ports are open, collect network traffic and find vulnerabilities.	Fport (Foundstone, 2008), tcpview (Microsoft, 2008c), nessus (Tenable Network Security, 2008), nmap (Insecure.org, 2008), wireshark (Combs, 2008), and snort (Sourcefire, 2008).
	Registry monitoring. Monitor registry activities as they occur.	Regmon (Microsoft, 2008c)
Code	Disassembly, debugging	IDA Pro OllyDbg (Yuschuk, 2008)

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

ANTI VIRUS ANALYSIS IN VIRTUAL MACHINE

(Garfnkel et al., 2003) were the first to propose to use a VMM to perform OS-aware introspection, and subsequently the idea was further elaborated. Other researchers instead proposed to use a VMM to protect the guest OS from attacks by supervising its execution, both with a software based VMM and by leveraging hardware support for virtualization. Similar ideas were also suggested by other authors. (Chen et al.) proposed

a solution to protect applications' data even in the presence of a compromised operating system. More recently, Vasudevan et al. presented XTREC, a lightweight framework to record securely the execution control flow of all code running in an untrusted system.

(Sharif et al., 2009) introduced a framework that allows in-VM" monitoring and detection. Those who employ virtual machine introspection techniques to isolate security tools from the

unprocessed environment are very efficient, but they are also expensive.

(Kolbitsch et al.) described a technique for efficient and effective Anti Virus detection. Their idea is to build models of the malicious samples offline, and then to verify at run-time if the behavior of a suspicious application adheres to a known model. (Martignoni et al.) used hierarchical behavior specifications to build a model of a malicious program. As the number of malicious samples keeps growing, efficiency is essential not only for detectors, but also for automatic Anti Virus analysis systems. To address this problem, (Bayer et al., 2010) proposed a technique that allows to detect if a binary is a polymorphic variation of a Anti Virus sample that has already been analyzed in the past.

ANTI ONLINE ANALYSIS ENGINE

1) Anubis (International Secure Systems Lab, Vienna University of Technology, Eurecom France, & UC Santa Barbara, 2008) is an online Anti Virus behavioral analysis service.

2) Sandboxie (Sandboxie, 2008) is an application where suspicious programs can be run in an environment that uses a transient storage area, known as a sand box, so that data is not written to the hard drive. This allows the analyst to observe what an unknown program is going to do.

3) Norman Sandbox (Norman, 2008) also provides an online service to analyze Anti Virus, but this also can be detected.(Krack ,2006) notes that the presence of the sandbox can be detected by "reading it's memory, and comparing it to that of a standard computer".

The detection performance of AV software has been shown by a number of researchers to be far less than ideal (Rutkowska, 2006; Yan et al., 2008; Yinnet al., 2007; Zhou et al., 2008).

Online analysis engines are offered that can give very valuable reports such as detected virus signatures, network activities, file activities, service activities, registry activities and process activities. Online analysis engines such as Anubis

(International Secure Systems Lab, Vienna University of Technology, Eurecom France, & UC Santa Barbara, 2008) have limitations (Bayer, 2009), such as the virtual machines in which the dynamic analysis is being conducted being detected (Innes et al., 2006; Smith et al., 2006) A series of articles by (Hudak, 2009a, 2009b) provided an introduction to automating Anti Virus analysis that can be further customized and extended by using additional tools and scripts.

A range of techniques exist to detect that the Anti Virus is running inside a virtual machine such as VMware or Virtual PC (Innes et al., 2006; Smithet al., 2006). The use of these techniques can be detected and mitigated (Eagle, 2004). The general methodology presented by Zeltser (2007) is as follows:

1. Run the Anti Virus in an isolated laboratory.

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

2. Monitor the interaction between the system and network from a behavioural sense.
3. Understand's the program code.
4. Repeat the process until enough information is gathered.

However, Anti Virus can use techniques to determine if it is running in a virtual machine as demonstrated by the logic of the following pseudo code reproduced from a presentation by (Smith et al., 2006) as shown:

IF detect_vmware
THEN do nothing, destroy self, destroy system ELSE
Continue with Anti Virus payload

Eagle, reported that VMware uses a registry key for the installation location of VMware as: HKLM\Software\VMware,Inc.\VMwareTools\Inst allPath

Anti Virus can identify the key to point out that it could be running in a virtual machine. Another technique Eagle points out, is to use the Windows Management Instrumentation (WMI) to iterate though the network interfaces to see if any of the MAC addresses used belongs to VMware. Eagle suggests the following to mitigate this technique:

• Uninstall VMware tools.

• Change the MAC address of the virtual adapter in the guest OS.

(Porras et al., 2007) noted that recent versions of Storm appear to have stopped checking to see if it is running inside a virtual machine and is instead focusing on hiding themselves from monitoring software.

ANALYSIS AVOIDANCE

Malicious software (Anti Virus) has a wide range of analysis avoidance techniques that it can employ to hinder forensic analysis. Although Rolegal software can incorporate the same analysis avoidance techniques to provide a measure of protection against reverse engineering and to protect intellectual property, Anti Virus invariably makes much greater use of such techniques to make detailed analysis labour intensive and very time consuming.

(Brand et al.) suggested that the discovery of the intent of deception may be a very good indicator of an underlying malicious objective of the software under investigation.

A review of the literature on Anti Virus analysis methodologies found that the most effective methodologies take the presence of analysis avoidance techniques into account (Skoudis et al., 2004; Zeltser, 2007). (Zeltser, 2007) presented an incremental, static and dynamic spiral analysis methodology for analysing Anti Virus which additionally moulds the analysis environment as understanding of the Anti Virus is attained.

Software with a malicious intent may be considered to be far more likely to employ anti- analysis techniques than legitimate software (Vuksan et al., 2009), to the extent that, detection of the presence of anti-analysis techniques may indicate the presence of Anti Virus (Wysopal, 2009).

OUTCOMES AFTER AN INVESTIGATION INTO THE ANALYSIS AVOIDANCE TECHNIQUES OF MALICIOUS SOFTWARE

To deter digital forensic examination a number of lessons learned from the techniques employed for the malicious executable software during investigation. Detection of Anti Virus signature has been recognized by researchers to be far less than ideal. Thus to manually analyse the suspicious files there's a requirement of forensic analyst. To avoid detection & hide its true intent, the attacker make use of packers, protectors or cryptors to obstruct the forensic analyst. Therefore the analyst must understand the limitations of tools, anti-analysis techniques & how to employ proper analysis methodology to uncover the aim of the Anti Virus

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

(Brand et al).

Modern Anti Virus incorporates stealth techniques to hide it from the analyst, deception techniques to hide its true intent, and active techniques to defeat common analysis tools in their default configurations (Grugq, 2003; Harbour, 2007; Rutkowska, 2006a, 2006b). Such techniques are commonly referred to as anti-forensics and are becoming a very important consideration for the digital forensic analyst, as the majority of modern Anti Virus employs these analysis avoidance techniques (Falliere, 2007; Ferrie, 2008; Yason, 2007). For the analysis of malicious network honeypots is used which is a rising forensic tool. Generally research lab and security firm use honeypots to capture new variant of Anti Virus. (Kumar et al.) used honeypots for generating and propagating direct cures for unknown and new Anti Virus in a network in the form of on-the-fly antiAnti Virus signature which spread in a way similar to the spread of Anti Virus in network. The remarkable gain of implementing this technique is that for new Anti Virus which has not been discovered by researcher and security firm the above proposed system would be capable of providing an effective cures.

Table 2- A proposed taxonomy of techniques employed by Anti Virus in order to avoid analysis technique is as follows:

Anti Emulation	A range of techniques exist to detect that the Anti Virus is running inside popular VM‘s such as VMware or Virtual PC.
Anti Online Analysis	A variety of techniques exist for Anti Virus to determine if it is running in a specific online analysis engine such as Anubis or Norman Sandbox.
Anti Hardware	Techniques that target hardware such as the CPU including the debug registers to determine if it is being debugged.
Anti Debugger	Target the way debuggers work and take advantage of these to take control of the flow of execution. This gives Anti Virus the opportunity to incorporate deception.
Anti Disassemblers	Target the way disassemblers work and take advantage of this to produce a false disassembly.
Anti Tools	Detect the presence of specific analysis tools and enter a deceptive mode.
Anti Memory	Target the way memory is used when a process is being debugged and take advantage of this as well as the way processes can be dumped from memory including the use of stolen bytes.
Anti process	Target the way processes are handled when being debugged and take advantage of this including structured exception handling.
Anti Analysis	Target the way analysis is conducted. Use junk code, code camouflage, check sum checks, destruction of the Import Address Table and other deceptive techniques to make analysis harder.
Packers and Protectors	Use run time packers and protectors to obfuscate code and data and make it hard to unpack to find the original entry point. This includes packers that use their own virtual machines such as HyperUnpackme2.
Rootkits	Insert rootkits at Ring 0 to take control of the way the operating system manages processes and use deception to hide malicious processes

METHODOLOGY

Based on suggestions given by various authors (Reed, 1998; Webster et al., 2002; Green et al., 2006; Levy et al., 2006; Ma et al., 2006; Armitage et al., 2008) for writing a literature review paper, the following steps were adopted to search the sources for the review.

The initial stage for the review covered doctoral thesis from various international universities because they have been reviewed at higher exams. Online library has been used as a source for all doctoral and master theses. Keywords such as

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

"Anti Virus", "Anti Virus analysis" have been used to identify relevant thesis for the study.

For the second stage leading journals and international conference papers were selected as these have gone through scientific peer reviews in order to be accepted at journals or conference proceedings.

The list of research papers (from serial number 1-10) and thesis (from serial number 11-14) included in the review and their classifications with respect to their topics and contributions have been summarized in Table 3.

S.No.	Title	Author	Contribution
1	Survey on Anti Virus detection methods	Vinod, Gaur	The work focused on various Anti Virus detection methods like signature based detection, reverse engineering of obfuscated code to detect malicious nature.
2	Anti Virus Forensics-Detecting the Unknown	Martin, Overton (2008)	Looked at what tricks, tools and techniques one can use to help establish the true state of the suspect system. Focused on step by step approach of what tools to use, what to look for & what to do with any suspicious files.
3	A Threat to Cyber Resilience: A Anti Virus Rebirthing Botnet.	Brand, Valli, Woodward (2011)	Paper presented a threat to cyber resilience in the form of a conceptual model of a Anti Virus rebirthing botnet.
4	Lessons learned from an	Brand, Valli,	Analyst must understand the anti-analysis technique that can be

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

	Investigation into the Analysis Avoidance Techniques of Malicious Software.	Woodward (2011)	employed & how to mitigate them, the limitations of existing tools & how to employ an appropriate analysis methodology to uncover the intent of Anti Virus.
5	Anti Virus Forensics: discovery of the intent of deception	Brand, Valli, Woodward (2011)	Suggested that the discovery of intent of deception may be a very good indicator of an underlying malicious objective of the software under investigation.
6	Detection & Preservation of New & Unknown Anti Virus using Honeypots.	Kumar, Pant	Proposed a system where honeypots used for generating & broadcasting instant cures for new & unknown Anti Virus in a network.
7	SPiKE: Engineering Anti Virus Analysis Tools using Unobtrusive Binary-Instrumentation	Vasudevan, Yerraballi	Developed a new dynamic coarse grained binary instrumentation framework codenamed SPiKE that aids in the construction of powerful Anti Virus analysis tools to combat Anti Virus that are becoming increasingly hard to analyse.
8.	The Anti Virus Analysis Body of Knowledge(MABOK)	Valli, Brand (2008)	Presented a foundation for a Anti Virus (MABOK) i.e, required to successfully forensically analyse Anti Virus.
9.	Static analysis of executables to detect malicious patterns	Christodorescu, Jha (2003)	Presented an architecture for detecting malicious patterns in executables that is resilient to common obfuscation transformations.
10.	TT Analyze: A tool for analyzing Anti Virus.	Bayer, Kruegel, Kirda (2006)	Presented a tool TT analyzer for dynamically analyzing the behavior of windows executables.
11.	Dealing with next generation Anti Virus.	Paleari (2011)	Presented a new framework for improving behavior based analysis of suspicious programs, that allows an end user to delegate security labs, the execution and the analysis of a program and to force the program to behave as if it were executed directly in the environment of the former.
12.	Robust & Efficient Anti Virus Analysis and host based monitoring	Sharif (2010)	1) Efficient Methods for enabling static Anti Virus analysis. 2) Making dynamic analysis approaches more robust. 3) Reversing emulator based obfuscation. 4) Anticipating obfuscations that hide trigger based behavior.
13.	Analysis avoidance techniques of malicious software.	Brand (2010)	Demonstrated anti analysis techniques can be very effective at hindering analysis by the tools typically used analysts.
14.	Data mining methods For Anti Virus detection.	Siddiqui (2008)	Presented a data mining framework to detect malicious programs.

DISCUSSION

From the review discussed so far remember that Anti Virus analysis is like a cat and mouse game. The findings reveal that there are two major techniques available for Anti Virus analysis; also it shows the defect of virtual machine, many researchers are unable to detect Anti Virus because Anti Virus has a wide range of analysis avoidance techniques. As new Anti Virus analysis techniques are developed, Anti Virus authors respond with new techniques to thwart analysis therefore detecting,analyzing and finally generating cures for them are themselves individual research topics.

CONCLUSIONS

Forensic analysis of evidences and residual traces of crimes is an ancient, tried and successful field in the realm of investigation. The latest inclusion in crime is with the advent of computers, communication and networking. The trend of growth in Anti Virus attack is increasing more rapidly. Networks have become more vulnerable and are under constant Anti Virus attacks. From a lone system (PC) to an entire organization network, no one is inescapable from the current sabotage. Financial siphoning of bank a/c, stalking, character assassination, duping are some

examples of Anti Virus attack (cyber crime). Owing to this fact, forensic digital analysis of such crimes has gained

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

immense importance in investigation of late.

ACKNOWLEDGEMENT

The corresponding author is thankful to Directorate of Forensic Science Services for providing fellowship to pursue Research and development work in forensic science.

REFERENCES

1. Arief, B. & Besnard, D. (2003). Technical and human issues in computer-based systems security. University of Newcastle upon Tyne. (CS-TR-790).
2. Armitage, A. and Keeble-Allen, D. (2008), "Undertaking a structured literature review or structuring a literature review: tales from the field", The Electronic Journal of Business Research Methods, Vol. 6 No. 2, pp. 103-14.
3. Akira Mori. "Detecting Unknown Computer Viruses - A New Approach -." *Lecture Notes in Computer Science*, pp. 226–241, 2004.
4. Aquilina J, Casey E., & Malin, C. Anti Virus Forensics Investigating and Analyzing Malicious Code; Burlington; MA: Syngress; 2008.
5. Aycock, J. (2006). *Computer Viruses and Anti Virus*. New York: Springer.
6. Bailey M, Oberheide J, Andersen J, Mao M Z, Jahanian F, Nazario J. Automated classification and analysis of internet Anti Virus. In Proceedings of the 10th Symposium on Recent Advances in Intrusion Detection (RAID'07); 178–197; 2007.
7. Bayer U, Kruegel C, Kirda E. TTAnalyze: A tool for analyzing Anti Virus. In Proceedings of EICAR; April 2006.
8. Bayer U, Moser A, Kruegel C, Kirda E. Dynamic analysis of malicious code; Journal in Computer Virology; 2:67–77; 2006.
9. Bayer U. Anubis A platform the analysis of malicious code. Journal. Retrieved from

-06-09/ANUBIS-OSSIR-EN-June-2009- v1.1.00.pdf;2009.

10. Bayer U, Kirda E, Kruegel C. Improving the efficiency of dynamic Anti Virus analysis. In Proceedings of the 25th Symposium on Applied Computing (SAC); Lusanne; Switzerland; March 2010.
11. Bergeron J, Debbabi M, Erhioui M M, Ktari B. "Static Analysis of Binary Code to Isolate Malicious Behavior." In Proceedings of the 8th Workshop on Enabling Technologies on Infrastructure for Collaborative Enterprises (WETICE'99); 184–189; 1999.
12. Bergeron J, Debbabi M, Desharnais J, Erhioui M M, Lavoie Y, Tawbi N. "Static Detection of Malicious Code in Executable Programs." Symposium on Requirements Engineering for Information Security (SREIS'01); 2001.
13. Brand M, Valli C, Woodward A. A Threat to cyber resilience: A Anti Virus rebirthing Botnet. In the Proceedings of the 2nd International Cyber Resilience Conference; 2011
14. Brumley D, Hartwig C, Liang Z, Newsome J, Song D, and Yin H. Towards automatically identifying trigger-based behavior in Anti Virus using symbolic execution and binary analysis. Technical Report CMU-CS-07-105; Carnegie Mellon University; 2007.
15. Chen X, Garfnkel T, Christopher E L, Subrahmanyam P, Waldspurger C A, Boneh D, Dwoskin J, Dan R. K. Ports. Overshadow: a virtualization-based approach to retroftting protection in commodity operating systems. Operating Systems Review, 42(2); 2008.
16. Chess B, & West J. *Secure Programming with Static Analysis*. Upper Saddle River; 2007.
17. Cohen F. "Experiments with Computer Viruses;"1984.
18. Chouchane M, Walenstein A, Lakhotia A. Statistical signatures for fast filtering of instruction-substituting metamorphic Anti Virus. In Proceedings of the 2007 ACM workshop on Recurring malcode; 2007.
19. Christodorescu M, Jha S. Static analysis of executables to detect malicious patterns. In Proceedings of the 12th

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

USENIX Security Symposium; 12–12; 2003.

20. Christodorescu M, Jha S, Seshia A S, Song X D, Bryant E R. Semantics aware Anti Virus detection. In *IEEE Symposium on Security and Privacy*; 32–46; 2005.
21. Debbabi M, Girard M, Poulin L, Salois M, Tawbi N. "Dynamic Monitoring of Malicious Activity in Software Systems." In Symposium on Requirements Engineering for Information Security (SREIS'01); 2001.
22. Falliere N. Windows Anti-Debug Reference. Retrieved October1, 2007 from http://www.securityfocus.com/infocus/1893;2 007
23. Ferbrache D. *A Pathology of Computer Viruses*. Springer-Verlag, 1992
24. Ferrie P. Anti-Unpacker Tricks. Paper presented at the 2nd International Caro Workshop. From http://www.datasecurity event.com/uploads/unpackers.pdf;2008
25. McGraw G, Morrisett G. Attacking malicious code: A report to the infosec research council IEEE Software; 2000;33–44.26) Garfnkel T, Rosenblum M. A virtual machine introspection based architecture for intrusion detection. In Proceedings of the Symposium on Network and Distributed Systems Security (NDSS); San Diego;CA; USA; February 2003
26. Gerrold D. *When Harlie Was One*. Doubleday; 1972.
27. Green, B.N., Johnson, C.D. and Adams, A. (2006), "Writing narrative literature reviews for peer-reviewed journals: secrets of the trade", Journal of Chiropractic Medicine, Vol. 5 No. 3, 101-17.
28. Grugq .The Art of Defiling, Defeating Forensic Analysis on UNIX File Systems. Paper presented at the Black Hat Asia ;2003; Singapore.
29. Gutmann P. "The Commercial Anti Virus Industry.", 2007. D.Wagner and D. Dean.

"Intrusion detection via static analysis." IEEE Symposium on Security and Privacy; 2001.

30. Harbour N. Stealth Secrets of the Anti Virus Ninjas. Retrieved October 20, 2007 from https://www.blackhat.com/presentations/bh- usa-07/Harbour/Presentation/bh-usa-07 harbour.pdf;2007.
31. Hoffman J L. *Rogue Programs: Viruses, Worms, and Trojan Horses*. Van Nostrand Reinhold; 1990.
32. Hofmeyr S, Forrest S, Somayaji A. "Intrusion detection using sequences of system calls." *Journal of Computer Security*; 151–180; 1998.
33. Hudak T. Automating Anti Virus Analysis. Hakin9; 64-69; July 2009
34. International Secure Systems Lab, Vienna University of Technology, Eurecom France & UC Santa Barbara. (2008). Anubis: Analyzing Unknown Binaries; Retrieved October 4, 2008; from http://anubis.iseclab.org.
35. Innes S, Valli C. Honeypots: How do you know when you are inside one? Paper presented at the 4th Australian Digital Forensics Conference; Edith Cowan University; Perth; Western Australia; 2006.
36. Jiang, Wang X .Out-of-the-Box" monitoring of VMbased high-interaction honeypots. In Proceedings of the International Symposium on Recent Advances in Intrusion Detection (RAID); 2007.
37. Kirda E, Kruegel C, Banks G, Vigna G, Kemmerer A R. Behavior-based spyware detection. In *Proceedings of the 15th USENIX Security Symposium*; 2006; 19–19.
38. Ko C, Ruschitzka M, and Levitt K. "Execution Monitoring of Security-Critical Programs in Distributed Systems: A Specification-Based Approach. In *Proceedings of the 1997 IEEE Symposium on Security and Privacy*; 1997.
39. Krack. Defeating Norman Sandbox. Retrieved July 21, Kruegel C, Robertson W, and Vigna G. Detecting kernel-level rootkits through binary analysis. In *Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC)*; 2004.
40. Lee T , Mody J J. Behavioral classification. In Proceedings of EICAR; April 2006.
41. Levy, Y. and Ellis, T.J. (2006), "A systems approach to conduct an effective literature review insupport of information systems

Security and Privacy; 2008 May; Oakland; CA.

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

50. Perrig A, Gligor V, and Vasudevan A. XTREC: secure realtime execution trace recording and analysis on commodity platforms; Technical report, Carnegie Mellon University; 2010.
51. Porras P, Saidi H, Yegneswaran V. A Multi- perspective Analysis of the Storm (Peacomm) Worm. Retrieved Dec 7, 2007 from
181-212.
43. Masood S G. Anti Virus Analysis for Administrators. Retrieved 17 March, 2007 from http://www.securityfocus.com/infocus/1780;2004.
44. Masri W, Podgurski A. "Using Dynamic Information Flow Analysis to Detect Against Applications." In Proceedings of the 2005 Workshop on Software Engineering for secure sytems Building Trustworthy Applications;2005.
45. Moser A, Kruegel C, and Kirda E.Limits of static analysis for Anti Virus detection. In *Proceedings of the 23rd Annual Computer Security Applications Conference (ACSAC)*; 2007.
46. Moser A , Kruegel C, and Kirda E. Exploring multiple execution paths for Anti Virus analysis. In *Proceedings of 2007 IEEE Symposium on Security and Privacy*; 2007; Oakland, CA.
47. Norman. Submit file for Sandbox analysis. Retrieved April 12; 2008;from http://www.norman.com/microsites/nsic/Submit/en-us
48. Overton M .Anti Virus Forensics: Detecting the unknown. Paper presented in Virus Bulletin Conference at the Westin Hotel;Ottawa; Canada;October 1st-3rd 2008.
49. Payne B D, Carbone M, Sharif M, and Lee W. Lares: An architecture for secure active monitoring using virtualization. In
Proceedings of the IEEE Symposium on Technical report; 1998.
61. Sekar R, T. Bowen and M. Segal. "On Preventing Intrusions by Process Behavior Monitoring." In USENIX Intrusion Detection Workshop, 1999.
62. Sekar R, Bendre M, Bollineni P, Dhurjati D. "A Fast Automaton-Based Approach for Detecting Anomalous Program Behaviors." In IEEE Symposium on Security and Privacy; 2001.
SRITechnical-Report-10-01-Storm-Analysis.pdf
52. Rabek C J, Khazan I R, Lewandowski M S, Cunningham K R. "Detection of Injected, Dynamically Generated, and Obfuscated Malicious Code." In Proceedings of the 2003 ACM Workshop on Rapid Malcode; 76–82; 2003.
53. Raymond W Lo, Karl N Levitt, Ronald A Olsson. "MCF: A Malicious Code Filter." Computers and Security; 541–566; 1995.
54. Reed, L.E. (1998), "Performing a literature review", 28th Annual Frontiers in Education Conference, FIE'98, Vol. 1, pp. 380-3.
55. Riley R, Jiang X, Xu D. Guest-transparent prevention of kernel rootkits with VMM-based memory shadowing. In Proceedings of the 11th International Symposium on Recent Advances in Intrusion Detection; 2008.
56. Rutkowska J. (2006). Introducing Stealth Anti Virus Taxonomy. Retrieved April 12; 2009
from
http://www.invisiblethings.org/papers/malware-taxonomy.pdf.
57. Rutkowska J. (2006a). Fighting Stealth Anti Virus - Towards Verifiable OSes. Journal. Retrieved from http://www.invisiblethings.org/papers/towards_verifiable_systems.ppt
58. Rutkowska J. (2006b). Introducing Stealth Anti Virus Taxonomy. Journal. Retrieved from http://www.invisiblethings.org/papers/Anti Virus-
taxonomy.pdf
October 4, 2008, from http://www.virustotal.com/en/virustotalf.html
72. Vuksan M., Peričin T, Milunovic V. Fast & Furious Reverse Engineering with TitanEngine.
63. Seshadri A, Luk M, Qu N, and Perrig A. SecVisor: A tiny hypervisor to provide lifetime kernel code integrity for commodity OSes. In Proceedings of the ACM Symposium on Operating Systems Principles.ACM, 2007.

ISSN: 0971-1260
Vol-22-Issue-4-October-December-2019

64. Sharif M, Lee W, Cui W, Lanzi A. Secure in vm monitoring using hardware virtualization. In Proceedings of the ACM Conference on Computer and Communications Security (CCS); 2009.
65. Shoch F J , Hupp A J. "The Worm Program-Early Experience with a Distributed Computation.";172–180; 1982.
66. Skoudis E, Zeltser L . Anti Virus Fighting Malicious Code. New Jersey: Prentice Hall; 2004.
67. Smith S, Quist D (2006). Hacking Anti Virus: Offense is the new Defense. Retrieved July 24, 2007 from http:// www. offensivecomputing.net /dc14/ valsmith_ dquist_ hacking_ Anti Virus_ us06.pdf.
68. Szor P. The Art of Computer Virus Research and Defense. Addison-Wesley; 2005.
69. Valli C, Brand M. Anti Virus Analysis Body of Knowledge. Paper presented at the 6th Australian Digital Forensics Conference; Edith Cowan University; Mount Lawley Campus; Western Australia; 2008.
Vasudevan and R. Yerraballi. Spike: Engineering Anti Virus analysis tools using unobtrusive binary-instrumentation. In Proceedings of the 29th Australasian Computer Science Conference, pages 311–320, 2006.
Black Hat USA 2009, from http://www.reversinglabs.com/blackhat-Titanium_gone_Blackhat-USA-09-Slides
73. Webster, J. and Watson, R.T. (2002), "Analyzing the past to prepare for the future: writing
74. a literature review", MIS Quarterly, Vol. 26 No. 2.69)Willems C, Holz T, and Freiling F. CWSandbox: Towards automated dynamic binary analysis. *IEEE Security and Privacy*; 2007.
75. Wysopal C. Good Obfuscation, Bad Code. Retrieved May 03 2009, from http://www.securityfocus.com/columnists/498?ref=oc; 2009.
76. Yan W, Zhang Z, Ansari N. Revealing Packed Anti Virus. IEEE Security and Privacy 6 (5); 65-69; 2008.
77. Yason M. The Art of Unpacking. Retrieved Feb 12, 2008 from https://www.blackhat.com/presentations/bh-usa-07/Yason/Whitepaper/bh-usa-07-yason-WP.pdf; 2007.
78. Yin H, Song D, Egele M, Kruegel C, Kirda E . Panorama: capturing system-wide information flow for Anti Virus detection and analysis. Paper presented at the Proceedings of the 14th ACM conference on Computer and communications security; 2007.
79. Zeltser L. Reverse Engineering Anti Virus: Tools and Techniques Hands-On. Bethesda: SANS Institute; 2007.
80. Zhou Y, Meador I W. Anti Virus detection using adaptive data compression. Paper presented at the Proceedings of the 1st ACM workshop on Workshop on AISec; 2008.