

## Windows registry harnesser for incident response and digital forensic analysis

Avinash Singh, Hein S. Venter & Adeyemi R. Ikuesan

**To cite this article:** Avinash Singh, Hein S. Venter & Adeyemi R. Ikuesan (2020) Windows registry harnesser for incident response and digital forensic analysis, Australian Journal of Forensic Sciences, 52:3, 337-353, DOI: [10.1080/00450618.2018.1551421](https://doi.org/10.1080/00450618.2018.1551421)

**To link to this article:** <https://doi.org/10.1080/00450618.2018.1551421>



Published online: 18 Dec 2018.



Submit your article to this journal [↗](#)



Article views: 778



View related articles [↗](#)



View Crossmark data [↗](#)



Citing articles: 5 View citing articles [↗](#)



# Windows registry harnesser for incident response and digital forensic analysis

Avinash Singh , Hein S. Venter and Adeyemi R. Ikuesan

Department of Computer Science, University of Pretoria, 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 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 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 minimizing human interaction through automation. To achieve this, an acquisition and analysis tool was developed. A comparative analysis of the developed tool to existing tools showed increased performance with respect to time and forensic soundness.

## ARTICLE HISTORY

Received 25 May 2018

Accepted 12 November 2018

## 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.<sup>1–3</sup> 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.<sup>4</sup> Cybercrimes can be mitigated through deterrence by attempting to uncover criminal footprints from potentially incriminating evidence found in electronic media.<sup>5</sup> 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 strict manual process used by a Digital Forensic Investigator (DFI) to search for evidence within the Registry during a digital forensic investigation. If a step is omitted, evidence can be altered, compromising the presented evidence in a court

of law.<sup>6</sup> Time for evidence collection, analysis and the complexity of extracting evidence from a cybercrime scene, are just some of the fundamental factors that inhibit the effectiveness of a forensic investigation.<sup>7</sup> Consequently, 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.<sup>8</sup> 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.<sup>9</sup> 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, increasing investigation time.<sup>8</sup> Furthermore, this 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 analyse digital evidence in a forensically sound manner, to aid digital forensic analysis and incident response.

## 2. Related work

The Windows Registry is a hierarchical database, which is comprised of several settings, drivers, services, interfaces and configurations about the system.<sup>10,11</sup> This information is stored as keys that are structured indices within the Registry, which provide easy access to the associated values of the key.<sup>11</sup> 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 piece of software installed, Operating System (OS) architecture, OS versions, as well as the connected networks. Existing tools can extract some of the above-mentioned information, 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.<sup>12</sup> 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.<sup>13,14</sup> Observations from Alghafli et al.<sup>11</sup> states that storage media has shown effective growth in storage capacity over the years. Several studies have been conducted on Windows XP and Windows 7 Registry with respect to identifying potential critical evidence and uncovering the keys associated with it.<sup>11,12,15</sup> Typically, the Windows Registry contains five hierarchical hives, 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<sup>12</sup>, 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.<sup>12</sup> 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<sup>11,16–18</sup> which attempt to extract potential evidential from the Registry keys, while other studies found new techniques that can be used to detect anomalies.<sup>16,18</sup> A study by Alghafli et al.<sup>11</sup> 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). Windows Registry has grown over the years with new files being added over time. A high-level view of the changes of the Registry hive files over the different versions of Windows can be seen in Table 2.

**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

It is generally agreed<sup>10–12,19</sup> that further studies are required to better understand the Windows Registry. This is because it is 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 will 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 off any version of Windows, and decreasing the process-time of an investigation.

### 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 (<https://avinashsingh786.github.io/RegSmart/#regacquire>). The acquisition process follows two types of data acquisition to ensure that the data acquired are verifiable and comprehensive. All processes in the acquisition phase are logged to enable the verification of the acquisition phase 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, which is carried out by the developed tool for Registry analysis – RegSmart (<https://avinashsingh786.github.io/RegSmart/>). This tool imports the acquired dumps from the RegAcquire tool, verifies and validates the forensic soundness<sup>20</sup> in terms of the integrity of the obtained dump. The RegSmart tool further processes the data and loads the data into objects for easier manipulation and analysis. After the data have been processed, several analysis processes can be run on the data to extract potential evidence, followed by customizable reports. 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 into an executable (exe) file for portability purposes. The script makes use of built-in commands and third-party tools which include reg export<sup>21</sup>, Raw Copy<sup>22</sup> and FCIV.<sup>23</sup> The reg export<sup>21</sup> was used to retrieve the textual contents of the Registry while RawCopy<sup>22</sup> is used to obtain the physical hive files from the machine. Since these files are constantly in use, Volume Shadow Copy Service (VSS)<sup>24</sup> was used for the extraction process. FCIV<sup>23</sup> is a simple file checksum integrity verifier, which was used to obtain the MD5 hashes of each acquired file. In order to achieve a level of optimization, abstraction and better management, a specific folder structure, as explained in the next section, was created.

##### 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 user's username, the machine name and the date of the acquisition). 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 mistakenly loading the file into the Registry at the crime scene or forensic lab through a double-click action.



```

USERNAME_MACHINENAME_DATE
|
|  DEFAULT
|  hash.hash
|  HKEY_CLASSES_ROOT.regacquire
|  HKEY_CURRENT_CONFIG.regacquire
|  HKEY_CURRENT_USER.regacquire
|  HKEY_LOCAL_MACHINE.regacquire
|  HKEY_USERS.regacquire
|  icon.ico
|  NTUSER.DAT
|  SAM
|  SECURITY
|  SOFTWARE
|  SYSTEM
|
|-----original
|
|  DEFAULT
|  hash.hash
|  HKEY_CLASSES_ROOT.regacquire
|  HKEY_CURRENT_CONFIG.regacquire
|  HKEY_CURRENT_USER.regacquire
|  HKEY_LOCAL_MACHINE.regacquire
|  HKEY_USERS.regacquire
|  NTUSER.DAT
|  SAM
|  SECURITY
|  SOFTWARE
|  SYSTEM

```

**Figure 2.** RegAcquire folder structure.

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 the 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.

## **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 feed 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 content 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 the data dump 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.

### 3.2.2. Data pre-processing

The RegSmart tool makes use of a Python library called *python-registry*<sup>26</sup> which converts hive files into Registry objects, enabling faster data access, and consequently ensuring faster processing time and analysis.

### 3.2.3. Method 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 can further add context to what is found in Registry and can be useful in user behavioural profiling. However, such would be case-specific and is thus not considered in this current study. The types of analysis the RegSmart tool addressed is explained in the subsequent 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



**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

computer is powered on. Registered applications are 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:** provides the details of drivers and physical devices plugged into the machine. The information contained in the Registry includes 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 number, 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:** covers the details related to cache information stored in the Registry regarding Windows application compatibility and Windows apps. This module was developed by Mandiant<sup>27</sup> and was incorporated into the 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](#).

## 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

**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

Windows-based machines as shown in [Tables 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 the read/write speeds to perform the acquisition. From this, it was observed that the time taken is relatively fast, thus is 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: 16 GB of

**Table 9.** Test results for acquisition.

Test environment	Architecture	Processor count	Time	System configuration	Size of dumps (copy and 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

RAM; 480 GB 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 short, which further shows the agility and time efficiency on the analysis time and reporting process.

## 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,<sup>28</sup> Registry Decoder<sup>29</sup> and Registry Explorer.<sup>30</sup> 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.

#### 5.1.1. 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 devices 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.** 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



worth 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 and which can positively identify the device used is the company's SSID, date of connection, and date of last connection. The dates provide the period when the crime was committed and are then used 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*

5.3. Discussion

An investigator can successfully use RegSmart to analyse the Registry dump in half the time required by the other existing tools, aiding the effectiveness of the investigation process. RegRipper takes more time than all the other tools and can, therefore, be considered least effective in this scenario. Regripper does find the Gateway, but RegSmart can't because it was not in scope. 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.

5.3.1. 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

Table 12. Results for Scenario 2.

Tools	Features						Time taken to retrieve evidence
	Automated	ID	Gateway	Name (SSID)	Date created	Date last connected	
RegSmart	✓	✓	✗	✓	✓	✓	00:00:08
RegRipper	✓	✗	✓	✓	✓	✓	00:01:09
Registry Decoder	✓	✗	✗	✓	✓	✓	00:00:52
Registry Explorer	✗	✓	✓	✓	✓	✓	00:00:53

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 several movies. Owing to a large amount of data being transferred 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 company owner. The owner confronted Sarah, but she denied doing this. 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\

**Table 13.** Results for Scenario 3.

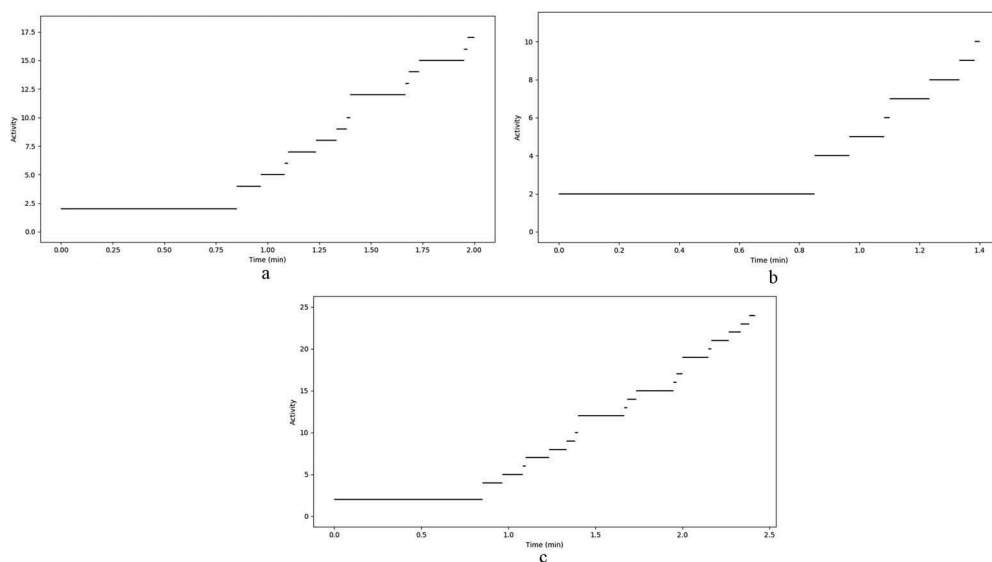
Tools	Features						Time taken to retrieve evidence
	Automated	Cache information	Registry traces	Installed application	Startup applications	Registered applications	
RegSmart	✓	✓	✓	✓	✓	✓	00:00:12
RegRipper	✓	✓	×	×	×	×	00:00:34
Registry Decoder	✓	×	×	✓	×	×	00:00:32
Registry Explorer	×	✓	✓	✓	✓	✓	00:02:02

## 5.4. Discussion

RegSmart could find the necessary information to help identify probable cause and some potential digital evidence. Furthermore, it could find traces of  $\mu$ Torrent in the cache, as well as all installed and start-up applications 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  $\mu$ Torrent 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.

### 5.4.1. Investigation timeline

The investigation timeline for each scenario is displayed in Figure 3. The USB scenario, as shown in Figure 3(a), took approximately 50 s 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 3(b), took approximately 48 s for an investigator to find the Wi-Fi network, with 1 min 24 s 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 3(c), took approximately 52 s for an investigator to find the application, with 2 min 18 s being the total time taken to conduct the investigation.



**Figure 3.** (a) Scenario 1. (b) Scenario 2. (c) Scenario 3.

**Table 14.** Comparative analysis of common tools.

Features	Tools							
	FTK	Autopsy	EnCase	OS for- ensics	Registry decoder	KUSTAR	RegRipper	RegSmart and 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	×	✓	×	×	✓	✓	✓	✓

### 5.5. Analysis of common tools

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

## 6. Future work

A number of aspects of the Windows Registry remain to be 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 a malware forensic technique 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 improve 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.

## Disclosure statement

No potential conflict of interest was reported by the authors.

## ORCID

Avinash Singh  <http://orcid.org/0000-0003-3024-4076>

Hein S. Venter  <http://orcid.org/0000-0002-3607-8630>

Adeyemi R. Ikuesan  <http://orcid.org/0000-0001-7355-2314>

## References

1. Kenneally EE, Brown CLT. Risk sensitive digital evidence collection. *Digit Investig.* 2005;2(2):101–119. doi:10.1016/j.diin.2005.02.001
2. Bergeron J, Debbabi M, Desharnais J, Erhioui MM, Lavoie Y, Tawbi N, et al. Static detection of malicious code in executable programs. *Citeseer.* 2001;184–189. Accessed 2017 Apr 27. Available from: <https://pdfs.semanticscholar.org/a09b/8f7bbf24f80e037b06578fe6308babfca1a8.pdf>.
3. Bennett WL. Changing citizenship in the digital age. John D Catherine T MacArthur Found Ser Digit Media Learn. 2008;1–24. doi:10.1162/dmal.9780262524827.001
4. Austin CG. *Personal connections in the digital age*. 2nd ed. Vol. 20, Consumption Markets & Culture. John Wiley & Sons;2017;293–296. doi:10.1080/10253866.2015.1135538
5. Hyman P. Cybercrime. *Commun ACM.* 2013;56(3):18. doi:10.1145/2428556.2428563
6. Alavi H, Khamichonak T. EU and US export control regimes for dual use goods: an overview of existing frameworks. *Rom J Eur Aff.* 2017;17(1):59–74. doi:10.3366/ajicl.2011.0005
7. Fisher K, Mandelbaum Y, Walker D. The next 700 data description languages. *J Acm.* 2010;57(2):1–51. doi:10.1145/1667053.1667059
8. Carrier B, Spafford EH. Getting physical with the digital investigation process. *Int J Digit Evid.* 2003;2(2):1–20.
9. Accorsi R. 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.* 2009;(1):94–110. doi:10.1109/IMF.2009.18
10. Carvey H. The windows registry as a forensic resource. *Digit Investig.* 2005;2(3):201–205. doi:10.1016/j.diin.2005.07.003
11. Alghafli K, Jones A, Martin T. Forensic analysis of the Windows 7 registry. *J Digit Forensics, Secur Law.* 2010;5(4):5–30. doi:10.15394/jdfsl.2010.1081
12. Zhu Y, James J, Gladyshev P. A consistency study of the windows registry. In: Chow K-P, Sheno S, editors. *IFIP advances in information and communication technology*. Vol 337 AICT. {IFIP} Advances in Information and Communication Technology. Berlin (Heidelberg): Springer Berlin Heidelberg; 2010:77–90. doi:10.1007/978-3-642-15506-2\_6
13. Aung PP, Naing TM. A novel secure combination technique of steganography and cryptography. *Int J Inf Technol Model Comput.* 2014;2(1):55–62.
14. Hendricks J, Sambasivan RR, Sinnamohideen S, Ganger GR. Improving small file performance in object-based storage. *Citeseer.* 2006. Accessed 2017 Jun 21. Available from: <http://www.pdl.cmu.edu/PDL-FTP/Storage/CMU-PDL-06-104.pdf>.

15. Lallie HS, Briggs PJ. Windows 7 registry forensic evidence created by three popular BitTorrent clients. *Digit Investig.* 2011;7(3–4):127–134. doi:10.1016/j.diin.2010.10.002
16. Morgan TD. Recovering deleted data from the Windows registry. *Digit Investig.* 2008;5 (SUPPL.). doi:10.1016/j.diin.2008.05.002
17. Case A. Recovering and Analyzing Deleted Registry Files. *Digit Forensics Solutions*. 2011. Accessed 2017 Jul 11. Available from: <http://dfir.org/research/recovering-and-analyzing-deleted-registry-hives.pdf>.
18. Stolfo SJ, Apap F, Eskin E, Heller K, Hershkop S, Honig A, Svore K. A comparative evaluation of two algorithms for Windows Registry Anomaly Detection. *J Comput Secur.* 2005;13 (4):659–693. doi:10.3233/JCS-2005-13403
19. Dolan-Gavitt B. Forensic analysis of the Windows registry in memory. *Digit Investig.* 2008;5 (SUPPL.):1–17. doi:10.1016/j.diin.2008.05.003
20. Omeleze S, Venter HS. Proof of concept of the online neighbourhood watch system. In: Glitho R, Zennaro M, Belqasmi F, Agueh M, editors. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, Vol. 171. Cham: Springer International Publishing; 2016:78–93. doi:10.1007/978-3-319-43696-8\_9
21. Microsoft. Reg Export. Published 2016. Accessed 2017 Feb 19. Available from: [https://technet.microsoft.com/en-us/library/cc742017\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc742017(v=ws.11).aspx).
22. Joakim Schicht. RawCopy. Published 2017. Accessed 2017 Mar 20. Available from: <https://github.com/jschicht/RawCopy/releases>.
23. Microsoft. FCIV. Accessed 2017 April 20. Available from: <https://support.microsoft.com/en-za/help/841290/availability-and-description-of-the-file-checksum-integrity-verifier-u>.
24. Russinovich ME, Solomon DA, Ionescu A. Volume shadow copy service. In: *Windows Internals*, 5th ed. O'Reilly; 2009:688–698. Accessed 2017 May 2. Available from: <https://www.oreilly.com/library/view/windows-internals-fifth/9780735625303/ch08s05.html>.
25. Microsoft. Icacls. Published 2016. Accessed 2017 May 2. Available from: [https://technet.microsoft.com/en-us/library/cc753525\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc753525(v=ws.11).aspx).
26. Ballenthin W. python-registry. Published 2015. Accessed 2017 Apr 26. Available from: <https://github.com/williballenthin/python-registry>.
27. MANDIANT. ShimCacheParser. Published 2017. Accessed 2017 Sep 2. Available from: <https://github.com/mandiant/ShimCacheParser>.
28. Carvey H. RegRipper. Published 2017. Accessed 2017 Aug 12. Available from: <https://github.com/keydet89/RegRipper2.8>.
29. Marziale L, Case A Registry Decoder. Published 2009. Accessed 2018 Jan 13. Available from: <http://code.google.com/p/registrydecoder/>.
30. EricZimmerman. Registry Explorer. Published 2016. Accessed 2018 Jan 13. Available from: <https://ericzimmerman.github.io>.
31. AccessData. FTK Imager. AccessData. Published 2016. Accessed 2017 Aug 25. Available from: <http://accessdata.com/product-download/digital-forensics/ftk-imager-version-3.2.0>.
32. Sleuthkit. Autopsy. Published 2017. Accessed 2017 June 26. Available from: <https://www.sleuthkit.org/autopsy>.
33. PassMarkSoftware. OS Forensics. Published 2017. Accessed 2017 Nov 15. Available from: <https://www.osforensics.com/download.html>.
34. OpenText. EnCase. Published 2017. Accessed 2017 Aug 6. Available from: <https://www.guidancesoftware.com/encase-forensic>.