Windows Registry Harnesser for Incident Response and Digital Forensic Analysis

Avinash Singh
asingh@cs.up.ac.za
Department of Computer Science
University of Pretoria
South Africa

Adeyemi R. Ikuesan
aikuesan@cs.up.ac.za
Department of Computer Science
University of Pretoria
South Africa

Hein S. Venter
hventer@cs.up.ac.za
Department of Computer Science
University of Pretoria
South Africa

Abstract: The extraction of digital evidence from storage media is a growing concern in digital forensics, due to the time and space complexity in acquiring, preserving and analysing digital evidence. Microsoft Windows Registry is an example of a potential source of digital evidence in a Windows computer that contains a database of evidential information about both the system and users. However, due to the vastness of the Registry, it is difficult to manually sift through this database to extract potential evidence. Furthermore, manually sifting through the database provides room for human error, which could invalidate the entire forensic investigation. This time-consuming and error-prone process can cause several delays in processing and presenting criminal cases during litigation. The need for an automated extraction and analysis process of digital evidence is therefore inherently needed. The aim of this research is to develop an automated forensically sound process for Windows Registry investigation. This entails setting up strict and reliable measures for an investigator to follow whilst attempting to make this process as automated as possible by minimizing human interaction. Consequently, this ensures the integrity and authenticity of the data. To achieve this, both an acquisition tool and an analysis tool was developed. A comparative analysis of the developed tool to existing tools showed increased performance with respect to time and admissibility.

Keywords: Windows forensics, Windows Registry analysis, Digital forensic investigation, Semi-automated analysis, Incident response.

1. Introduction

Security attacks and digital crimes are increasingly prevalent in the modern digital age [1]–[3]. Attackers use sophisticated techniques to exploit security vulnerabilities, leaving companies and organizations helpless and vulnerable. Cyber-crimes are continuously rising due to the increase in risks [4]. Cybercrime can be mitigated through deterrence by attempting to uncover criminal footprints from potentially incriminating evidence found in electronic media [5]. These footprints can be found in the Windows Registry, where large amounts of data regarding several aspects of the host computer are stored. Finding evidence and performing analysis on Windows Registry is a tedious and time-consuming process. This complexity can be attributed to the manual process used by a Digital Forensic Investigator (DFI) to search for evidence within the Registry. During a digital forensic investigation, strict processes are followed. If a step is omitted, the possible alteration of evidence can prevent the eventual of the evidence in a court of law [6]. Time for evidence collection, analysis and the complexity of extracting evidence from the cybercrime scene, are just some of the fundamental factors that inhibit the effectiveness of a forensic investigation [7]. Therefore, there is a need to transcribe the

traditional evidence collection process from a manual time-consuming process to an automated process, characterized by a reduced likelihood of error and higher reliability and integrity of the digital evidence.

In a digital-related crime incident, a DFI is required to identify and extract potential digital evidence from the host machine; a process generally referred to as Incident Response [8]. An incident response procedure involves the method of evidence identification, collection, and preservation for the digital investigation. The evidence collection process follows a forensically acceptable method, which ensures the integrity and the availability of the potential digital evidence. Upon completion of the acquisition/collection process, a method of evidence preservation and documented chain-of-custody is adhered to, in order to conform to the standardized procedure for evidence admissibility [9]. Similar conformity is required in the analysis process, where chain-of-evidence is also required.

The Windows Registry is a constantly growing repository on the host machine, with a significant growth in size over time. This need for an increase in storage capacity and the manual-searching process has created a difficult platform for a DFI to scavenge for evidence during an investigation, consequently, increasing investigation time [8]. Furthermore, such a manual process is vulnerable to the error of omission and commission, while subjecting the investigator to the complexity of data analysis.

Based on these observed limitations, this study attempts to develop *an automated Windows Registry collection and analysis tool*, which can extract and analyze digital evidence in a forensically sound manner, to aid digital forensic analysis and incident response. The next section discusses the related work.

2. Related Work

The Windows Registry is a hierarchical database, which is comprised of several settings, drivers, services, interfaces and configurations about the system [10], [11]. This information is stored as keys which are structured indices within the Registry, that provide easier access to the associated values of the key [11]. These values contain complex representations of information about the keys. The structure does not provide any inbuilt mechanism for anomaly detection and/or criminal activity detection. Since the Registry is well structured, a systematic process can be developed to extract information from potentially known keys that could contain significant evidentiary value and weight. However, these keys may differ from one machine to another, depending on factors such as the duration of use, volume of user activity, number of software installed, Operating System (OS) architecture, OS versions, as well as the connected networks. Existing tools can extract some of the information mentioned above by providing a DFI with the utility to manually uncover incriminating evidence. There are several keys within Registry and it is not feasible to manually perform an exhaustive search because some keys are reliant on the system usage. Furthermore, allowing the DFI to manually sift through the Windows Registry can illicitly cause an error, which could lead to evidence inadmissibility.

Values can be represented in diverse data-formats such as hexadecimal, little endian, big endian and symbolic links [12]. These formats make the Registry more robust for complex storage and easier access while ensuring a degree of security measures. It also enhances effective data storage capabilities [13][14]. Observations from [11] state that storage media has shown effective growth in storage capacity over the years. Several studies have been conducted on the Windows XP Registry, as well as on Windows 7 Registry [4][6][16][17]. In these studies, Windows-XP and Windows-7 were covered in significant detail with respect to identifying potential critical evidence and uncovering the keys associated with it. Typically, the Windows Registry contains five hierarchical hives also known as Root key functions. An overview

description of these root key functions is shown in Table 1. The locations of these hive files can be in two locations one for the system (*C:\Windows\System32\config*) and the other for the user (*C:\Users\{user}*).

Several models have been developed, as identified [12], to detect modifications to the Windows Registry for more advanced crime scenarios. These advanced crimes take place when criminals attempt to erase digital fingerprints and invalidate some of the keys stored in the Registry. In order to detect advanced criminal activity, consistency models have been created to verify or detect inconsistencies within the Registry [12]. Some examples of consistency models include the time checking, system clock adjustment, Registry hive modification, and Registry information modification.

There have been a few studies on Windows Registry analysis [11], [18]–[20] which attempts to extract potential evidential from the Registry keys, while others found new techniques that can be used to detect anomalies [18]·[20]. A study in [11] proposed several categories of Registry analysis namely; system, application, network, device and user history. These categories explore many Registry keys and provide meaningful information that is useful as a source of potential digital evidence (PDE).

It is generally agreed [15]·[11]·[12] that further studies are required to better understand the Windows Registry. This assumes that it is almost impractical to assume that events extracted from the Registry represent a comprehensive content of potential digital evidence that can be discovered, due to the constantly growing size. This study, therefore, attempts to develop an analysis tool for the Windows Registry which can aid in the discovery of existing Registry keys that could potentially contain an evidential value for investigative processes. Such a tool should be capable of pulling evidence on any version of Windows, and decrease the process-time of an investigation, whilst ensuring harnessing capabilities.

Table 1. Root keys within the Windows Registry

Hive/Root Key	Description
HKEY_CLASSES_ROOT	Information about software and user interfaces
HKEY_CURRENT_USER	Configuration information for the current logged in user
HKEY_LOCAL_MACHINE	The machine hardware-specific information
HKEY_USERS	Configuration information for the all the users
HKEY_CURRENT_CONFIG	Current system configuration

Table 2. Longitudinal view of Windows Registry over the different versions of the Operating System

Hive Files	Windows 95	Windows 98	Windows 2000	Windows XP	Windows VISTA	Windows 7	Windows 8	Windows 10
BCD	-	-	-	-	✓	✓	✓	✓
DRIVERS	-	-	-	-	-	-	-	✓
SAM	-	-	✓	✓	✓	✓	✓	✓
SECURITY	-	-	✓	✓	✓	✓	✓	✓
SOFTWARE	-	-	✓	✓	✓	✓	✓	✓
SYSTEM	-	-	✓	✓	✓	✓	✓	✓
DEFAULT	-	-	✓	✓	✓	✓	✓	✓
COMPONENTS	-	-	-	-	✓	✓	✓	✓
NTUSER.DAT	-	-	✓	✓*	✓*	✓*	✓*	✓*
USRCLASS.DAT	-	-	-	✓*	✓*	✓*	✓*	✓*
SYSTEM.DAT	✓	✓	-	-	-	-	-	-
USER.DAT	✓	✓	-	-	-	-	-	-

Policy.pol	-	✓

Items marked with a (*) may have more than one location for the file

3. Experimental Methodology

The operational framework used for this study is shown in Figure 1. The framework starts with the acquisition of live data using a developed Registry data acquisition tool (RegAcquire[1]). The acquisition process follows two types of data acquisition to ensure that the data acquired is verifiable and comprehensive. All processes that the acquisition process follows is logged to enable the verification of the acquisition process and the authenticity of the acquired data dumps. These data dumps are stored on the acquisition drive as input data to the subsequent analysis process. The analysis process is carried out by a developed tool for a Registry data analysis (RegSmart). This tool imports the acquired dumps from the RegAcquire tool verifies and validates the forensic soundness [21] in terms of the integrity of the obtained dump. The RegSmart tool further processes the data and loads it into objects for easier data manipulation and data analysis. After the data has been processed, several analysis processes can be run on the data to extract potential evidence, followed by customizable report generation. The report process also verifies the dumps prior to report generation, to mitigate any external alterations and modifications performed on the imported dumps.

3.1 Data-Collection Tool Development (RegAcquire)

The RegAcquire tool utilizes a Batch script, which was compiled to an executable for portability purposes. The script makes use of built-in commands and third-party tools which include *reg export* [22]*, RawCopy* [23] and *FCIV* [24]. *reg export* [22] was used to retrieve the textual contents of the Registry while RawCopy [23] is used to obtain the physical hive files from the machine. Since these files are constantly in use, Volume Shadow Copy Service (VSS) was used for the extraction process. FCIV [24] is a simple file checksum integrity verifier. It was used to obtain the MD5 hashes of each file that was acquired. In order to achieve a level of optimization, abstraction and better management, a specific folder structure was created and is explained in the next section.

3.1.1 File structure collection mechanism

A specific file structure was chosen in order to achieve consistency and easier verification as shown in Figure 2. This structure comprises a root folder using the nomenclature (the logged in users username, the machine name and the date of the acquisition). The date may be inaccurate due to different machine times and time zone, however, the other attributes help identify the dump due to its uniqueness. The extracted physical hive files and the additional files generated by the RegAcquire tool are saved as a different file extension, to prevent a DFI to mistakenly load the file into the Registry at the crime scene or forensic lab through a double-click action.

3.1.2 Acquisition integrity mechanism

The hash digest of each file is stored in the file called *hash.hash.* All files in the dumps are marked as read-only, with the hash file having more restrictions by using Windows built-in icacls [25] to restrict access to read-only as well protecting the file from unauthorized deletion. This is done to ensure the integrity of the dumps.

3.1.3 Data integrity assurance mechanism

Once the data dumps have been extracted the tool creates a forensic copy and verifies the copy by comparing each digest to the digest of the original file. Since the tool is required to be automated with minimal modification to the host system, it preserves the state of the machine

[1] Reference redacted for blind review

in an incident response scenario by implementing the acquisition process on a removal device (a USB stick for instance). This process was tested by acquiring two dumps from the same machine at a ten-minute time interval (whilst the system is idle). The digest of both dumps was compared and revealed different digests. This is due to Registry constantly being in use, even at a small time interval, the digests may be different. This depends on the version of the operating system and the current state of the system. Windows 7 and later versions write to the log files and only periodically write to Registry in order to prevent overhead.

3.1.4 Automated integrity verification mechanism

The acquisition process operates an autorun configuration file that automates the acquisition tool once the removable device is connected to the system. This feature has been removed from Windows 8 and later. Therefore, an investigator would need to execute the tool. However, the tool remains automated and no user interaction is required to retrieve dumps from a given machine once the tool has been executed.

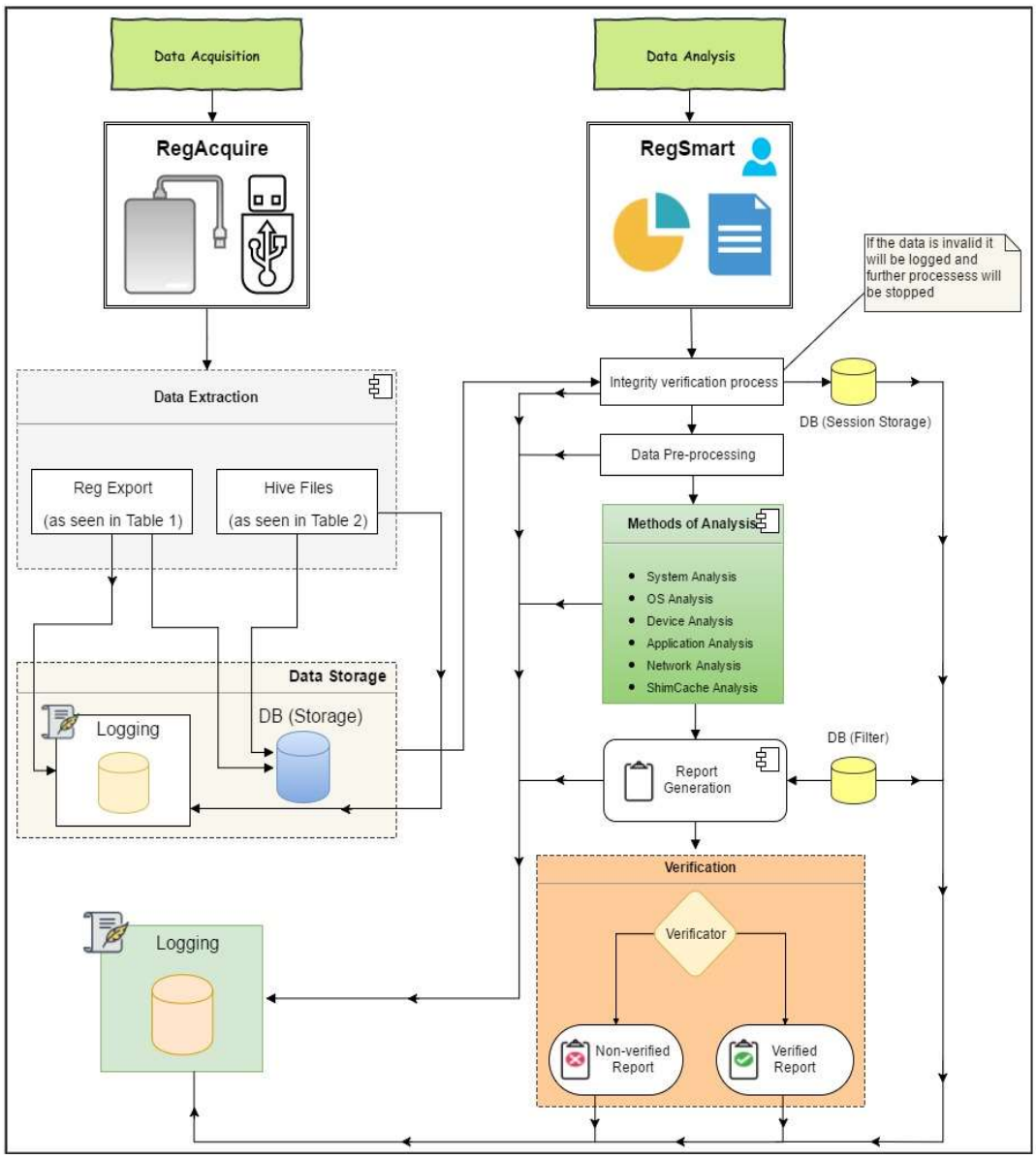

Figure 1. The operational framework for Registry acquisition and analysis in incident response.

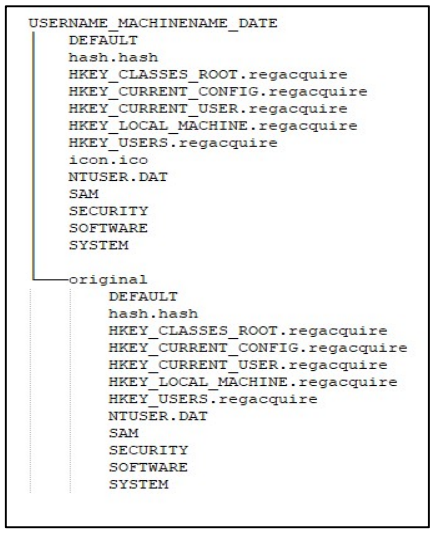

Figure 2. RegAcquire folder structure.

3.2 Data-Analysis Tool Development (RegSmart)

The RegSmart tool was developed using Python-3, due to the robustness, utility, support and cross-platform capabilities of the programming language. The Registry dumps obtained from the RegAcquire tool feeds as input to the RegSmart tool. This tool performs several types of analysis based on the different aspects in which potential evidence can be found in the Registry. This tool aimed to harness the vast contents and the agile architecture of the Windows Registry, by providing support for all Windows NT systems.

3.2.1 Integrity verification process

The RegSmart tool verifies each data dump that is imported from the RegAcquire tool by checking the logs and hash file to validate and verify that the dumps are valid and unaltered. The authenticity and verification process of the data dumps are achieved by taking the hash digests of the original files and matching them to the digest that is stored in the hash file. The analysis is permitted only if the authenticity and verification process is successful. Otherwise, a report on the failed authenticity is enumerated. In addition, the tool logs each activity performed, to further corroborate the claim of integrity during data analysis.

3.2.1 Data pre-processing

The RegSmart tool makes use of a Python library called *python-registry* [26]. The *python-registry* converts hive files into Registry objects which enables faster data access, consequently, ensure faster processing time and faster analysis.

3.2.2 Methods of analysis

The methods of data analysis are categorized into different analysis types such as system, OS, application, network, device, shim cache and user. These categories were chosen because they relate to the type of information that is stored in Registry as well as the evidence an investigator would find. User analysis was not considered in this study because it entails a significant amount of key searching and scanning through user-based applications within the Registry to uncover any other evidence. The mapping of events and logs which can further add context to what is found in Registry and can be useful in user behavioural profiling. However, such would be situation specific. It is therefore not considered in the current study. The types of analysis the RegSmart tool addressed is explained in the proceeding subsections.

System Analysis: contains the system configurations and settings defined by the user. This information may reveal traces of uninstalled applications. This type of analysis extracts the

computer name, operating system version, the last shutdown time, the process architecture and core count. Analysis of running services is also contained in the system analysis. The tool automatically checks which control set was used and extracts information from the keys accordingly. The source locations for system analysis is presented in Table 3.

OS Analysis: is centred on the operating system installed on the machine as well as the users and operating system specific information. The OS information available in the Registry include product name, release id, current build, product id, path name, installed date (the date the OS was installed or upgraded), organization information (name of the organization to which the OS is licensed), owner name, SID, and the path to the ntuser.dat file. An example of the OS analysis information source is presented in Table 4.

Application Analysis: involves the analysis of the installed applications on the machine for all users as well as the current user. Such analysis includes the startup applications, registered and installed applications. Start-up applications are the applications that are automatically executed when the operating system loads once the computer is powered on. Registered applications are the applications that are mostly installed through official providers such as Microsoft installers (with .msi file extension). Examples of the application analysis information sources are presented in Table 5.

Network Analysis: involves the analysis of the network cards and network connections (wired/wireless) of the machine. The information contained in the Registry include network cards, intranet, and wireless connections. Other information includes the name of the network, date of first connection, date of last connection, as well as the unique identifier. An example of the source of the information contained in the network analysis is presented in Table 6.

Device Analysis: are the details of drivers and physical devices plugged into the machine. The information contained in the Registry include the printer drivers and USB information, as well as other devices that could be connected to the system. USB information includes the type of device, vendor, product name, revision number, serial, installed date, last plugged in date and unplugged date. An example of the source of such information is shown in Table 7.

Shim Cache Analysis: are the details related to cache information stored in the Registry regarding Windows application compatibility and Windows apps. This module was developed by Mandiant [27] and was incorporated into RegSmart tool as is without any modifications. This aids the analysis process by trying to retrieve passwords and cached information pertaining to the activity of the apps. An example of the source of such information is shown in Table 8.

Table 3. Keys used for System Analysis

HKLM\\SYSTEM\\ControlSet001\\Control\\Session Manager\\Environment
HKLM\\SYSTEM\\ControlSet001\\Control\\ComputerName\\ComputerName
HKLM\\SYSTEM\\ControlSet001\\Control\\Windows
HKLM\\SYSTEM\\ControlSet001\\Services

Table 4. Keys used for OS Analysis

HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion
HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\ProfileList
HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\ProfileList\\{SID}
HKLM\\SAM\\Domains\\Account\\Users\\Names

Table 5. Keys used for Application Analysis

HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run
HKLM\\SOFTWARE\\RegisteredApplications
HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall
HCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run
HCU\\Software\\RegisteredApplications
HCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Uninstall
HKLM\\SOFTWARE\\WOW6432Node\\Microsoft\\Windows\\CurrentVersion\\Uninstall
HKLM\\SOFTWARE\\WOW6432Node\\Microsoft\\Windows\\CurrentVersion\\Run

Table 6. Keys used for Network Analysis

HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\NetworkCards
HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\NetworkList\\Nla\\Cache\\Intranet
HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\NetworkList\\Nla\\Wireless
HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\NetworkList\\Profiles

Table 7. Keys used for Device Analysis

HKLM\\SYSTEM\\ControlSet001\\Control\\Print\\Environments
HKLM\\SYSTEM\\ControlSet001\\Enum\\USBSTOR

Table 8. Keys used for Shim Cache Analysis

HKLM\\SYSTEM\\CurrentControlSet\\Control\\Session Manager\\AppCompatibility\\ AppCompatCache
HKLM\\SYSTEM\\CurrentControlSet\\Control\\Session Manager\\AppCompatCache\\ AppCompatCache

4. Testing

In this section, the RegAcquire and RegSmart tools were tested with respect to speed and capabilities. The testing environments utilized different configurations/versions of Windows-based machines as shown in Table 9 and 10. The tests were based on several different cases and the results revealed the efficiency of the process of the tools developed. Both the developed tools support 32-bit and 64-bit OS architecture.

4.1 Registry Acquisition

In Table 9, the use of different OS setups and configurations revealed the robustness and agility of the acquisition tool. The size of the dumps has a directly proportional relationship to the time taken and read/write speeds to perform the acquisition. From this, the study observed that the time-taken is relatively fast thus capable of enhancing the acquisition time.

4.2 Registry Analysis and Reporting

The result of the analysis process is presented in Table 10. The analysis of the acquired Registry dumps was performed on a machine with the following configurations: 16GB of RAM; 480GB SSD, Windows 10 OS, 64-bit architecture, core i7 2720QM (2.20 GHz) processor. From Table 10, it was observed that the time taken to import and validate dumps are relatively shorter which further shows the agility and time efficiency on the analysis time and reporting process.

Table 9. Test results for acquisition

Test Environment	Architecture	Processor Count	Time	System Configuration	Size of Dumps (Copy & Original)
Windows XP VM	64-bit	2	**00:00:46**	RAM: 1 GB; SSD: 10 GB	167.01 MB
Windows 7	64-bit	8	**00:01:56**	RAM: 8 GB; HDD: 250 GB	719.30 MB
Windows 8.1	32-bit	8	**00:02.54**	RAM: 8 GB; HDD: 500 GB	1.68 GB
Windows 10	64-bit	8	**00:01:54**	RAM: 16 GB; SSD: 480 GB	997.87 MB

Table 10. Test Results for Analysis and Reporting

Test Environment	Import dumps Time (mm:ss)	Validating Dumps Time (mm:ss)	Report Generation Time (mm:ss)
Windows XP VM	00:06	00:01	00:08
Windows 7	00:22	00:03	00:11
Windows 8.1	00:45	00:09	00:19
Windows 10	00:30	00:05	00:14

5. Comparative Analysis of Various Tools

In order to evaluate the effectiveness of existing tools to the needs of a DFI, this section explores the capabilities of existing tools with regards to the time it will take a DFI to perform an investigative procedure utilizing these tools. Experimental case-based observations are used to achieve this objective.

5.1 Experimental Scenarios

Three distinct experimental scenarios are presented as case studies, followed by an investigative process, which involved Registry acquisition and analysis by the respective tools. The following tools were selected for the analysis of Windows Registry, based on their degree of functionality: RegRipper [28], Registry Decoder [29], and Registry Explorer [30]. The analysis process involves the loading of the Registry hives followed by the analysis of the loaded hives in each tool. The first scenario involves the utilization of these tools to find any evidence pertaining to a removable device investigation. The second and third scenarios involved data/policy breach through illegal network access, and illicit website access/download, respectively.

a) Scenario 1

"After the preparation of the final examination paper, University Professor Bob went out of his office for lunch, without switching off or locking his computer. While the computer was still running, one of his colleagues, Alice, who works in the same office as Bob, grabbed the opportunity to obtain a copy of the final examination paper using a USB removable device. Alice sold the examination paper for a large amount of money. The news of the leaked-examination was reported to the University management, and Bob was placed on suspension. Bob maintained his innocence throughout the interrogation with the University authorities, causing the case to be handed over to the Police. All electronic components in Bob's office were transferred to the forensic unit of the Police. The forensic team eventually suspected that another device, which was not included in the list of the seized electronic devices, was used on the computer between the exam preparation and examination-delivery date. Unfortunately for

Alice, the removable device which she used, was configured with "Alice" as the device name, which matched the serial number of the device registered to her. Bob was eventually acquitted while Alice was prosecuted for the crime."

Investigative Procedure: A Registry dump was carried out on Bob's computer using the RegSmart tool, while analysis was done using the identified Registry tools. Since Bob claimed his innocence, the investigation was directed at removable devices that may have been plugged into the computer without his knowledge. To do this, the Registry analysis focused on the aspect of the database that relates to removable devices. Important information that relates to the removable device which can positively identify the device used is the serial number, vendor name, product name, revision number, installed date, last plugged in date and last unplugged date. A Summary of the outcome of the tool explorations is presented in Table 11. The following Registry keys contain the location of evidence:

1. ***HKLM\\SYSTEM\\ControlSet001\\Enum\\USBSTOR***
2. ***HKLM\\SYSTEM\\ControlSet001\\Enum\\USBSTOR\\{Device}***
3. ***HKLM\\SYSTEM\\ControlSet001\\Enum\\USBSTOR\\{Device}\\{Serial}\\Properties\\{ID}\\0064***
4. ***HKLM\\SYSTEM\\ControlSet001\\Enum\\USBSTOR\\{Device}\\{Serial}\\Properties\\{ID}\\0066***
5. ***HKLM\\SYSTEM\\ControlSet001\\Enum\\USBSTOR\\{Device}\\{Serial}\\Properties\\{ID}\\0067***

Discussion

RegSmart found all the information within seconds. It would be significantly faster, if not instant if the secure process of validating and verifying the integrity of the data dumps before performing analysis was not considered in each scenario. RegRipper provides all these details in a long string e.g. "Disk&Ven_SanDisk&Prod_Cruzer_Blade&Rev_1.27" as found in the Registry key which is not reader-friendly, it does not provide more details about the USB itself with relation to timelining. Registry Decoder finds most information quickly but could not retrieve all the information regarding USB history and timelining. Registry Explorer is a basic Registry viewer which takes time to load, the reason the time is faster is that the path to the evidence is already known and is just a matter of navigation. The USB information is not deep within the huge storage structure allowing navigation to the evidence faster as opposed to the other possible scenarios.

b) Scenario 2

"A data-centric company discovered a data breach on their system. The company does not have many visitors and are not allowed to take any electronic devices into the data warehouse. One-day Garry visited the data company with malicious intent and snuck his laptop through security. The data company is situated in an old building and makes use of wireless communications such as Wi-Fi (since running cables through the building may weaken its structure). Garry knew that with accessing the Wi-Fi he will get unrestricted access to the servers and mainframes where all the data is captured. Garry then stole a few gigabytes of data which contained personal information. In time, the security team discovered unauthorized access through the network logs. The security head confronted management, who then alerted the Police causing an investigation to occur. During the investigation, the security cameras were checked, narrowing down the footage to the period of the unauthorized access from the logs. After positively identifying Garry on the security footage his electronic devices were seized making him the primary suspect."

Investigative Procedure: Since the only access to the network was through the wireless network, the investigation was directed at network connections that may have been plugged into Gary's computer or connected wirelessly. To do this, the Registry analysis focused on the aspect of the database that relates to wireless connections. This includes important information that relates to the wireless networks which can positively identify the device used is the companies SSID, date of connection, and date of last connection. The dates provide the period

of when the crime was committed and then use to further corroborate the evidence through the surveillance cameras. A Summary of the outcome of the comparative process is presented in Table 12. The following Registry keys contain the location of evidence:

1. ***HKLM\\SOFTWARE\\Microsoft\\ Windows NT\\CurrentVersion\\NetworkList\\Nla\\Cache \\Intranet***
2. ***HKLM\\SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\NetworkList\\Profiles***

Discussion

An investigator can successfully use RegSmart to analyse the Registry dump in half the time required by the other existing tools Consequently, aid the effectiveness of the investigation process. RegRipper takes more time than all the other tools thus can be considered least effective in this scenario. However, RegRipper could find the Gateway to the network. Network gateways were not considered in the development of RegSmart. Future works will consider the integration of such capability. Registry Decoder and Registry Explorer were able to extract the basic information from the Registry dump.

Table 11. Results for Scenario 1

	Features								
Tools	Automated process	Serial Number	Vendor name	Product	Revision Number	Installed Date	Plugged Date	Unplugged Date	Time Taken to Retrieve Evidence
RegSmart	✓	✓	✓	✓	✓	✓	✓	✓	00:00:10
RegRipper	✓	✓	✓	✓	✓	×	×	×	00:00:38
Registry Decoder	✓	✓	✓	✓	×	✓	×	×	00:00:21
Registry Explorer	×	✓	✓	✓	✓	✓	✓	✓	00:00:23

Table 12. Results for Scenario 2

	Features						
Tools	Automated	ID	Gateway	Name (SSID)	Date Created	Date Last Connected	Time Taken to Retrieve Evidence
RegSmart	✓	✓	×	✓	✓	✓	00:00:08
RegRipper	✓	×	✓	✓	✓	✓	00:01:09
Registry Decoder	✓	×	×	✓	✓	✓	00:00:52
Registry Explorer	×	✓	✓	✓	✓	✓	00:00:53

c) Scenario 3

"A software house provides all its employees with laptops. These laptops belong to the software house and a strict policy is followed which legally binds the employees. Employees are not allowed to use these laptops for anything illegal (any action not acceptable to the organization) or install any applications without formal approval. Sarah, an employee, wanted to download illegal content from torrents such as movies and music. Despite knowing that peer-to-peer networks are against organization policy, she installed µTorrent (a peer-to-peer client) whilst at work. She eventually downloaded a huge number of movies. Due to a large amount of data transferring to Sarah's computer, an alert was created due to the high bandwidth usage. A

technician found out that the high usage was coming from Sarah's computer on peer-to-peer ports, which he was suspicious about and informed the owner. The owner confronted Sarah, but she denied doing such. An investigation was therefore conducted. Sarah's company laptop was seized. The forensic team performed concrete analysis and Sarah was found guilty and prosecuted."

Investigative Procedure: Since the technician identified the large bandwidth usage to be a program, the investigation was directed at software applications and cache. To do this, the Registry analysis focused on the aspect of the database that relates to applications installed and cache information. Important information that relates to the applications which can positively identify the program used is the name of the application and any cache information A Summary of the outcome of the tool explorations is presented in Table 13. The following Registry keys contain the location of evidence:

1. ***HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Run***
2. ***HKLM\\SOFTWARE\\RegisteredApplications***
3. ***HKLM\\SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall***
4. ***HCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Run***
5. ***HCU\\Software\\RegisteredApplications***
6. ***HCU\\Software\\Microsoft\\Windows\\CurrentVersion\\Uninstall***
7. ***HKLM\\SOFTWARE\\WOW6432Node\\Microsoft\\Windows\\CurrentVersion\\Uninstall***
8. ***HKLM\\SOFTWARE\\WOW6432Node\\Microsoft\\Windows\\CurrentVersion\\Run***
9. ***HKLM\\SYSTEM\\CurrentControlSet\\Control\\Session Manager\\AppCompatibility\\ AppCompatCache***
10. ***HKLM\\SYSTEM\\CurrentControlSet\\Control\\Session Manager\\AppCompatCache\\ AppCompatCache***

Discussion

RegSmart could find the necessary information to help identify probable cause and some potential digital evidence. Furthermore, it could find traces of µTorrent in the cache as well as all installed and start-up application within 12-seconds. However, the time required by both RegRipper and Registry decoder is approximately triple the time required by RegSmart. In addition, RegRipper was able to find only traces of potential evidence in the cache through a plugin. Registry decoder, on the other hand, shows a list of installed applications in which µTorrent was listed. The fourth tool, Registry Explorer, required manual searching and since the keys were known, the search was fast. However, since many keys are involved, it becomes a time-consuming process relative to the other tools. From the results above, RegSmart shows significant time improvement, whilst also providing a platform for minimal human interaction. It can also retrieve significant evidential information which can be useful in successful corroboration during the investigation. Further analysis of the effectiveness of RegSmart through timeline analysis is presented in the next section.

Table 13. Results for Scenario 3

	Features						
Tools	Automated	Cache Information	Registry Traces	Installed Application	Startup Applications	Registered Applications	Time Taken to Retrieve Evidence
RegSmart	✓	✓	✓	✓	✓	✓	00:00:12
RegRipper	✓	✓	×	×	×	×	00:00:34
Registry Decoder	✓	×	×	✓	×	×	00:00:32
Registry Explorer	×	✓	✓	✓	✓	✓	00:02:02

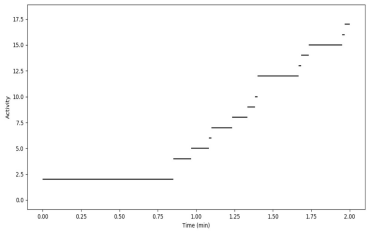

Figure 3a: Scenario 1

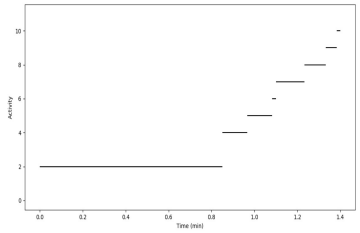

Figure 3b: Scenario 2

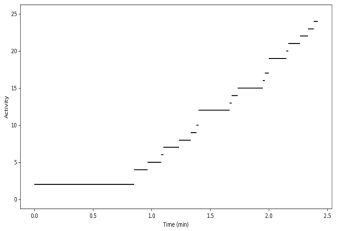

Figure 3c: Scenario 3

d) Investigation Timeline

The investigation timeline for each scenario is displayed in Figure 3. The USB scenario as shown in Figure 3a, took approximately 50-seconds for an investigator to find the device with 2-minutes being the total time taken to conduct the investigation. This resulted in 17-activities between the investigator and the system. The Wi-Fi scenario as presented in Figure 3b, took approximately 48-seconds for an investigator to find the Wi-Fi network with 1-minute 24-seconds being the total time taken to conduct the investigation. Consequently, the process resulted in 10-activities between the investigator and the system. Finally, the Application scenario as shown in Figure 3c, took approximately 52-seconds for an investigator to find the application with 2-minutes 18-seconds being the total time taken to conduct the investigation.

5.2 Analysis of Common Tools

A comparative analysis of the functionality of the developed tools to other existing tools is further shown in Table 14. The selected tools include Forensic Toolkit (FTK v6.2.1) [31], Autopsy (v4.4.1) [32], OS Forensics (v5.1.1003) [33], EnCase (v8.05) [34] and KUSTAR [11]. These tools were selected because they are the commonly used tools in digital investigations, particularly on Registry analysis.

Table 14. Comparative Analysis of Common Tools

	Tools							
Features	**FTK**	**Autopsy**	**En-Case**	**OS Forensics**	**Registry Decoder**	**KUSTAR**	**Reg-Ripper**	**RegSmart & RegAcquire**
Open source	×	✓	×	×	✓	×	✓	✓
License Not Required	×	✓	×	×	✓	-	✓	✓
MD5 hashing	✓	✓	✓	✓	×	×	×	✓
Windows Support	✓	✓	✓	✓	✓	✓	✓	✓
User Friendly	✓	✓	✓	✓	×	✓	×	✓
GUI	✓	✓	✓	✓	✓	✓	×	✓
Reliability	✓	✓	✓	✓	✓	✓	✓	✓
Documentation	✓	✓	✓	✓	×	✓	×	✓
Reporting	✓	✓	✓	✓	✓	×	×	✓
Examiner Log	✓	✓	✓	✓	×	×	✓	✓
Shim Cache Analysis	×	×	✓	×	✓	×	✓	✓
Timeline Analysis	×	✓	✓	✓	✓	×	✓	✓
Manual Searching	✓	×	✓	✓	✓	×	×	✓
Effective for Windows Registry	×	✓	×	×	✓	✓	✓	✓

6. Future Work

A number of aspects of the Windows Registry remain to be more comprehensively explored in future work. Probable areas to consider include user profiling for user attribution, Registry and RAM mapping techniques, the development of a Registry readiness mechanism for recovering deleted keys, methods to rate the effectiveness of Registry analysis tools through their harnessing capabilities, as well the extension of Registry analysis to other operating systems. The Registry and RAM mapping is currently a major research challenge, for which an effective investigation of malware forensic techniques needs to be developed. These research areas could assist in addressing malware attacks such as ransomware and could lead to the development of a more comprehensive Registry forensics process.

7. Conclusion

The observation from this study supports the assertion that the Windows Registry is a continuously-growing repository, which contains several pieces of potential digital evidence. Existing tools are not capable of swift and secure extraction and analysis of Windows Registry. The tool developed in this study attempted to address this limitation and thus improving investigation durations and admissibility. This capability can be attributed to the integration of robust forensic processes at the data acquisition and analysis phases. The tools developed also have backward capabilities, such that only one tool is needed to perform acquisition and/or analysis. This research extends the body of literature on digital forensics and tools required for digital investigations, based on evidence within the Windows Registry, with an emphasis on the speed of analysis processes and the assurance of the forensic checks required during litigation.

8. References

[1] E. E. Kenneally and C. L. T. Brown, "Risk sensitive digital evidence collection," *Digit. Investig.*, vol. 2, no. 2, pp. 101–119, 2005.

[2] J. Bergeron, M. Debbabi, J. Desharnais, M. M. Erhioui, Y. Lavoie, N. Tawbi, J. Bergeron, M. Debbabi, J. Desharnais, M. Erhioui, Y. Lavoie, and N. Tawbi, "Static Detection of Malicious Code in Executable Programs," *Int. J. Req. Eng*, pp. 184–189, 2001.

[3] W. L. Bennett, "Changing Citizenship in the Digital Age," *Civ. life online Learn. how Digit. media can Engag. youth*, vol., pp. 1–24, 2008.

[4] C. G. Austin, *Personal connections in the digital age, second edition*, vol. 20, no. 3. John Wiley & Sons, 2017.

[5] P. Hyman, "Cybercrime," *Commun. ACM*, vol. 56, no. 3, p. 18, 2013.

[6] H. Alavi and T. Khamichonak, "EU and US export control regimes for dual use goods: An overview of existing frameworks," *Rom. J. Eur. Aff.*, vol. 17, no. 1, pp. 59–74, 2017.

[7] K. Fisher, Y. Mandelbaum, and D. Walker, "The next 700 data description languages," *J. ACM*, vol. 57, no. 2, pp. 1–51, 2010.

[8] B. Carrier and E. H. Spafford, "Getting physical with the digital investigation process," *Int. J. Digit. Evid.*, vol. 2, no. 2, pp. 1–20, 2003.

[9] R. Accorsi, "Safekeeping digital evidence with secure logging protocols: State of the art and challenges," *IMF 2009 - 5th Int. Conf. IT Secur. Incid. Manag. IT Forensics - Conf. Proc.*, no. 1, pp. 94–110, 2009.

[10] H. Carvey, "The Windows Registry as a forensic resource," *Digit. Investig.*, vol. 2, no. 3, pp. 201–205, 2005.

[11] K. Alghafli, A. Jones, and T. Martin, "Forensic Analysis of the Windows 7 Registry," *J. Digit. Forensics, Secur. Law*, vol. 5, no. 4, pp. 5–30, 2010.

[12] Y. Zhu, J. James, and P. Gladyshev, "A consistency study of the windows registry," in *IFIP Advances in Information and Communication Technology*, vol. 337 AICT, K.-P. Chow and S. Shenoi, Eds. Berlin,

Heidelberg: Springer Berlin Heidelberg, 2010, pp. 77–90.

[13] P. P. Aung and T. M. Naing, "A Novel Secure Combination Technique of Steganography and Cryptography," *Int. J. Inf. Technol. Model. Comput.*, vol. 2, no. 1, pp. 55–62, 2014.

[14] J. Hendricks, R. R. Sambasivan, S. Sinnamohideen, and G. R. Ganger, "Improving small file performance in object-based storage," 2006.

[15] B. Dolan-Gavitt, "Forensic analysis of the Windows registry in memory," *Digital Investigation*, vol. 5, no. SUPPL. 2008.

[16] H. S. Lallie and P. J. Briggs, "Windows 7 registry forensic evidence created by three popular BitTorrent clients," *Digit. Investig.*, vol. 7, no. 3–4, pp. 127–134, 2011.

[17] L. W. Wong, "Forensic Analysis of the Windows Registry | Forensic Focus - Articles," *Digit. Forensics Artic. Res. Pap.*, 2011.

[18] T. D. Morgan, "Recovering deleted data from the Windows registry," *Digit. Investig.*, vol. 5, no. SUPPL., 2008.

[19] A. Case, "Recovering and Analyzing Deleted Registry Files," *Digital Forensics Solutions*. .

[20] S. J. Stolfo, F. Apap, E. Eskin, K. Heller, S. Hershkop, A. Honig, and K. Svore, "A comparative evaluation of two algorithms for Windows Registry Anomaly Detection," *J. Comput. Secur.*, vol. 13, no. 4, pp. 659–693, 2005.

[21] S. Omeleze and H. S. Venter, "Proof of concept of the online neighbourhood watch system," in *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 2016, vol. 171, pp. 78–93.

[22] Microsoft, "Reg Export," 2016. [Online]. Available: https://technet.microsoft.com/en-us/library/cc742017(v=ws.11).aspx. [Accessed: 19-Feb-2017].

[23] Joakim Schicht, "RawCopy," 2017. [Online]. Available: https://github.com/jschicht/RawCopy/releases. [Accessed: 20-Mar-2017].

[24] Microsoft, "FCIV," 2017. [Online]. Available: https://support.microsoft.com/en-za/help/841290/availability-and-description-of-the-file-checksum-integrity-verifier-u. [Accessed: 20-Apr-2017].

[25] Microsoft, "Icacls," 2016. [Online]. Available: https://technet.microsoft.com/en-us/library/cc753525(v=ws.11).aspx. [Accessed: 02-May-2017].

[26] W. Ballenthin, "python-registry," 2015. [Online]. Available: https://github.com/williballenthin/python-registry. [Accessed: 26-Apr-2017].

[27] MANDIANT, "ShimCacheParser," 2017. [Online]. Available: https://github.com/mandiant/ShimCacheParser. [Accessed: 02-Sep-2017].

[28] H. Carvey, "RegRipper," 2017. [Online]. Available: https://github.com/keydet89/RegRipper2.8. [Accessed: 12-Aug-2017].

[29] L. Marziale and A. Case, "Registry Decoder," 2009. [Online]. Available: http://code.google.com/p/registrydecoder/. [Accessed: 13-Jan-2018].

[30] EricZimmerman, "Registry Explorer," 2016. [Online]. Available: https://ericzimmerman.github.io. [Accessed: 13-Jan-2018].

[31] AccessData, "FTK Imager," *AccessData*, 2016. [Online]. Available: http://accessdata.com/product-download/digital-forensics/ftk-imager-version-3.2.0. [Accessed: 25-Aug-2017].

[32] Sleuthkit, "Autopsy," 2017. [Online]. Available: https://www.sleuthkit.org/autopsy. [Accessed: 26-Jun-2017].

[33] PassMarkSoftware, "OS Forensics," 2017. [Online]. Available: https://www.osforensics.com/download.html. [Accessed: 15-Nov-2017].

[34] OpenText, "EnCase," 2017. [Online]. Available: https://www.guidancesoftware.com/encase-forensic. [Accessed: 06-Aug-2017].