University of Plymouth

PEARL **https://pearl.plymouth.ac.uk**

Faculty of Science and Engineering School of Engineering, Computing and Mathematics

2019-08

A proactive malicious software identification approach for digital forensic examiners

Ali, M

http://hdl.handle.net/10026.1/14705

10.1016/j.jisa.2019.04.013
Journal of Information Security and Applications
Elsevier

All content in PEARL is protected by copyright law. Author manuscripts are made available in accordance with publisher policies. Please cite only the published version using the details provided on the item record or document. In the absence of an open licence (e.g. Creative Commons), permissions for further reuse of content should be sought from the publisher or author.

Journal of Information Security and Applications 47 (2019) 139–155

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

A proactive malicious software identification approach for digital forensic examiners

Muhammad Ali[a], Stavros Shiaeles[a,*], Nathan Clarke[a], Dimitrios Kontogeorgis[b]

[a] Centre for Security, Communications, and Networks Research (CSCAN), School of Computing and Mathematics, Plymouth University, UK
[b] School of Applied Science, Open University of Cyprus, Latsia, Nicosia, Cyprus

ARTICLE INFO

Article history:
Available online 16 May 2019

Keywords:
Digital forensics
Malware
Machine learning
Registry hives
Windows Registry
Windows 7/8/10
Sandbox
Agentless sandbox
Cuckoo

ABSTRACT

Digital investigators often get involved with cases, which seemingly point the responsibility to the person to which the computer belongs, but after a thorough examination malware is proven to be the cause, causing loss of precious time. Whilst Anti-Virus (AV) software can assist the investigator in identifying the presence of malware, with the increase in zero-day attacks and errors that exist in AV tools, this is something that cannot be relied upon. The aim of this paper is to investigate the behaviour of malware upon various Windows operating system versions in order to determine and correlate the relationship between malicious software and OS artifacts. This will enable an investigator to be more efficient in identifying the presence of new malware and provide a starting point for further investigation.

The study analysed several versions of the Windows operating systems (Windows 7, 8.1 and 10) and monitored the interaction of 90 samples of malware (across three categories of the most prevalent (Trojan, Worm, and Bot) and 90 benign samples through the Windows Registry. Analysis of the interactions has provided a rich source of knowledge about how various forms of malware interact with key areas of the Registry. Using this knowledge, the study sought to develop an approach to predict the presence and type of malware present through an analysis of the Registry. To this end, different classifiers such as Neural Network, Random forest, Decision tree, Boosted tree and Logistic regression were tested. It was observed that Boosted tree was resulting in a correct classification of over 72% – providing the investigator with a simple approach to determining which type of malware might be present independent and faster than an Antivirus. The modelling of these findings and their integration in an application or forensic analysis within an existing tool would be useful for digital forensic investigators.

© 2019 Elsevier Ltd. All rights reserved.

1. Introduction

As malware evolves and becomes more complex, malicious attackers are able to adapt their behaviour depending on the system they wish to infect. Malicious software can only be revealed after the recognition of specific factors of the system and the combination of many parameters and conditions. For example, a particular malware might reveal it's behaviour when installing on a Windows 7 platform or when specific software is installed on the victim's computer (for example a PDF Reader) and to remain totally inactive in any other situation. Similarly, it can reveal a part of his behaviour, while parts of the functionality remain hidden until certain conditions that will cause additional activity. Attempts have been made to uncover malware activated behaviour [8,47] but it has also shown for it to be possible to trick such analyzers [58]. During the examination of a case, the possibility always exists that digital evidence or criminal activity is the result of malware activity. It could be that the owner of a computer system is unjustly suspected due to the presence of malicious software. Therefore, in each case prior to the recording of evidence, a thorough investigation for the presence of malware should be undertaken. Traditionally, this is achieved using one or more Anti-Virus (AV) systems. However, weaknesses in AV technology and the increasing presence of zero-day vulnerabilities make them less than full-proof.

Many researchers have conducted studies to find digital artifacts on the Windows operating system, including earlier versions of Windows, were analysed, such as Vista [53], 7 [64] and 8 [61]. Further research has also been undertaken on the study of specific operating regions such as the Registry [44], volatile memory [23,56,57,64], USB devices [12,19] and the file system [9,10,40]. In this research work a comparative study of three core Windows OSs (Windows 7, 8.1 and 10) is undertaken in order to study whether the version of OS has an impact over the behaviour and performance of malicious software. This will provide digital forensics

* Corresponding author.
E-mail address: sshiaeles@ieee.org (S. Shiaeles).

https://doi.org/10.1016/j.jisa.2019.04.013
2214-2126/© 2019 Elsevier Ltd. All rights reserved.

analysts with invaluable help [13,32], as they will have a guide for the locations to which are expected to have digital evidence. With a targeted investigation at specific locations, it is possible to identify whether a system is infected with malware or not. The paper also develops and evaluates an approach to automatically predict which type of malware is present. This allows forensic examiners to more quickly and reliably identify the presence and type of malware.

2. Background and related work

The detection of malware through an analysis of unknown executables is not a new problem. Consequently, many solutions already exist. These solutions can be divided into two categories: static and dynamic analysis.

2.1. Static analysis

In static analysis, an incident response team analyses the code or the structure of a program to determine the functionality without running the program [60]. First steps include the use of all available anti-virus programs. This could give information to a known malware for which signature is available, saving valuable time in the process. A major disadvantage of this technique is the dependence upon the detection of the virus based largely on file signatures. Malicious code developers can easily change the code in order to avoid detection [21]. Another technique used in the static analysis is binary code disassembling, which convert the binary code into an assembly and then analytical techniques control data flow resulting in a report of the running program. A series of binary code analysis techniques [16,18,34] have been presented for the detection of different types of malware. The advantage of static analysis is that its carried out quickly and that can cover the entire application code. Whilst, there is rich literature on static analysis techniques, which indicates that many problems can be tackled well in practice due to predictability, often this is because it is being applied to real applications rather than malware. Unfortunately, since malware is directly created by cyber criminals it can be deliberately crafted so that it is difficult to analyse. Specifically, the attacker can make use of technical binary obfuscation to prevent both the disassembly of the code and analysis, methods which are used by static analysis techniques.

2.2. Dynamic analysis

The dynamic analysis techniques of malware behaviour characterized by the analysis of the actual instructions of a program or the results it brings the program to the operating system. Compared with the static approach, dynamic analysis is less susceptible to various code obfuscation techniques [47]. Christodorescu et al. [17] introduce the specifications of malware using data flows between the system calls. They found the actual relationships between system calls are difficult to overcome with random system calls. Since then, this knowledge of malicious software has been widely used in malware analysis tasks such as extraction of distinct malware functions, mining the difference between malware behaviour and benign behaviour of the program [26], determining malware families in which samples are sharing common functions [3,5,51] and to detect malicious behaviour [7,7,33,37]. Another method uses a representative audience behaviour chart for all samples of malware in a family, instead of a behaviour chart per case. The proposed approach is valid and effective since most new malware variants are from known families [28,52,69]. Despite various metamorphic and polymorphic blackouts, samples of malicious software within the same family tend to reveal similar malicious behaviour [36].

The most popular method of analysing the malware operation in a safe way is to use sandbox technology. The sandbox is running as a separate system, contains the untrusted program and prevents any action from accessing the real network and often provides network services for malware in a form of "black hole." If the untrusted program makes a DNS request, for example, the sandbox will answer the question, usually with 127.0.0.1 (loopback).

Since the spread of metamorphic and polymorphic viruses, dynamic analysis of malware has been established as an effective approach to understanding and classifying malware by observing the execution of malware samples in quarantine environment [24,72]. The interaction between the execution of the malicious sample and operating system allows dynamic malware analysis systems to collect those behavioural characteristics that help shape technical defense.

A problem that was found in modern viruses is that the malicious code is often equipped with detection routines that check for the presence of a virtual machine or a simulated operating system environment. When such an environment is detected, the malware modifies its behaviour and the analysis yields incorrect results or even worst, the malware stops to function making analysis impossible. Moreover, some malware also checks for software (even material) having breakpoints to detect whether the program is running in a debug program. In order to bypass the aforementioned problems, the analysis environment should be invisible to malicious code, comprehensive and cover all aspects of the interaction of an environmental program.

Efforts to investigate the possible prevention of malware incidents have prompted earlier studies on malicious codes [4,20,74]. Current research in cybersecurity focuses on the characterization and modelling of specific attacks, with the aim of understanding the mechanisms of penetration, detection, and response. As cyber threats are increased both in number and in complexity, it has increased the interest in infectious malware [38]. In theory, one of the interesting issues is the creation of reliable mathematical models that can be applied to effectively describe and forecast the evolution of malicious computer software. Since the spread of malicious code is similar to the biological epidemics [66], some epidemiological models have been employed to study the behaviour of malicious software [15,35,45,46,59]. In addition, new strategies and methodologies necessary to prevent invasions and addressing their effects [27].

3. Experimental methodology

The purpose of this research was to examine in a dead-box mode the impact that malware has on the Windows Registry. Furthermore, the research sought to understand what differences exist in differing versions of the OS. To this end, 90 samples of both malware (split between Trojans, worms, and bot) and cleanware were selected to provide a robust and comprehensive analysis. The Microsoft's Windows operating systems was focussed upon as, it is still the prominent OS in use today [48].

3.1. Virtual lab

The analysis laboratory consists of two testbeds. The first one is running locally on a host machine with a CPU Intel Core i7-4790, 16 GB RAM and Windows 10 Pro as the bare metal configuration. In this machine we installed VMware Workstation Pro 12 [70] and we created a Virtual Machine|(VM) with Ubuntu 16.04 operating system which hosted the Cuckoo Sandbox. Ubuntu compatibility with the Cuckoo Sandbox [50] is excellent and has been used by other investigators [57] for the same purpose (Fig. 1). Cuckoo utilised a methodology for the understanding of malicious code based upon sandboxing [29] and execution of arbitrary code

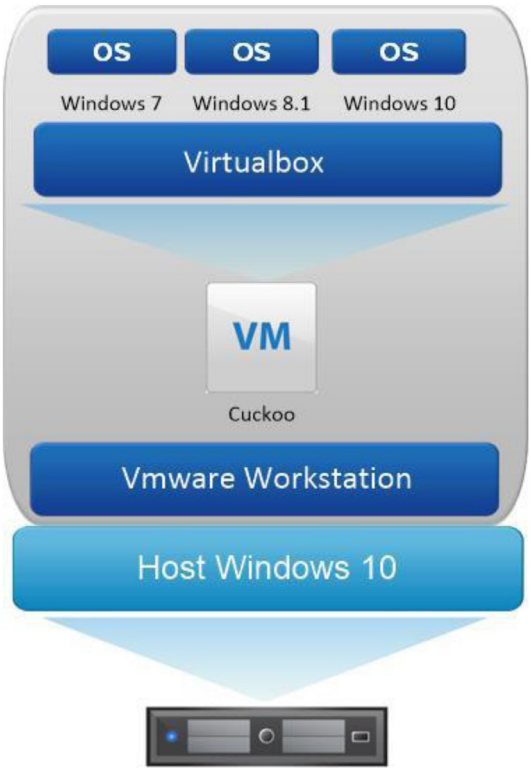


Fig. 1. Virtual lab architect.

in a controlled manner that allows direct observation of results through a Python agent script. The script in Python and the libraries are Cuckoo's important components [31]. The system consists of a server where the Cuckoo software is installed along with a virtualization program such as Oracle VirtualBox [22]. In Oracle VirtualBox, various VMs can be created with different versions of Operating Systems where the agent of Cuckoo should be also installed. In the guest VM an examiner has to make sure to disable any services that may prevent the malware executed successfully such as User Account Control (UAC), automatic updates, Antivirus and firewall in case of Windows Operating System. The isolated environment allows the sample to run without adversely affecting the system host computer or the quest and simultaneously documenting evidence, such as open ports, registry keys, IP addresses, file incorporated and domain names, which are important for an examiner. Once the desired state of the system has accomplished, a system snapshot is taken. This snapshot can be used to restore the system to a known clean state after the sample is analysed. In our case in VirtualBox, three VMs were created with the following characteristics: CPU a core of the Intel Core i7-4790, 2 GB RAM and Operating System Windows. In each of these three VMs, a different version of Windows Operating System is installed and specifically 7, 8.1 and 10. In order Cuckoo to communicate with each guest machine, Cuckoo network card and a unique guest VM card were connected to a virtual isolated network (192.168.56.0/24) as shown in Fig. 2.

The second testbed was hosted on cloud and we utilised two cloud sandboxes to withdraw as much information as possible in order to find more unique registry hives from the malware and cleanware. For this experiment we utilised an Agentless (VMRay Analyzer) and AI-based (SNDBOX) sandbox. VMRay Analyser as aforementioned is an agentless sandbox cloud solution and the reason choosing this platform is that some sophisticate malware usually monitor the running environment and to prevent their discovery they usually stop their execution which provides insignificant features to the analysis [1]. SNDBOX applies an invisible kernel mode agent and AI to offer the next generation Sandbox, extending the individual capabilities and expertise of security and research teams through AI, dynamic analysis and network mapping. It is Located between the User mode and Kernel mode, SNDBOX's invisible agent deceives malware into executing its full range of intended functionality, revealing its true malicious nature, intent, and capabilities [63].

3.2. Standardized naming scheme for malware

Security analysts and researchers from different AV companies in 1991 developed a standardized naming scheme for malware known as Computer Antivirus Research Organization (CARO) ("A New Virus Naming Convention (1991) - CARO - Computer Antivirus Research Organization," n.d.). The philosophy behind the development of this standard is to remove the confusion among the users

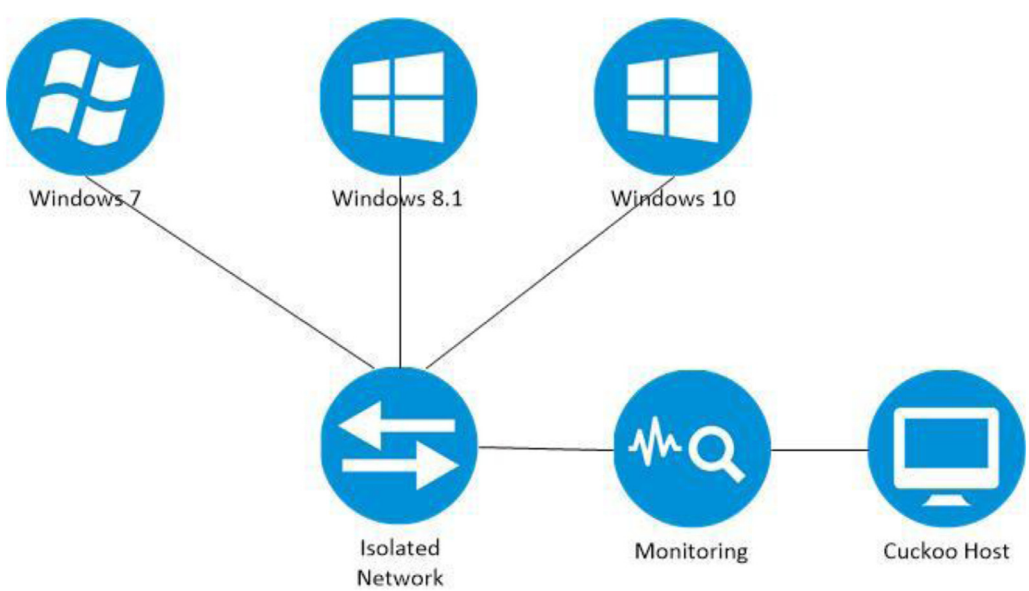


Fig. 2. Virtual network configuration.

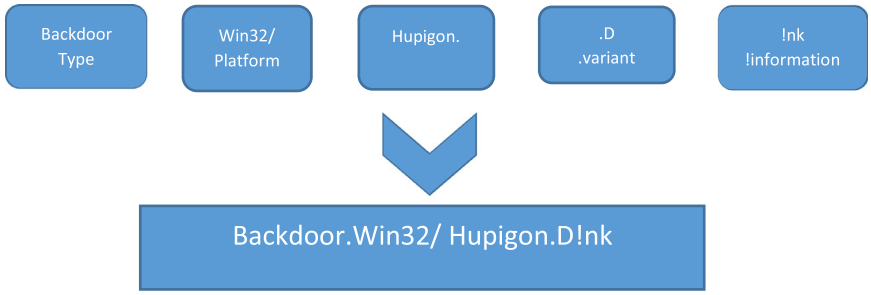


Fig. 3. CARO malware naming scheme ("Malware names | Microsoft Docs," n.d.).

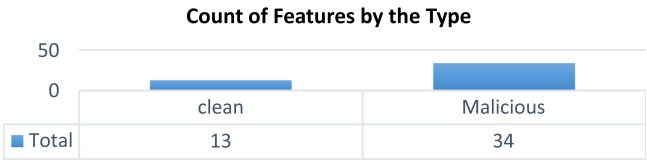


Fig. 4. The ratio of clean and malicious hives.

and AV-Vendors by having a common standard or syntax for naming malware. The generic form of this format is mentioned below

<malware type>://<platform>/<family name>.<group name>.<infective length>.<sub-variant><devolution><modifiers>

In the string above only family name is compulsory and the rest of the fields are optional. Although most companies claim that they follow the CARO scheme, but in practice only Microsoft is using this convention in their AV software for MSE or the Win8 version and windows defender etc. (Fig. 3).

In this research work we used the CARO naming scheme to name malware as shown in Section 3.3.

3.3. Malware samples used

Often, malware investigators have to deal new threats and unknown executable. In some cases, there are scenarios where you can handle malware that already knows its name and is classified, e.g. for research purposes as in this work. To analyse such malicious software, there are many places where one researcher can collect known samples. The Lenny Zeltser, who is the head of the private SANS Institute [55], recommends several free resources on his website [73]. The samples of malicious code used in this research were taken from Malware.lu [41], Virussign [67], Vx Heaven [71], Malekal [39] and MalwareTips [42]. Ninety samples of malicious code were selected, 41 Trojan, 28 Worms, and 21 Bot. These three malware families were selected (Table 1) as they are the main categories that are detected more often [2,43,65].

3.4. Clean samples used

The need to differentiate the clean registry hives from malicious hives, cleanware samples such as chrome, teamviewer, skype etc. were collected and analysed. During the analysis emphasis was given on collecting system changes along with register hives in order to have more information. The Table 2 below shows the samples used as well as the type of each sample.

3.5. Sandbox analysis procedure

To export information from the samples we performed experiments in three different environments. In the first experiment, Cuckoo was utilised for behaviour analysis of malware of files mentioned in Sections 3.3 and 3.4 above. For each analysis request, a separate subfolder containing all the reports is produced with raw logs, .pcap files, images and any other information obtained during the analysis. Using the Cuckoo as the main malicious software analysis tool, each sample was studied in three different software environments (Windows 7, 8.1 and 10) and the results of the analysis are stored in a suitable form for further study and analysis.

The program sends the sample to the virtual machine that we have selected in the settings file. When injection of the sample into the operating system has completed successfully, Cuckoo monitors all system activity and records it. Once the analysis of the virtual machine is terminated, the .html file with the report of the analysis is created.

The second experiment was completed in two different cloud sandboxes named VMRay analyzer and SNDBOX. In both these sandboxes, benign and malicious samples of Sections 3.3. and 3.4 respective were executed. For each analysis request, a separate subfolder containing all the reports is produced, the raw logs, .pcap files, images, JSON and any other information obtained during the analysis.

3.5.1. Dataset preparation

Data constitute the input/output variables required to make a prediction. Usually, data comes in two forms either structured or unstructured data. In this research we have taken structure data which implies that data are defined and properly labelled. In order to label data VIT and Virus Total reputation scoring were introduced, to categorize samples as malicious and benign. VirusTotal inspects items with over 70 antivirus scanners database along with URL/domain blacklisting services, in addition to a myriad of tools to extract signals from the studied [68]. Cuckoo sandbox uses VirusTotal to perform the experiments, moreover, in the case of VMRay analyzer VTI score is used to label the samples.

To evaluate the proposed research and create the raw and integrated feature set, malware and benign samples were collected from a different source as mentioned in the above section. In order to validate the propose works different portion of samples were taken for validation purpose.

3.6. Pre-processing and feature generation

In this stage data were processed and cleaned from noise and irrelevant entries and string information was extracted from the logs file generated by the different sandboxes and features set were constructed. In this research, 34 register hives were identified for malware and 13 register hives for cleanware as shown in the Table 3. Furthermore, these strings were converted into binary feature vector, so they can be given as input to the machine learning algorithm.

Table 1
The malware samples that were used.

	Category	Virus name	SPY-STEAL DATA	C&C	BACKDOOR	STEALTH
1	Trojan	Trojan-Spy.Win32.Zbot.wijf	X	X		
2	Trojan	Trojan.GenericKD.3015891		X		
3	Trojan	Trojan.GenericKD.3015909			X	
4	Trojan	Trojan/Win32.Yakes			X	
5	Trojan	Trojan.GenericKD.3016131			X	
6	Trojan	Trojan/W32.KRBanker		X		
7	Trojan	Trojan-Spy.Win32.FlyStudio.ij			X	X
8	Trojan	Trojan-Dropper.Win32.Injector.nyds			X	
9	Trojan	Trojan.Zboter	X	X		
10	Trojan	Trojan-Spy.Win32.Recam.yue	X	X		
11	Trojan	Trojan. Tesla!1.A322		X		X
12	Trojan	Trojan.Win32.Waldek.cbp			X	
13	Trojan	Trojan.Win32.Waldek.cbm			X	
14	Trojan	Trojan.Win32.Dridex.v	X	X		X
15	Trojan	Trojan.Win32.Tepfer.psxezj	X	X		
16	Trojan	Trojan.Win32.Yakes.owmp			X	
17	Trojan	Trojan.Win32.KeyLogger.auqd	X			
18	Trojan	Trojan.GenericKD.3023498		X		
19	Trojan	Trojan.Generic.8742442	X	X		X
20	Trojan	Trojan.Generic.7738292	X			
21	Trojan	Trojan.Generic. AAA._xeDropperSpywareTrojan	X	X		
22	Trojan	Trojan.Generic .Badi	X	X		X
23	Trojan	Trojan.Win32.CretClient.exe				
24	Trojan	Trojan.Generic .InstallBC201401		X		
25	Trojan	Trojan.Generic.pony	X			
26	Trojan	Trojan.Generic.Potao_Dropperswdecoy				X
27	Trojan	Trojan.Win32.zeus	X	X	X	X
28	Trojan	Trojan.Generic.kotbjxfkzeq	X			
29	Trojan	Trojan.Generic.Locky	X	X		X
30	Trojan	Trojan. Win32.njRAT.exe	X	X		X
31	Trojan	Trojan.Generic.pafish	X			
32	Trojan	Trojan.Win32win32.duqu				
33	Trojan	Trojan.Generic.Cerber.exe				X
34	Trojan	Trojan. Win32Mole.exe				X
35	Trojan	Trojan. Win32.Spora.exe				X
36	Trojan	Trojan.Win32GrandCrab-01.exe				X
37	Trojan	Trojan. Win32.Delf.xo			X	
38	Trojan	Trojan. Win32.DarkTequila.exe				
39	Trojan	Trojan. Win32.psiphon.exe	X			
40	Trojan	Trojan.Generic.yigzwl				X
41	Trojan	Trojan.Generic.Vcffipzmnipbxzdl				X
42	Worm	Win32.Gamarue	X	X		X
43	Worm	W32.Cridex.A.worm	X	X		X
44	Worm	Worm.VBS.Agent		X		
45	Worm	Worm.Win32.3DStars			X	X
46	Worm	Worm.Generic3.PEM			X	
47	Worm	Worm.Win32.Mira.A	X			
48	Worm	Worm.Generic2.CMVO	X			
49	Worm	Worm.Win32.Cake				X
50	Worm	Worm.Win32.Fever	X			X
51	Worm	Worm.Win32.Monkey.exe	X			
52	Worm	Worm.Win32.Mydoom.a.exe			X	X
53	Worm	Worm.Win32.Pikachu.exe	X			
54	Worm	Worm.Win32.Postman.exe				X
55	Worm	Worm.Win32.Sharpei.a.exe				X
56	Worm	Worm.Win32.Silver.exe				X
57	Worm	Worm.Win32.Sobig.exe	X	X		
58	Worm	Worm.KOOBFCE.SMC	X		X	
59	Worm	W32/Wabot	X	X		
60	Worm	Worm.vid.exe	X			
61	Worm	Email-Worm.Win32.Mydoom.l			X	X
62	Worm	Email-Worm.Win32.Naked	X			
63	Worm	Worm.Christmas-wishes.doc	X			
64	Worm	Win32.WannaCry.EXE	X		X	X
65	Worm	Win32.F7F105F9.exe				
66	Worm	Win32.2tetup.exe	X			
67	Worm	Win32.GrandCrab-01.exe	X			
68	Worm	Win32.GlobeImposter.exe	X			
69	Botnet	Win32.Lolbot.aoi			X	
70	Botnet	WORM/IrcBot.tlq	X	X	X	

(*continued on next page*)

Table 1 (*continued*)

	Category	Virus name	SPY-STEAL DATA	C&C	BACKDOOR	STEALTH
71	Botnet	W32.Jorik_Lolbot.O!tr	X		X	
72	Botnet	Win32.SdBot.aamk	X	X	X	
73	Botnet	W32.ZBot.42352	X		X	X
74	Botnet	Win32.Jorik.SdBot.e			X	
75	Botnet	MSIL.NanoBot.ibh			X	
76	Botnet	Win32.Zbot.vtii	X		X	X
77	Botnet	Win32.Ngrbot.anak				X
78	Botnet	Win32.Alinaos.G	X	X		
79	Botnet	GenericKD.2143403			X	
80	Botnet	Win32/ChkBot.A			X	
81	Botnet	MSIL/Lizarbot.A	X	X	X	
82	Botnet	Win32.Jorik.Lolbot.f	X	X	X	
83	Botnet	Win32.Zbot.sbdj	X		X	X
84	Botnet	MSIL.NanoBot.bi	X		X	
85	Botnet	Win32.Ngrbot.uyk				X
86	Botnet	Win32.Boht.qo		X	X	
87	Botnet	W32/Zbot.AJJU!tr	X		X	X
88	Botnet	Win32.VBInject				X
89	Botnet	Trickbot		X		
90	Botnet	obfuscated.js			X	

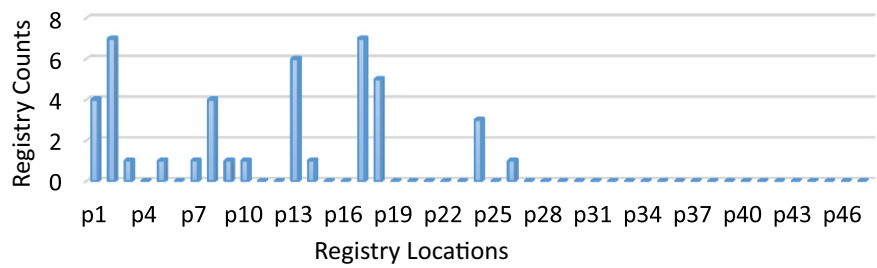


Fig. 5. The impact of malware on the registry.

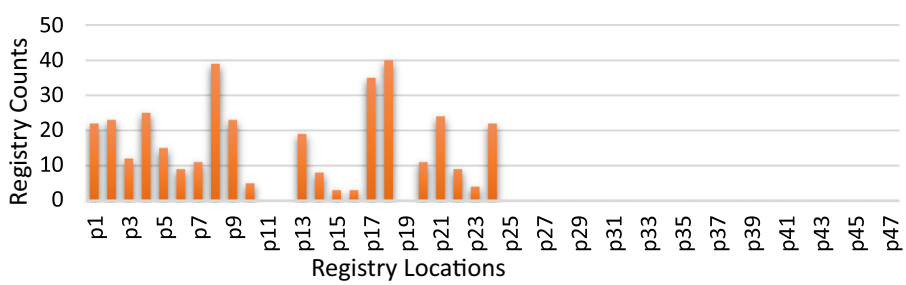


Fig. 6. The impact of Bot on the registry.

4. Experimental results

During the analysis of malware, some locations in the registry and in the Windows file system, have been recognized as important for potential contamination data. Based upon prior work, the following locations were recorded in Table 3 [6,11,14,25,30,40,49,54,62]:

The following sub-sections present an analysis of the findings based on three perspectives:

- The purpose/payload/motivation of the malware (e.g., spying or command and control)
- The type of malware (i.e., Bot, Trojan or Worm)
- The version of the operating system (i.e., Windows 7, 8, or 10)

In each case, the previously identified 47 registry and file locations are analysed against the 90 samples of both malware and cleanware to proof our initial research question. The raw results from the analysis derive can be found in Appendix A.

4.1. Analysis of malware motivation

The first analysis is concerned the research question, whether the motivation of the malware affected the frequency of digital evidence in a particular position within the Registry. As previously identified in Table 1, the types of functionality include:

- Spying and/or steal user data (Trojan)
- Communicating with a control centre to receive commands (Botnet)
- Self-Propagation (Worm)
- Benign file

Figs. 5–7 illustrate the degree to which the registry locations are affected. The analysis focussed upon an analysis of the registry against the type of malware. Fig. 4 illustrates the proportion of each type of malware upon the 34 Registry locations. The significant register keys/values for malware and benign values are mentioned below

Table 2
The cleanware samples that were used.

	Category	Sample name	Type
1	Normal	grammarlyaddinsetup.pe32	Application software plug
2	Normal	Poweriso6-x64.	Executable
3	Normal	Vlc-2-2-1-win32	Executable
4	Normal	Wireshark-win64-2.6.5	Executable
5	Normal	ProtonVPN.exe	Executable
6	Normal	Notepad.exe	Executable
7	Normal	McAfeeWebAdvisor.exe	Executable
8	Normal	Putty2.exe	Executable
9	Normal	FTPDesktopClient.exe	Executable
10	Normal	SQLiteStudio-3.2.1.exe	Executable
11	Normal	KeePass-2.40-Setup	Executable
12	Normal	LinuxLiveUSB Creator 2.9.4.exe	Executable
13	Normal	flashplayer32_install.exe	Executable
14	Normal	Firefox Setup 14.0.1	Executable
15	Normal	7za.EXE	Executable
16	Normal	GoogleUpdateSetup.exe	Executable
17	Normal	Epson512523eu.exe	Executable
18	Normal	Microsoft-Toolkit.exe	Executable
19	Normal	Googlewebdesigner_win.exe	Executable
20	Normal	PDFSAM_Installer.exe	Executable
21	Normal	FoxitReader_Setup.exe	Executable
22	Normal	TeamViewer_Setup.exe	Executable
23	Normal	Internet.Download.Manager.exe	Executable
24	Normal	TrueCrypt.exe	Executable
25	Normal	SkypeSetup.exe	Executable
26	Normal	HottNotes4.1Setup.exe	Executable
27	Normal	TorchSetup	Executable
28	Normal	GitHubDesktopSetup	Executable
29	Normal	Nektar Bolt v1.0 CE.exe	Executable
30	Normal	ForkInstaller.exe	Executable
31	Normal	hashcat32.exe	Executable
32	Normal	AdobePatchInstaller.exe	Executable
33	Normal	TWUploader.exe	Executable
34	Normal	vmnat.exe	Executable
35	Normal	SenseDriver.exe	Executable
36	Normal	ISSetup.dll	DLL
37	Normal	SrvCtl.dll	Executable
38	Normal	panfinder.exe	Executable
39	Normal	strings.exe	Executable
40	Normal	procexp.exe	Executable
41	Normal	cbhqgi.vbs	vbs
42	Normal	acc.exe	Executable
43	Normal	KutoolsforExcelSetup.exe	Executable
44	Normal	DTools.exe	Executable
45	Normal	winsdk_web.exe	Executable
46	Normal	ClipboardHistory.exe	Executable
47	Normal	MEGAsync.exe	Executable
48	Normal	AnyDesk.exe	Executable
49	Normal	npp.7.6.Installer.exe	Executable
50	Normal	CVHP.exe	Executable
51	Normal	WinSCP-5.13.6-Setup.exe	Executable
52	Normal	coreftplite64.exe	Executable
53	Normal	eagleget_setup.exe	Executable
54	Normal	NetAssemblyInfo.exe	Executable
55	Normal	Morgan Spencer.htm	htm
56	Normal	fdminst-lite.exe	Executable
57	Normal	sigcheck.exe	Executable
58	Normal	RBInternetEncodings500.dll	DLL
59	Normal	cryptolibcps-5.0.43.exe	Executable
60	Normal	Trustlook	PDF
61	Normal	shell.hta	Executable
62	Normal	rufus-usb-3-3.exe	Executable
63	Normal	photosync_setup.exe	Executable
64	Normal	Home Sweet Home 2 - Kitchens and Baths.exe	Executable
65	Normal	ThrottleStop.exe	Executable
66	Normal	Portal.2.incl.upd30-NSIS.exe	Executable
67	Normal	libeay32.dll	Executable
68	Normal	PwDump7.exe	Executable
69	Normal	UaInstall-7.0.6-4.msi	MSI
70	Normal	TP8-2019.exe	Executable
71	Normal	tlscntr.exe	Executable
72	Normal	cccredmgr.exe	Executable
73	Normal	fdminst-lite.exe	Executable

(*continued on next page*)

Table 2 (*continued*)

	Category	Sample name	Type
74	Normal	HoMM3_HD_Latest.exe	Executable
75	Normal	ILSpy.exe	Executable
76	Normal	AnyDesk.exe	Executable
77	Normal	vs_community__1072350829.1545770560.exe	Executable
78	Normal	winsdk_web.exe	Executable
79	Normal	KutoolsforExcelSetup.exe	Executable
80	Normal	acc.exe	Executable
81	Normal	cbhqgi.vbs	VBS
82	Normal	PDFsam_Basic3_3_Installer.exe	Executable
83	Normal	A_info.pdf	PDF
84	Normal	Angry Birds.exe	Executable
85	Normal	aspcmd.msi	MSI
86	Normal	Research_Paper1.pdf	PDF
87	Normal	SupportAssistLauncher.exe	Executable
88	Normal	meda-mp3-joiner-install.exe	Executable
89	Normal	AutoCopyFiles.exe	Executable
90	Normal	soffice.exe	Executable

Table 3
The locations were investigated for digital forensics.

	Digital forensics locations
1	HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls\CustomLocale\en-US
2	HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls
3	HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\SESSION
4	HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control
5	HKEY_LOCAL_MACHINE\SYSTEM
6	HKEY_LOCAL_MACHINE\Software\Microsoft\Rpc
7	HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion
8	HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\
9	HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\
10	HKEY_CURRENT_USER\Software\Microsoft\Windows NT\CurrentVersion\Windows
11	HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Setup
12	HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall
13	HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\
14	HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer
15	Documents and Settings\[user name]\Start Menu\Programs\Startup
16	%systemdrive%\Documents and Settings\[User Name]\Local Settings\Temp
17	%Systemdrive%\Users\victim_user\AppData\
18	%Systemdrive%\Windows\System32
19	%Systemdrive%\Windows\INF\
20	%Systemdrive%\Windows\Globalization\Sorting\sortdefault.nls
21	%Systemdrive%\
22	HKEY_LOCAL_MACHINE\software\policies
23	HKEY_LOCAL_MACHINE\SOFTWARE\Classes\
24	HKEY_CURRENT_USER\Software\Microsoft
25	HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Explorer\Shell Folders
26	HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\explorer\UserShell
27	HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\RunServices
28	HKEY_CLASSES_ROOT\exefile\shell\open\command
29	HKEY_CLASSES_ROOT\comfile\shell\open\command
30	HKEY_LOCAL_MACHINE\Software\CLASSES\batfile\shell\open\command
31	HKEY_LOCAL_MACHINE\Software\CLASSES\exefile\shell\open\command
32	HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Shell
33	HKEY_LOCAL_MACHINE\Software\Microsoft\Active Setup\Installed Components\KeyNameHKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\
34	Advanced\Start_ShowDownloads
35	HKEY_CURRENT_USER\Control Panel\Desktop
36	HKEY_LOCAL_MACHINE \SOFTWARE\Classes\Interface
37	HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\software_name
38	HKEY_LOCAL_MACHINE\SOFTWARE\Policies\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMapKey
39	HKEY_CURRENT_USER\Software\Microsoft\Office\Software_name
40	HKEY_USERS\% account id%\Software\Adobe\
41	HKEY_LOCAL_MACHINE\Software\Classes
42	HKEY_CLASSES_ROOT\software_name
43	HKEY_LOCAL_MACHINE\software\microsoft\windows\currentversion\appmanagement\arpcache\
44	%Systemdrive%\Users\Public\Documents
45	%systemdrive%\Program Files\Software_name\
46	%SYSTEMDRIVE%\Windows\Fonts
47	%Systemdrive%\Users\Public\Documents

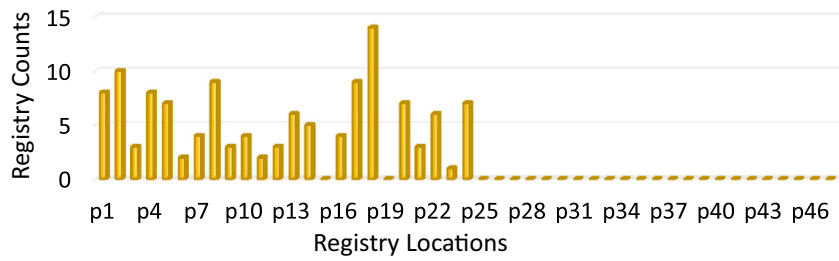


Fig. 7. The impact of Trojan on the registry.

4.1.1. Values for malware

It has been observed that few hives values are of significant importance when the forensic investigator is looking for malicious activities in the system. The modification of P2, P17, P3, P18 and P1 are higher in proportion as compared to other counterpart, although P17, P18 were also present in Bots and Trojan but other keys impact and modification is higher in malware as compared to them.

- P2 (HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls),
- P17 -%Systemdrive%\Users\victim_user\AppData\
- P3 -HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\SESSION
- P18 -%Systemdrive%\Windows\System32
- P1-HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls\CustomLocale\en-US

An analysis of the Fig. 5 and Fig. 9(cleanware) shows that Malware has a slightly different profile in many cases to that of Bots, Trojans, and Worms. For example, Bot has distinctive impacts in the following locations: P1, P2, P8, P13, P17, P18.

4.1.2. Values for Bots

In the case of bots, the Modification of below-mentioned keys are of indicative of bots activities, the detail of these keys are mentioned below

- P18 – %Systemdrive%\Windows\System32
- P8 – HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\
- P19 – %Systemdrive%\Windows\INF\
- P17 – %Systemdrive%\Users\victim_user\AppData\
- P1 – HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls\CustomLocale\en-US
- P2 – HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls
- P4 – HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control

4.1.3. Values for Trojan

It is notable during analysis that modifications of few keys are higher in the Trojan as compared to malware, the details of these are as follows

- P18 – HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer
- P2 – HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls
- P19 – %Systemdrive%\Windows\INF\
- P17 – %Systemdrive%\Users\victim_user\AppData\
- P1 – HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls\CustomLocale\en-US
- P2 – HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control\Nls
- P4 – HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control
- P20 – %Systemdrive%\Windows\Globalization\Sorting\sortdefault.nls

Table 4
Classes for machine learning.

Class	Label
Cleanware	0
Malware	1
Worm	−1
Botnet	−2
Trojan	−3

The analysis of the charts shows that Bot and Trojan have similar values. Also the following locations tend to be higher: P18, P19, and P17, P1 in both bots and Trojan.

4.1.4. Values for Worm

- P18 – HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer
- P17 – %Systemdrive%\Users\victim_user\AppData\
- P8 – HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\
- P13 – HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\
- P4 – HKEY_LOCAL_MACHINE\SYSTEM\ControlSet001\Control
- P21 – %Systemdrive%\
- P20 – %Systemdrive%\Windows\Globalization\Sorting\sortdefault.nls

From the empirical analysis, it has been identified that modification of keys P18 (HKEY_CURRENT_USER\SOFTWARE\Microsoft\Windows\CurrentVersion\Explorer) and P17(-%Systemdrive%\Users\victim_user\AppData\) tends to be higher in all three classes for e.g., Bots, Worms, and Trojans thus we can conclude that these two keys are of great importance for forensic investigator (Fig. 8).

4.1.5. Values for Cleanware

The distinct thing about this research is that forensic investigator will not only able to find the compromised system on the basis of aforementioned values but he will be also able to distinguish between clean systems if below-mentioned keys will be taken in consideration.

- P45 –%systemdrive%\Program Files\Software_name\
- P38 – HKEY_LOCAL_MACHINE\SOFTWARE\
- Policies\Microsoft\Windows\CurrentVersion\Internet Settings\ZoneMapKey
- P37 – HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\
- Windows\CurrentVersion\Uninstall\software_name
- P40 – HKEY_USERS\% account id%\Software\Adobe\
- P42 – HKEY_CLASSES_ROOT\software_name
- P43 – HKEY_LOCAL_MACHINE\software\microsoft\windows\
- currentversion\appmanagement\arpcache\
- P44 – %Systemdrive%\Users\Public\Documents
- P39 – HKEY_CURRENT_USER\Software\Microsoft\Office\Software_name

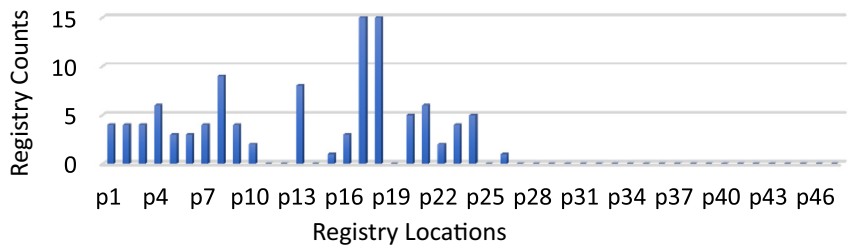


Fig. 8. The impact of Worm on the registry.

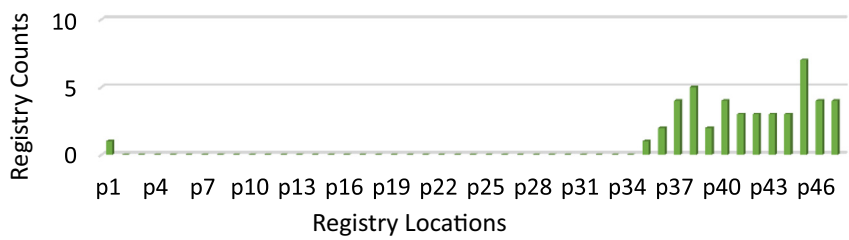


Fig. 9. The impact of Cleanware on the registry.

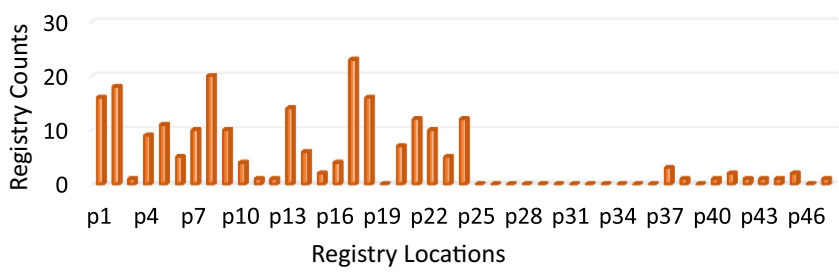


Fig. 10. Impact of malware on the Windows 7 registry.

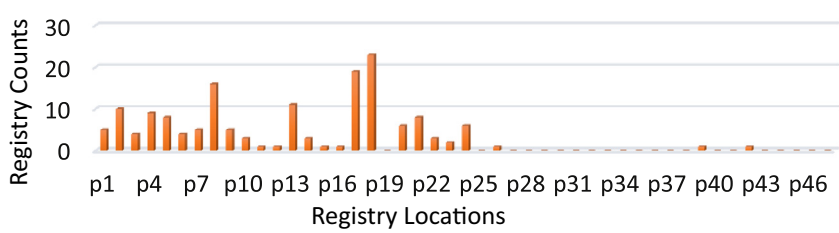


Fig. 11. Impact of malware on the Windows 8 registry.

The modification of these keys will help the forensic investigator to consider system clean instead of malicious without considering the AV scan and test report which will save lots of time as well as resources of the system (Fig. 9).

4.2. Analysis against operating system version

In comparison to the previous two sections where the identification of similarities and differences may assist in helping an investigator in identifying both the presence of malware and the type of payload, the purpose of this comparison is to identify whether any significant differences exist across the last three principal versions of the Windows OS. Notably, as illustrated in Figs. 10–12 the profile exhibited against each OS version is very similar. Going beyond the current state of the art, this study demonstrates that Windows 10 has a very similar impact upon the previously identified 47 registry locations as previous versions.

Upon further examination, there are some small differences that might help identify malware in different versions of the OS.

- Modification to P18 (%Systemdrive%\Windows\System32), P17 (%Systemdrive%\Users\victim_user\AppData\) and P8 (HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\) has significant impact in all three versions of Windows as shown in Figs. 10–12. It has been observed that register values from P35 to P47 are at lowest level in all three versions of Windows.

5. Discussion

From the statistical analysis of the results, the main locations of digital assets from malicious and benign software were identified. Furthermore, in each type of software, in each operating system and in each functionality of malware, the most common locations that create digital evidence were recorded. In future research, this analysis could be extended to other categories of malware (Ransomware, Backdoor, etc.) and other forms of functionality.

Whilst there is value to the investigator in better understanding the impact differing forms of malware have upon the different versions of Windows and in particular how the Registry is effected, it

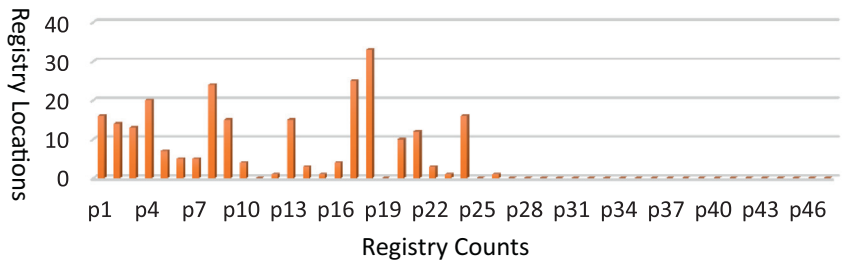


Fig. 12. Impact of malware on the Windows 10 registry.

Table 5
Malware classification performance.

Test ID	Train/Test ratio	Feature tested	Accuracy				
			Logistic regression	Neural net	Decision tree	Random forest	Boosted tree
Test 1	**80/20**	**47**	58%	34%	62%	58%	72%
Test 2	**70/30**	**47**	50%	33.3%	56%	64%	64.9%
Test 3	**60/40**	**47**	58%	33%	68.4%	67.5%	71%
Test 4	**50/50**	**47**	64%	34%	62%	62%	65%

would arguably be more useful if this analysis could be applied in a manner that would provide a proactive approach for investigators to be able to detect and classify the type of malware present within a case without having to rely upon AV. This is not designed to replace or remove AV but to complement the approach, particularly in cases where the malware is not being detected by the AV. To this end, an extended experiment was conducted to determine the degree to which the impact upon the registry and file locations is unique to each family of malware (i.e., is it possible given the impact upon the registry and file locations to determine which family the malware belongs).

The results from Appendix A were used, with the 47 locations forming the features from both malware and benign samples. Then a supervised pattern classification approach was selected because they have stronger reliability than unsupervised approaches and a dataset was easily created from existing malware. The samples from the three families were randomized and split into training and testing datasets – with differing proportions to measure the impact that training data has on the overall performance. In this experiment we have used different machine learning algorithm for classification of cleanware and malware classes. The classes that were used are shown in the table below (Table 4).

During the train, test and validate phases the efficiency and efficacy of the model was measured. Python was utilised and specifically IPython Jupyter notebook v 5.7.2. The Jupyter notebook is an open-source project which is web-based, interactive computing notebook environment which is developed to support data science and scientific computing across the different platform. The first experiment was performed with the label 'Test 1', in which train/test ratio of 80/20 was taken with 47 features, furthermore logic regression, Neural net, Decision tree, Random forest and Boosted tree supervised learning algorithms were utilised and their efficiency and efficacy was measured. It was observed during analysis that Boosted tree algorithm was performing well with 72% accuracy as compared to all other classifiers as depicted in Table 5. Furthermore, investigation was performed by taking training/test ratio of 70/30 to train the model, it had been found that once again Boosted tree outperform all other classifiers with 64% accuracy as shown in Table 5. We extend our experiments by taking training/test ratio to 60/40 to see the impact of accuracy on the classifier, we found that decision tree and Random forest accuracy increases to 68.4% and 67.5% respectively, furthermore boosted tree accuracy decrease from 72% to 71%.

From these experiments, it was observed that the best classification accuracy was produced by the Boosted tree with setting 80/20 as compared to other learning algorithms, moreover it was noticed that the Random forest, Logistic regression, and Decision tree classifiers accuracy increased drastically when we have taken 60/40 ratio but in contrast the accuracy of Boosted tree decrease from 72% to 71% except the neural net whose accuracy remained constant.

The results of this extended experiment demonstrate that modelling the impact that malware has upon the registry and hard disk would be a useful approach to detecting the type of malware family. This type of modelling is far faster than traditional AV software and could be applied either as a standalone tool or integrated into existing computer forensic software as an additional forensic analysis. It also has the advantage over AV tools in that, once trained, it does not need to be continually updated to reflect new signatures (which can be hourly for some tools) – merely periodically updated to reflect the general trends in malware composition. Furthermore, the approach could find applications in host-based intrusion detection systems (HIDS) or intrusion protection systems (HIPS) as well as vulnerability scanners.

6. Conclusion

The paper has undertaken an investigation into the impact that three core types of malware have upon different versions of the Windows OS – specifically targeting the Registry. Whilst previous research has presented the impact of limited volumes of malware upon the Registry, this is the first study to utilize a large volume of malware across the three core types (Bot, Trojan and Worm) along with clean samples. The results from this analysis largely confirm previous studies but provide a greater granularity as to the impact based on the different types of malware. This study has also extended the prior work by including Windows 10 and evidencing that it has overall a similar impact profile on the Registry as previous versions of the software.

The results have shown that it is possible to accelerate a digital forensics analysis through a preliminary analysis of the registry, the modified timestamps and the use of machine learning or deep learning. Targeting these 47 registry locations can provide a first indication to the digital forensics examiner on whether or not malware is present, but also the type across all common versions of the Windows OS.

This type of analysis has several key advantages over existing approaches: it is faster to scan and identify than AV, it is able to detect and classify new malware prior to AV signatures being developed, it does not need frequent updating and can be built into existing tools with applications in both the forensic and security fields.

Conflict of interest

None.

Acknowledgment

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement no. 786698. This work reflects authors' view and Agency is not responsible for any use that may be made of the information it contains.

Appendix A

Malware Dynamic Analysis Locations (1–34) that forensics artifacts have been recorded during dynamic analysis for each of 180 samples (Malware [p1–p34] and clean [p35–p47]) of Worm, Bot and Trojan (Tables A1–A3), Clean (Table A4) and for the three Windows operating systems (Tables A5–A7).

	Worm																										
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**	**P13**	**P14**	**P15**	**P16**	**P17**	**P18**	**P19**	**P20**	**P21**	**P22**	**P23**	**P24**	**P25**	**P26**	**P27-34**
Locations	•																										
	•																										
	•																										
																								•		•	
													•				•										
													•				•	•									
													•				•	•									
		•																•									
										•							•									
										•							•	•								
				•													•										
	•	•		•			•	•					•				•	•									
								•										•		•			•				
																				•							
																	•	•		•							
																	•	•		•							
								•									•	•		•							
				•	•	•	•	•	•				•		•	•	•				•	•		•			
			•	•	•	•	•	•					•			•	•	•			•			•			
			•	•	•	•	•	•					•			•	•	•			•			•			
		•		•	•	•	•	•									•										

	Trojan																									
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**	**P13**	**P14**	**P15**	**P16**	**P17**	**P18**	P19	**P20**	**P21**	**P22**	**P23**	**P24**	P25-34
Locations										•			•				•								
	•									•							•	•		•					
		•							•								•	•		•					
	•									•							•	•			•				
										•							•	•							
		•											•				•	•		•					
				•				•									•	•							
	•	•		•	•	•												•			•	•		•	
				•				•										•							
																•									
	•	•			•		•	•	•				•	•			•			•	•	•	•	•	
	•	•		•	•	•	•	•			•		•	•		•				•		•		•	
	•	•					•	•			•			•			•	•				•		•	
	•	•	•	•				•				•		•		•		•		•		•		•	
	•	•	•	•				•				•		•		•		•		•		•		•	
				•	•			•				•	•					•				•			
			•	•	•		•	•	•				•					•						•	
					•													•							
					•													•							
			•															•							

Bots																									
	P1	**P2**	**P3**	**P4**	**P5**	**P6**	**P7**	**P8**	**P9**	**P10**	**P11**	**P12**	**P13**	**P14**	**P15**	**P16**	**P17**	**P18**	**P19**	**P20**	**P21**	**P22**	**P23**	**P24**	P25-34
Locations							•			•							•	•							
			•					•									•	•							
				•	•												•	•				•			
										•							•	•							
		•															•	•							
																	•	•							
										•							•	•							
		•															•	•							
													•				•	•							
				•													•	•							
					•												•	•							
																	•	•		•					
	•	•					•	•												•				•	
				•				•										•							
					•		•						•				•	•			•	•	•		
	•	•		•	•	•	•	•					•		•					•	•	•		•	
	•							•	•				•				•	•			•			•	
		•						•	•												•			•	
	•	•					•	•					•							•				•	
								•	•									•			•				
	•		•	•				•	•				•					•			•			•	
				•				•										•							
				•				•										•							
		•						•																	

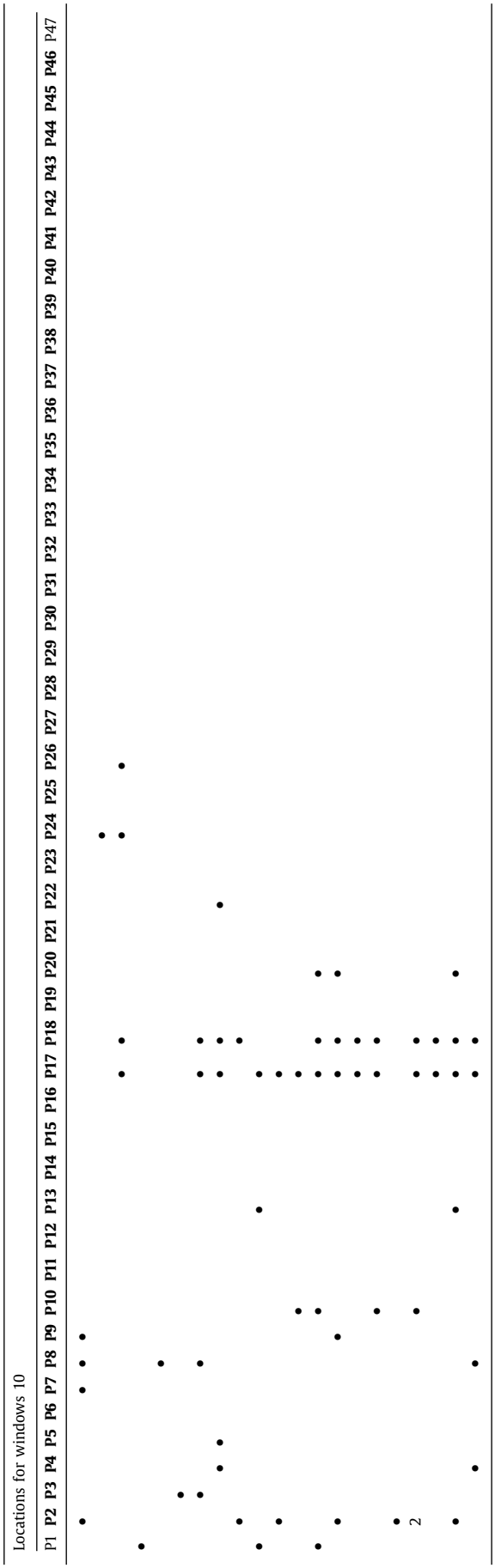
Cleanware														
	P1	**P2-P34**	**P36**	**P37**	**P38**	**P39**	**P40**	**P41**	**P42**	**P43**	**P44**	**P45**	**P46**	P47
Locations	•											•	•	•
							•							
				•										
					•									
												•		
									•					
												•		
												•		
												•		
													•	
														•
					•									
			•											
				•										
					•									
							•							
								•						
										•				
											•			
												•		
													•	
														•
			•											
				•										
					•									
						•								
							•							
								•						
									•					
										•				
											•			
													•	
												•		
														•
													•	
												•		
											•			
										•				
									•					
								•						
							•							
						•								
					•									
				•										

Locations for windows 7

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30	P31	P32	P33	P34	P35	P36	P37	P38	P39	P40	P41	P42	P43	P44	P45	P46	P47
•	•																																													
•																																														
																																				•										
							•																																							
																																								•						
•																																														
																																											•			
												•				•																														
												•				•																														
												•				•	•																													
																•																														
																	•																														
•	•																																													
•																																														
									•							•	•																													
	•															•	•																														
																																				•											
•									•							•	•																													
									•																														•							
																	•																														
	•															•	•																														
												•																																		
												•																																		
			•													•	•																														
															•																															
•	•			•		•	•	•					•			•				•	•	•	•																						
•	•						•		•							•				•			•																							
•			•	•	•		•	•						•		•					•		•																							
	•		•	•			•									•				•																										
							•	•								•																														
•	•		•	•		•	•	•				•	•			•	•		•	•	•		•																							
				•	•	•	•	•				•	•			•	•			•		•	•																							
	•						•																																							
			•	•	•		•	•													•		•																							
•	•		•	•	•	•	•			•		•	•		•						•		•																							
•	•	•	•				•				•		•		•		•		•		•		•																							
				•																																										
																																											•			

Locations for windows 8

P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25	P26	P27	P28	P29	P30	P31	P32	P33	P34	P35	P36	P37	P38	P39	P40	P41	P42	P43	P44	P45	P46	P47
						•			•							•	•																													
																							•		•																					
												•	•																																	
							•					•																																	
												•				•	•																													
									•			•				•																													
																•	•																													
	•																																													
																•	•																													
									•							•																														
			•				•										•																													
																	•																													
•	•		•	•	•	•	•					•		•					•	•	•		•																						
			•				•																																						
•	•			•		•	•					•				•																													
							•										•		•																											
							•									•	•		•																											
		•	•	•	•	•	•								•	•	•			•																										
	•	•					•	•								•	•			•		•																								
							•	•									•			•																										
			•				•										•																													
	•	•														•	•			•	•																										
•	•		•	•	•		•	•								•	•		•	•																										
•	•						•	•				•	•			•	•		•	•																										
			•				•	•								•	•			•			•																							
	•	•															•																													
			•	•	•		•																																						
•	•					•	•			•			•			•	•						•																							
			•	•			•				•	•					•				•																									
				•													•																													



Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.jisa.2019.04.013.

References

[1] Ali M, Shiaeles S, Papadaki M, Ghita BV. Agent-based vs agent-less sandbox for dynamic behavioral analysis. In: Proceedings of the 2018 global information infrastructure and networking symposium (GIIS). IEEE; 2018.

[2] AlienVault (2013). What are the most common types of malware? Available at: https://www.alienvault.com/blogs/security-essentials/what-are-the-most-common-types-of-malware. Accessed: 22 Mar 2019.

[3] Babić D, Reynaud D, Song D. Malware analysis with tree automata inference. In: Computer aided verification. Berlin Heidelberg: Springer; 2011. p. 116–31.

[4] Balthrop J, Forrest S, Newman ME, Williamson MM. Technological networks and the spread of computer viruses. Science 2004;304(5670):527–9.

[5] Bayer U, Comparetti PM, Hlauschek C, Kruegel C, Kirda E. Scalable, behaviour-based malware clustering. In: Proceedings of the NDSS, 9; 2009. p. 8–11.

[6] Bayer U, Habibi I, Balzarotti D, Kirda E, Kruegel C. A view on current malware behaviours. Available at: http://static.usenix.org/event/leet09/tech/full_papers/bayer/bayer_html/. Accessed: 31 Jan 2019.

[7] Bayer U, Kirda E, Kruegel C. Improving the efficiency of dynamic malware analysis. In: Proceedings of the 2010 ACM symposium on applied computing. ACM; 2010. p. 1871–8.

[8] Brumley D, Hartwig C, Liang Z, Newsome J, Song D, Yin H. Automatically identifying trigger-based behaviour in malware. In: Botnet detection. US: Springer; 2008. p. 65–88.

[9] Carrier B. File system forensic analysis. Addison-Wesley Professional; 2005.

[10] Carvey H. Windows forensic analysis DVD toolkit. Syngress; 2009.

[11] Carvey H. Windows registry forensics: advanced digital forensic analysis of the windows registry. Elsevier; 2011.

[12] Carvey H, Altheide C. Tracking USB storage: analysis of windows artifacts generated by USB storage devices. Digit Investig 2005;2(2):94–100.

[13] Casey E, Katz G, Lewthwaite J. Honing digital forensic processes. Digital Investigation 2013;10(2):138–47.

[14] Cert-Eu. Incident response methodology. Cert-Eu; 2012.

[15] Cheng SM, Ao WC, Chen PY, Chen KC. On modeling malware propagation in generalized social networks. IEEE Commun Lett 2011;15(1):25–7.

[16] Christodorescu M, Jha S. Static analysis of executables to detect malicious patterns. Wisconsin University - Madison Department of Computer Sciences; 2006.

[17] Christodorescu M, Jha S, Kruegel C. Mining specifications of malicious behavior. In: Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering. ACM; 2007. p. 5–14.

[18] Christodorescu M, Jha S, Seshia SA, Song D, Bryant RE. Semantics-aware malware detection. In: Proceedings of the IEEE symposium on security and privacy. IEEE; 2005. p. 32–46.

[19] Collie J. The windows IconCache. db: a resource for forensic artifacts from USB connectable devices. Digit Investig 2013;9(3):200–10.

[20] Costa M, Crowcroft J, Castro M, Rowstron A, Zhou L, Zhang L, Barham P. Vigilante: end-to-end containment of internet worms. In: Proceedings of the ACM SIGOPS operating systems review, 39. ACM; 2005. p. 133–47.

[21] Dalziel H. How to defeat advanced malware: new tools for protection and forensics. Syngress; 2014.

[22] Dash P. Getting started with oracle VM virtualbox. PACKT Books; 2013.

[23] Dolan-Gavitt B. Forensic analysis of the Windows registry in memory. Digit Investig 2008;5:S26–32.

[24] Egele M, Scholte T, Kirda E, Kruegel C. A survey on automated dynamic malware-analysis techniques and tools. ACM Comput Surv (CSUR) 2012;44(2):6.

[25] Fnal (2016). Common Windows Trojan/application startup locations. Available at: https://security.fnal.gov/cookbook/WinStartup.html. Accessed: 27 Mar 2016.

[26] Fredrikson M, Jha S, Christodorescu M, Sailer R, Yan X. Synthesizing near-optimal malware specifications from suspicious behaviours. In: Proceedings of the IEEE symposium on security and privacy (SP). IEEE; 2010. p. 45–60.

[27] Gil S, Kott A, Barabási AL. A genetic epidemiology approach to cyber-security. Scientific Report 2014;4:5659.

[28] Gordon S. What is wild?. In: Proceedings of the 20th national information system security conference; 1997.

[29] Greamo C, Ghosh A. Sandboxing and virtualization: modern tools for combating malware. IEEE Secur Priv 2011;9(2):79–82.

[30] Horsman G, Laing C, Vickers P. A case-based reasoning method for locating evidence during digital forensic device triage. Decis Support Syst 2014;61:69–78.

[31] Hosmer C. Python forensics: a workbench for inventing and sharing digital forensic technology. Elsevier; 2014.

[32] James JI, Gladyshev P. A survey of digital forensic investigator decision processes and measurement of decisions based on enhanced preview. Digit Investig 2013;10(2):148–57.

[33] Kolbitsch C, Comparetti PM, Kruegel C, Kirda E, Zhou XY, Wang X. Effective and efficient malware detection at the end host. In: Proceedings of the USENIX security symposium; 2009. p. 351–66.

[34] Kruegel C, Robertson W, Vigna G. Detecting kernel-level rootkits through binary analysis. In: Proceedings of the 20th annual computer security applications conference. IEEE; 2004. p. 91–100.
[35] Li Y, Hui P, Jin D, Su L, Zeng L. Optimal distributed malware defense in mobile networks with heterogeneous devices. IEEE Trans Mob Comput 2014;13(2):377–91.
[36] Lindorfer M, Di Federico A, Maggi F, Comparetti PM, Zanero S. Lines of malicious code: insights into the malicious software industry. In: Proceedings of the 28th annual computer security applications conference. ACM; 2012. p. 349–58.
[37] Lippmann R, Clark A. Recent advances in intrusion detection. Berlin/Heidelberg: Springer; 2008.
[38] Liu W, Liu C, Liu X, Cui S, Huang X. Modeling the spread of malware with the influence of heterogeneous immunization. Appl Math Modell 2016;40(4):3141–52.
[39] Malekal (2016). Available at: http://malwaredb.malekal.com/. Accessed: 10 Apr 2019.
[40] Malicious-streams (2014). Digging for malware: suspicious filesystem geography. Available at: http://www.malicious-streams.com/resources/articles/DGMW1_Suspicious_FS_Geography.html. Accessed: 31 Jan 2016.
[41] Malware.lu (2016). Available at: https://malware.lu/. Accessed: 10 Mar 2019.
[42] MalwareTips (2016). Available at: https://malwaretips.com/. Accessed: 10 Mar 2019.
[43] Malwaretruth (2016). A list of malware types and their definitions. Available at: http://www.malwaretruth.com/the-list-of-malware-types/ Accessed: 22 Mar 2019.
[44] Mee V, Tryfonas T, Sutherland I. The windows registry as a forensic artefact: illustrating evidence collection for Internet usage. Digital Investigation 2006;3(3):166–73.
[45] Mishra BK, Pandey SK. Dynamic model of worm propagation in computer network. Appl Math Modell 2014;38(7):2173–9.
[46] Misra AK, Verma M, Sharma A. Capturing the interplay between malware and anti-malware in a computer network. Appl Math Comput 2014;229:340–9.
[47] Moser A, Kruegel C, Kirda E. Limits of static analysis for malware detection. In: Proceedings of the Twenty-third annual computer security applications conference, ACSAC. IEEE; 2007. p. 421–30.
[48] NetMarketShare.com (2019). Operating system market share. Available at: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=10&qpcustomd=0&qpcustomb. Accessed: 21 Mar 2019.
[49] Norton (2010). Malware removal guide.
[50] Oktavianto D, Muhardianto I. Cuckoo malware analysis. Packt Publishing Ltd.; 2013.
[51] Park Y, Reeves DS, Stamp M. Deriving common malware behaviour through graph clustering. Comput Secur 2013;39:419–30.
[52] Park Y, Zhang Q, Reeves D, Mulukutla V. AntiBot: clustering common semantic patterns for bot detection. In: Proceedings of the IEEE 34th annual computer software and applications conference (COMPSAC). IEEE; 2010. p. 262–72.
[53] Purcell DM, Lang SD. Forensic Artifacts of Microsoft windows Vista system. In: Proceedings of the intelligence and security informatics. Berlin Heidelberg: Springer; 2008. p. 304–19.
[54] RSA (2013). The cyber espionage blueprint.
[55] SANS. Available at: https://www.sans.org/. Last Accessed: 10 Apr 2019.
[56] Schuster A. Pool allocations as an information source in windows memory forensics. In: Proceedings of the IMF; 2006. p. 104–15.
[57] Shanks W. Enhancing incident response through forensic, memory analysis and malware sandboxing techniques. SANS Institute.; 2014. p. 39.
[58] Sharif MI, Lanzi A, Giffin JT, Lee W. Impeding malware analysis using conditional code obfuscation. In: Proceedings of the NDSS; 2008.
[59] Shukla JB, Singh G, Shukla P, Tripathi A. Modeling and analysis of the effects of antivirus software on an infected computer network. Appl Math Comput 2014;227:11–18.
[60] Sikorski M, Honig A. Practical malware analysis: the hands-on guide to dissecting malicious software. No Starch Press; 2012.
[61] Stormo, J.M. (2013). Analysis of Windows 8 registry artifacts.
[62] Symantec (2009). Common loading points for viruses, worms, and Trojan horse programs. Available at: https://support.symantec.com/en_US/article.TECH99331.html. Accessed: 2 Feb 2019.
[63] SNDBOX (2019). Available at: https://app.sndbox.com/upload/ (Accessed: 23 January 2019).
[64] Thomas S, Sherly KK, Dija S. Extraction of memory forensic artifacts from windows 7 ram image. In: Proceedings of the IEEE conference on information & communication technologies (ICT). IEEE; 2013. p. 937–42.
[65] Veracode (2012). Common malware types: cybersecurity 101. Available at: https://www.veracode.com/blog/2012/10/common-malware-types-cybersecurity-101. Accessed: 22 May 2019.
[66] Vespignani A. Complex networks: behind enemy lines. Nat Phys 2005;1(3):135–6.
[67] VirusSign (2016). Available at: http://www.virussign.com/. Accessed: 10 Apr 2019.
[68] VirusTotal (nd). VirusTotal – free online virus, malware and URL scanner. Available at: https://www.virustotal.com/. Accessed: 25 Mar 2019.
[69] Vlachos V, Ilioudis C, Papanikolaou A. On the evolution of malware species. In: Proceedings of the global security, safety and sustainability & e-democracy. Berlin Heidelberg: Springer; 2012. p. 54–61.
[70] Vmware (2016). VMware virtualization for desktop & server, application, Public Hybrid Clouds. Available at: http://www.vmware.com/. Accessed: 22 Mar 2019.
[71] VxHeaven (2016). Available at: http://vxheaven.org/. Accessed: 10 Apr 2016.
[72] Willems C, Holz T, Freiling F. Toward automated dynamic malware analysis using cwsandbox. IEEE Security Privacy 2007;5(2):32–9.
[73] Zeltser L. (nd). Available at: https://zeltser.com/. Accessed: 10 Mar 2019.
[74] Zou CC, Towsley D, Gong W. Modeling and simulation study of the propagation and defense of internet e-mail worms. IEEE Trans Depend Secure Comput 2007;4(2):105–18.