Forensic Tools for Analyzing Windows Registry

Jozef Mihale
Faculty of Informatics and Information Technologies
Slovak University of Technology
Bratislava, Slovakia
xmihale@stuba.sk

***Abstract*—The Windows Registry is an integral part of Windows operating systems, and with the current number of Windows operating systems on the market, it is important that forensic analysts understand the Windows Registry and know how to extract as much useful information from this component as possible. This document will try to explain at least the basics of this issue and allow the reader to look into the world of Windows Registry.**

***Index Terms*—Digital forensics, forensic analysis, Windows Registry, forensic tools, comparison of forensic tools, practical lab**

I. Basic terms

In this part we will take a closer look at terms like digital forensics, forensic analysis and forensic process. To each of the three words there will be a short explanation followed by more detailed description of each word.

A. *Digital forensics*

Digital forensics or digital forensic science is a branch of forensic science that focuses on the recovery and investigation of material found in digital devices related to cybercrime. This adopted explanation from the following link is telling us, that the digital forensics is a science, which belongs to the larger whole of forensic science and the primary focus or goal of digital forensics is investigate, analyze the inputs, produce outputs (e.g. search for evidence) and hand over all the data, that had been gathered (mainly the evidence) or present them to the court. Of course, all the gathered digital evidence must be indisputable. Therefore, all evidence must be marked in the chain of custody and with the help of hash functions we can ensure that the evidence or data will not lose their integrity. And this is not everything, but more about this topic will be closer examined in the part *Forensic process*.

You are maybe wondering why the digital forensics even exists. And the simple answer is - ”Someone must be responsible for the security incident.” It would be best for all this responsibility not to remain with the security manager but to look for a real attacker, who caused this security incident in the first place. And for all this the digital forensics will come in handy.

The second question that probably comes to your mind is, how we can track down the attacker back, if he left us only compromised machine (e.g. laptop or computer). To answer this, we will have to look a few years back. Specifically a statement from Edmond Locard - ”Any action of an individual, and obviously the violent action constituting a crime, cannot occur without leaving a trace.” This is also true in the digital environment. But on second thoughts it is not always true, because like in the real world, attackers can clean up after themselves, what could be harder in digital enviroment (maybe even deemed impossible). For now, let’s say that every action on computer or other machine leaves behind a set of traces and these traces make the digital forensics possible.

If we want to summarize this part, we can do so with following description: ”The high-level process of digital forensics includes the acquisition of data from a source, analysis of the data and extraction of evidence, and preservation and presentation of the evidence” [2]. In the following part we will take a closer look at the described process.

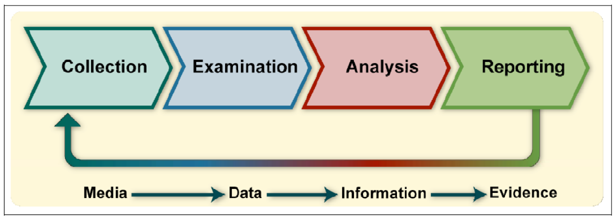


Fig. 1. Forensic process model [4]

B. *Forensic process*

Forensic process is defined as a list of steps, which will be used against the compromised computer in order to solve a possible cybercrime. There are many lists on how to perform forensics but we will take a closer look at the four-phase model, which can be seen at figure 1.

This model consists of 4 phases/parts:

- **Collection** - ”The first step in the forensic process is to identify potential sources of data and acquire data from them” [4]. That is why this phase of the process is also called data gathering. The primary goal of this phase is to acquire data from compromised machine (e.g. laptop or computer). If we also take a closer look at the figure 1 we can see, that we work with media e.g. HDD, SSD or USB stick and mainly we want to create a digital copy of the compromised media. This bit-by-bit exact copy is called an image and it can be acquired with help of write-blocker and some software of choice. Write-blocker is a device, which ensures, that the original data will not be the least altered. After the image has been taken, it is very convenient to calculate the hash value of the acquired image. Analyst can for example use SHA-256 and the calculated hash value has to be also stored somewhere (more about hash functions/algorithms can be found on following link). There are also potential pitfalls in this phase, because the data we work with are fragile, in the sense that they can be changed very quickly and that is why ”A procedure for evidence acquisition and preservation should be simple, rapid and effective, saving time and money. The complexity of the environment (e.g. OS or files ystem) however, demands that you define the details ahead of time” [9].

- **Examination** - After the gathering phase is the data in the so-called raw form, so they have to be somehow processed. Another important part of this phase is, that the accumulated data can reach high capacity and the analyst only needs to select the data, which are relevant for the next step - analysis. ”For example, yesterday's firewall log might hold millions of records, but only five of the records might be related to the event of interest” [4]. Therefore the goal of this phase is to clearly identify which data will be further investigated and which will be set aside for the time being.
- **Analysis** - At the given phase, data should be further studied and analyzed to achieve goal of forensic analysis and that is the production of output or a document with findings and evidence from given data. The analysis can include thing like user identification, file system analysis, file analysis or determining the timeline of the interesting events, that may have occurred during the security incident and finding out if those events were related to the same security incident. ”Often, this effort will include correlating data among multiple sources” [4]. For example, it is advisable that (for Windows OS) the Windows Registry is not the only source of information, but that the file system analysis should be performed as well with all actions over the Windows Registry and at the end the results from individual sources could be compared, or complemented by each other.
- **Reporting** - The last phase includes the process of preparing and presenting the information that follows from the analysis phase. So we can imagine it as a summary of information that will be important later. At this stage, it is also necessary to keep in mind that the document may be presented in court as evidence.

C. *Forensic analysis*

The term *forensic analysis* is basically the same as the third phase of the forensic process model and just as a reminder it is the stage, where the gathered data is further investigated. This will also be the main phase in which the tested forensic tools work.

II. Windows Registry

In this part we will take a closer look at Windows Registry, mainly what is Windows Registry and what is its purpose, structure of Windows Registry etc.

Value type	Description
REG_BINARY	Binary data in any form
REG_DWORD	A 32-bit number
REG_DWORD_LITTLE_ENDIAN	A 32-bit number in little-endian format
REG_DWORD_BIG_ENDIAN	A 32-bit number in big-endian format
REG_EXPAND_SZ	A null-terminated string that contains unexpanded references to environment variables
REG_LINK	A null-terminated Unicode string that contains the target path of a symbolic link
REG_MULTI_SZ	A sequence of null-terminated strings, terminated by an empty string (\0)
REG_NONE	No defined value type
REG_QWORD	A 64-bit number
REG_QWORD_LITTLE_ENDIAN	A 64-bit number in little-endian format
REG_SZ	A null-terminated string

TABLE I
Value types in Windows Registry [7]

A. *What is Windows Registry?*

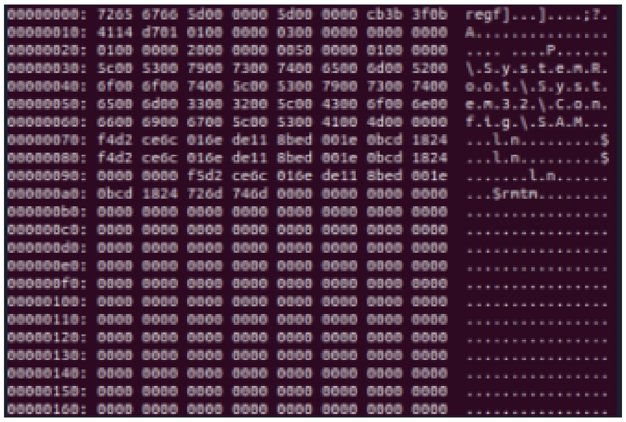
”Windows Registry is a central repository or hierarchical database of configuration data for the operating system and most of its programs” [1]. ”The Windows Registry maintains a great deal of configuration information about the system, maintaining settings for various functionality within the system” [3]. As mentioned above Windows Registry contains a great deal of information or potential interesting data and therefore it is appropriate to at least understand how it works, because it is an important part of windows forensic analysis.

The first appearance of Windows Registry was introduced in the OS Windows 3.1 (1992) and with upcoming operating systems it was further improved and upgraded. Of course, there are differences among the registry between the Windows OS, but our focus will be mainly on Windows 10, because based on following link it is the most used desktop OS.

B. *Structure of Windows Registry*

At the top of the hierarchy is the hive file, which can contain multiple keys or values. The data in Windows Registry is structured in a tree format. Each node in the tree is called a key. Each key can contain both subkeys and data entries called values (following link contains more information about the structure of Windows Registry).
Let's take a closer look at Windows Registry structure:

- **Hive files** - These are the main files, that can be found on our disks. They can contain multiple levels of keys and/or values. Examples of these files are SAM, SECURITY, SOFTWARE, SYSTEM, NTUSER.DAT, USERCLASS.DAT, which will be later analyzed and described in more details.
- **Keys** - Keys are the nodes in the mentioned hierarchical tree format. They can contain other keys or subkeys and values.



```
00000000: 7265 6766 5d00 0000 5d00 0000 cb3b 3f0b  regf]...]....;?.
00000010: 4114 d701 0100 0000 0300 0000 0000 0000  A...............
00000020: 0100 0000 2000 0000 0050 0000 0100 0000  .... .....P.....
00000030: 5c00 5300 7900 7300 7400 6500 6d00 5200  \.S.y.s.t.e.m.R.
00000040: 6f00 6f00 7400 5c00 5300 7900 7300 7400  o.o.t.\.S.y.s.t.
00000050: 6500 6d00 3300 3200 5c00 4300 6f00 6e00  e.m.3.2.\.C.o.n.
00000060: 6600 6900 6700 5c00 5300 4100 4d00 0000  f.i.g.\.S.A.M...
00000070: f4d2 ce6c 016e de11 8bed 001e 0bcd 1824  ...l.n.........$
00000080: f4d2 ce6c 016e de11 8bed 001e 0bcd 1824  ...l.n.........$
00000090: 0000 0000 f5d2 ce6c 016e de11 8bed 001e  .......l.n......
000000a0: 0bcd 1824 726d 746d 0000 0000 0000 0000  ...$rmtm........
000000b0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000000c0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000000d0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000000e0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
000000f0: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000100: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000110: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000120: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000130: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000140: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000150: 0000 0000 0000 0000 0000 0000 0000 0000  ................
00000160: 0000 0000 0000 0000 0000 0000 0000 0000  ................
```

Fig. 2. Beginning of the hive file (SAM)

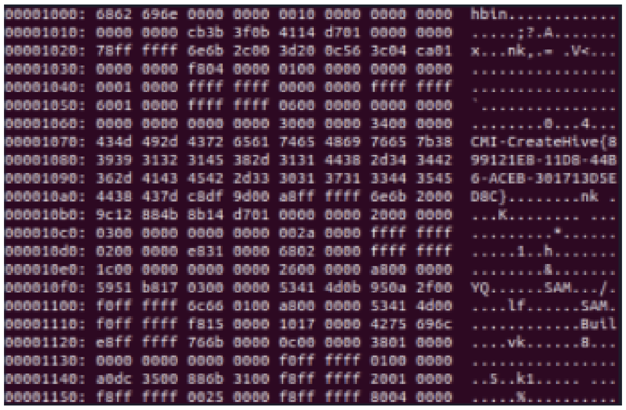
- **Values** - Values are like leafs in a tree. They cannot contain any additional subkeys or values compared to keys. A registry value can store data in various format, these types can be found in the table I. The most commonly used data types are DWORD (doubleword = 4 bytes), QWORD (quadword = 8 bytes) and SZ or string.

”If we compare the Windows folders and files with Windows Registry, then the keys and subkeys could be considered as folders and sub folders, and the values of a key could be considered as the files in a folder” [10].

C. *Binary structure of Windows Registry*

In forensic analysis but also in general it is important to know the binary structure of hive files. It is quite easy to just run one of the forensic tools against hive files, but for better understanding of this topic it is also necessary to know, what the hive files contain and how we can parse and further analyze them.

For this first minor practical part you will need a hex editor and also one of the hive files (you can extract it from one of the paths listed in table IV). In our case we will be using *xxd* and the sansforensics *SIFT* workstation.



```
00001000: 6862 696e 0000 0000 0010 0000 0000 0000  hbin............
00001010: 0000 0000 cb3b 3f0b 4114 d701 0000 0000  .....;?.A.......
00001020: 78ff ffff 6e6b 2c00 3d20 0c56 3c04 ca01  x...nk,.= .V<...
00001030: 0000 0000 f804 0000 0100 0000 0000 0000  ................
00001040: 0001 0000 ffff ffff 0000 0000 ffff ffff  ................
00001050: 6001 0000 ffff ffff 0600 0000 0000 0000  `...............
00001060: 0000 0000 0000 0000 3000 0000 3400 0000  ........0...4...
00001070: 434d 492d 4372 6561 7465 4869 7665 7b38  CMI-CreateHive{8
00001080: 3939 3132 3145 382d 3131 4438 2d34 3442  99121E8-11D8-44B
00001090: 362d 4143 4542 2d33 3031 3731 3344 3545  6-ACEB-301713D5E
000010a0: 4438 437d c8df 9d00 a8ff ffff 6e6b 2000  D8C}........nk .
000010b0: 9c12 884b 8b14 d701 0000 0000 2000 0000  ...K........ ...
000010c0: 0300 0000 0000 0000 002a 0000 ffff ffff  .........*......
000010d0: 0200 0000 e831 0000 6802 0000 ffff ffff  .....1..h.......
000010e0: 1c00 0000 0000 0000 2600 0000 a800 0000  ........&.......
000010f0: 5951 b817 0300 0000 5341 4d0b 950a 2f00  YQ......SAM.../.
00001100: f0ff ffff 6c66 0100 a800 0000 5341 4d00  ....lf......SAM.
00001110: f0ff ffff f815 0000 1017 0000 4275 696c  ............Buil
00001120: e8ff ffff 766b 0000 0c00 0000 3801 0000  ....vk......8...
00001130: 0000 0000 0000 0000 f0ff ffff 0100 0000  ................
00001140: a0dc 3500 886b 3100 f8ff ffff 2001 0000  ..5..k1..... ...
00001150: f8ff ffff 0025 0000 f8ff ffff 8004 0000  .....%..........
```

Fig. 3. Binary analysis of the block/hbin (SAM)

Offset (bytes)	Size (bytes)	Description
0	4	Size
4	2	Node ID (sequence ”nk”)
6	2	Node type
8	8	LastWrite time
20	4	Offset to this key’s parent
24	4	Number of subkeys
32	4	Offset to the list of subkey records
36	4	Number of values
44	4	Offset to the value list
48	4	Offset to security identifier record
76	2	Length of the key name

TABLE II
REGISTRY KEY STRUCTURE [3]

Everything in the hive file is divided into 4KB (4096 bytes) blocks, also called ”bins”. The first block starts with characters ”regf” and contains metadata about the analyzed hive file (e.g. the path to the analyzed hive file). Every other block begins with characters ”hbin”. In these declared blocks we will then find the defined keys and values that create the individual hive files. Of course, each key or value consists of a header and the data itself. All of the mentioned things can be seen at the figures 2 and 3.

In this part we will take a closer look at the binary structure of keys and values. They are stored in the hbins or 4KB blocks (as mentioned above) and we can identify them by looking for character sequence ”vk” for value or ”nk” for key. Keys have longer headers because they are building the hierarchical tree structure and values have smaller headers but they also contain additional data. The contents of the header structure, whether for a key or a value, can be seen in the tables II and III.

There are several important things to mention about the binary structure of the key. Firstly, it is the size. It is stored in little-endian format, so after reading these 4 bytes we will (usually) get number 4294967200. That would mean that our analyzed key will be about 4GB in size. This method is actually not correct and the proper way is to read this value as signed integer, then we will get a value of -96. This is more convenient, only the sign could bother us. And this is the way the Windows Registry indicates whether a given key is deleted or not. If the value of the number is greater than zero, it is a deleted key that has not yet been overwritten. This method could be a essential part of forensic analysis.

Secondly, it is the timestamp or LastWrite time. This part of the structure is also in the little-endian format, so after reading each byte and covert it from hexadecimal to decimal format, we will get a multiple digit number. Then it is necessary to

Offset (bytes)	Size (bytes)	Description
0	4	Size (as a negative number)
4	2	Node ID (sequence ”vk”)
6	2	Value name length
8	2	Data length
12	4	Offset to data
16	4	Value type

TABLE III
REGISTRY VALUE STRUCTURE [3]

use a converter (e.g. silisoftware) and covert the 64-bit number to file-time format.

After the 80 bytes, which is the length of the header, comes the name of the key. The size of the key name is always aligned to the nearest higher power of the number 16 (e.g. key name size = 11, actual key name size stored in registry = 16).

As for the binary structure of the value, there are also some interesting things to talk about. The size is the same as for the keys. The value type indicates in what format the corresponding data is stored. List of the possible value types can be found in the table I. Values also include the offset to the data itself, which is part of each value. What could be a disadvantage in terms of forensic analysis is that this binary structure does not include timestamps.

D. *Hive files list*

In this section we will take a closer look at each hive file listed in table IV. There will be a short description to each hive file and list of several important keys or values. Of course, there are more keys and values in the individual hive files than will be in the list, but the aim of this document is not to show all the possibilities, but to point out the most important ones (especially from the point of view of forensic analysis).

SYSTEM "hive file contains a great deal of configuration information about the system and devices that were included in and have been attached to it" [3]. List of the most important keys and values:

- *CurrentControlSet* - indicates ControlSet, which is used on a live system (volatile key), cannot be found during post-mortem analysis
- *Select* - suggest which ControlSet is loaded as CurrentControlSet
- *ComputerName* - name of the analyzed computer
- *TimeZoneInformation* - information about the time zone in which the machine is located
- *Network Interfaces* - serves to display all network interfaces, which are used in the ControlSet
- *FileSystem* - contains file system configuration information
- *Services* - under the given key are all services that run on the system and belong to the given ControlSet
- *AppCompatCache* - also known as "Shimcache is a component of the Application Compatibility Database, which was created by Microsoft and used by the Windows operating system to identify application compatibility issues" [6]
- *USB Devices* - contains considerable portion of information about USB devices that have been connected to the system
- *MountedDevices* - holds information about mapping USB devices on a specific drive letter

Name of the hive file	Path to the hive file
SYSTEM	C:\Windows\System32\config
SAM	C:\Windows\System32\config
SECURITY	C:\Windows\System32\config
SOFTWARE	C:\Windows\System32\config
AMCACHE	C:\Windows\appcompat\Programs
NTUSER.DAT	C:\account_name
USRCLASS.DAT	C:\Users\account_name\AppData \Local\Microsoft\Windows

TABLE IV
LOCATION OF THE HIVE FILES [7]

SAM or Security Account Manager is hive file, which contains information about the system-users. There is a short list with all essential keys and values:

- *RID* - or Relative Identifier (number to identify a specific user e.g. Administrator = 500) contains multiple user-specific values like username, when the account was created, last log-in time, password hint, etc.

SECURITY "hive contains some useful information regarding the system configuration and settings" [3]. Following list contains the most important keys or values:

- *PolAcDms* - "determines which users on a system are local users, and which are domain users" [3]
- *PolAdtEv* - contains audit policy (same as program auditpol.exe)

SOFTWARE contains mainly software and windows settings. It is most often modified or changed by running applications. List of the most important keys and values:

- *CurrentVersion* - contains values like *ProductName*, *InstallDate*, *InstallTime*, *CurrentVersion*, etc.
- *ProfileList* - information about local and domain users
- *NetworkCards* - maintains information about network cards
- *NetworkList* - maintains for example a list of values that include each of the networks to which the computer has been connected [8]
- *Run* - serves for programs to run at system startup, can be used for malware persistance
- *AppInit_DLLs* - "the AppInit_DLLs infrastructure provides an easy way to hook system APIs by allowing custom DLLs to be loaded into the address space of every interactive application" (more information can be found on the following link)
- *ShellExtensions* - can be exploited as malware persistence because all values under this key are loaded when Windows Explorer starts [3]
- *Browser Helper Objects* - loads .dll files that can act as browser extensions, again can be exploited as malware persistence

AMCACHE "file is a registry file that stores the information of executed applications" [6]. It is not considered part of the Windows Registry but has a similar binary structure [3] to other hive files and may also be subject to forensic analysis. Following keys and values will be the subject of our observation:

- *File* - "Files are grouped by their volume GUIDs. These are the same Volume GUIDs that you can find in the SYSTEM hive under MountedDevices and also under NTUSER.DAT MountPoints2" [5].

- *Programs* - based on the following link, this key includes information about installed programs on the analyzed OS

NTUSER.DAT contains user-specific information. Each system user has its own NTUSER.DAT hive file. This hive file contains following keys or values:

- *Run* - identical to the key in SYSTEM hive file (malware persistence)
- *RunOnce* - run the program under the given key when the user logs in (after execution the value will be removed from the key), malware persistence
- *Applets* - contains applications that are part of a set of Windows OS (e.g. Paint, Wordpad, etc.), includes also subkeys like *RecentFileList* with files that were last opened by these applications
- *RecentDocs* - provides a very good list with all files that have been recently visited [3]
- *ComDlg32* - ”refers to common dialogs available on Windows systems” [3]
- *TypedPaths* - captures paths entered manually in Windows Explorer
- *TypedURLs* - binds to the Internet Explorer and stores all url addresses specified by the user manually to the application
- *UserAssist* - contains a list of subkeys with the GUID that tells what programs were executed, at which time were they last executed and how many times were they executed in total

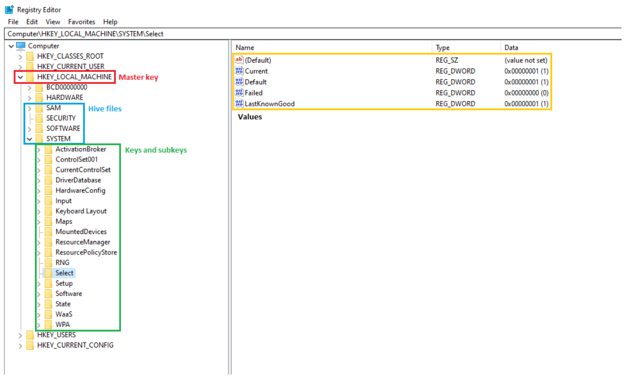
USRCLASS.DAT is similar to NTUSER.DAT hive file, but is not user-specific. With evolution of Windows operating systems, there are more and more keys and values migrating from NTUSER.DAT to this hive file [3]. List of the most important keys and values:

- *MUICache* - based on following link, ”each time that you start using a new application, Windows operating system automatically extract the application name from the version resource of the exe file, and stores it for using it later”
- *ShellBags* - everything under this key contains folder browsing information, especially the preferences of specific users

This brief sheet is just an example of what is all located in the Windows Registry. For a more detailed look at the individual keys and values, a book (Windows Registry Forensics: Advanced Digital Forensic Analysis of the Windows Registry), reliable sources or various cheat sheets are recommended for the further investigation.

III. Registry Editor

”Perhaps the commonly available tool for viewing the contents of the Windows Registry is the Registry Editor” [3]. It is also called simply RegEdit and it is a baseline tools on every Windows OS, which is using Windows Registry. If you have to use Registry Editor, just type in your search bar ”regedit” and after accepting the disclaimer from Windows you will be presented with the similar GUI as can be seen at the figure 4.



Computer\HKEY_LOCAL_MACHINE\SYSTEM\Select

```
Computer
  HKEY_CLASSES_ROOT
  HKEY_CURRENT_USER
  HKEY_LOCAL_MACHINE
    BCD00000000
    HARDWARE
    SAM
    SECURITY
    SOFTWARE
    SYSTEM
      ActivationBroker
      ControlSet001
      CurrentControlSet
      DriverDatabase
      HardwareConfig
      Input
      Keyboard Layout
      Maps
      MountedDevices
      ResourceManager
      ResourcePolicyStore
      RNG
      Select
      Setup
      Software
      State
      WaaS
      WPA
  HKEY_USERS
  HKEY_CURRENT_CONFIG
```

Name	Type	Data
(Default)	REG_SZ	(value not set)
Current	REG_DWORD	0x00000001 (1)
Default	REG_DWORD	0x00000001 (1)
Failed	REG_DWORD	0x00000000 (0)
LastKnownGood	REG_DWORD	0x00000001 (1)

Fig. 4. Registry Editor tool (GUI)

As a reminder, on the figure 4 you can see also the structure of Windows Registry (described in the chapter Windows Registry). On the left the master keys, which are used only for logical distribution, then hive files and keys. On the right side of the figure are values with types and with actual content of these values. The navigation between the keys is clickable, and if you happen to get lost, the path to the individual keys will come in handy. This path is located right above the set of master keys.

There are several benefits to this tool like quick forensic analysis of keys that may have been compromised, if you know the exact path to them (also possible to use *Edit->Find* or hit *Ctrl + F*), you can import/export hive files as (*File->Import/Export*) and among many other functions you can also create a list of your favorite keys or maybe all keys that should be analyzed in case of security incident, which would be useful, but this list would have to be created before a security incident occurs and creating it on all of your computers would take a portion of your time. Surely this tool would be great if you have nothing else or for a quick live (live == computer is running) analysis but there are many if-s and this tool is not suited for post-mortem analysis.

Let us take a closer look at the few main disadvantages of this tool. One of the more severe disadvantages is the fact, that this tool does not contain timestamps and timestamps are a critical part of forensic analysis. If you have read the chapter on binary structure, then you must know that each key also has a timestamp of last modification, but Registry Editor does not display them. Another disadvantage is that the registry editor is intended for commercial purposes and was not designed for forensic analysis. It means that you can freely edit, add or delete any key or value, which is unacceptable for the case of forensic analysis. For example you can modify the essential data in the analysis and thus spoil all the evidence you had at your disposal. Such a fact could be avoided if the forensic process model is followed correctly.

As mentioned above Registry Editor was not designed for forensic analysis, but following chapters will describe tools that were created for digital forensics.

A. *PowerShell and Windows Registry*

A small detour until we move on to the next chapter is the PowerShell, which is a crucial part of Windows OS. As this tool also allows scripting, it can be used to extract individual keys and values. Subsequently, such scripts could be written before the forensic analysis and only applied to specific hive files during the analysis. There is one big disadvantage that in can not parse offline registry hives (unless you use program reg.exe). More details can be found on following links (Working with Registry Entries and Working with Registry Keys).

Example of such command to extract values from key *Run* is shown bellow:
Get-ItemProperty -Path Registry::HKCU\SOFTWARE \Microsoft\Windows\CurrentVersion\Run

IV. Registry Explorer

"Registry Explorer is a GUI based tool used to view the contents of offline registry hives" [11]. This explanation together with the figure 5 describes this tool perfectly. At first glance, this tool looks similar to a registry editor, but after closer look, it has more capabilities and is considered as a good forensic tool. Registry explorer was developed by Eric Zimmerman and you can find this tool on the following link. There are several number of features in this tool like loading hive file (*File->Load hive* or hit *Ctrl + o*). After loading a hive you can navigate through it by clicking at keys and subkeys etc., similar to Registry Explorer. You can search for specific key, view the values or binary structure and explore the *Options*. There are several options in this menu, and we focus mainly on those that allow us to view deleted keys or values and possibly hidden keys.

This program has many advantages like timestamps, viewing deleted records, exporting hive files as various formats etc. and last but not least bookmarks. These bookmarks are really a fantastic way to quickly analyze loaded hive file. They are available almost immediately after the hive is loaded and not only do they contain the actual content of the keys and subkeys, they also contain description of each listed key. This feature is partly similar to the favorite in registry explorer but overall it is better thought out and implemented. With newer version it can also analyze live system, but you have to run this program as administrator, otherwise it does not have access to the hive files.

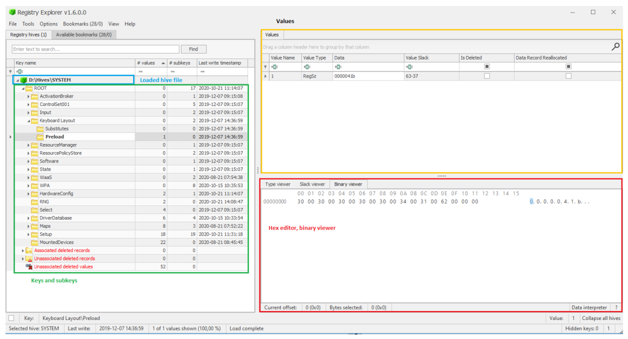
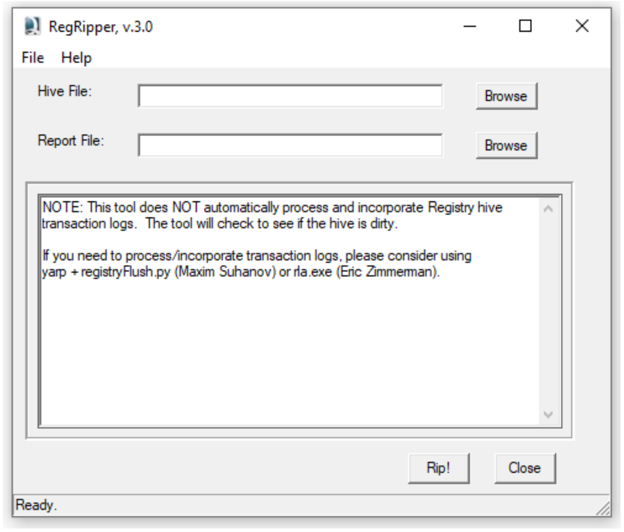

Registry Explorer v1.6.0.0
File Tools Options Bookmarks (28/0) View Help
Values
Loaded hive file
Keys and subkeys
Hex editor, binary viewer

Fig. 5. Registry Explorer (GUI)



NOTE: This tool does NOT automatically process and incorporate Registry hive transaction logs. The tool will check to see if the hive is dirty.

If you need to process/incorporate transaction logs, please consider using yarp + registryFlush.py (Maxim Suhanov) or rla.exe (Eric Zimmerman).

Fig. 6. RegRipper (GUI)

Registry explorer has almost no disadvantages. It is not working automatically like RegRipper which will be analyzed later and it shows really a great portion of information on just one screen/window. To capture all the important stuff, you will need a high resolution screen.

The listed disadvantages are only subjective and Registry Explorer is really an example of an excellent forensic tool.

V. Registry Ripper

Registry ripper also called RegRipper is another Windows Registry parser and it is created by Harlan Carvey. This tool contains a lot of plugins, each suited for the specific hive file. It was written and developed in perl programming language and as for now it is in version 3.0 (https://github.com/keydet89/RegRipper3.0). You can find two main programs in this repository, "rip.exe" or the command line version of this program and "rr.exe", which is the GUI version of this program. Our focus will be only on the GUI version, which can be seen at the figure 6. After starting the rr.exe, you will be welcomed with this simple GUI. There are several important components that we would like to briefly describe. At first you need to choose the path to the hive file, then you enter the path to the report file (.txt file with findings/evidence from the provided hive file). Lastly, there is a button "Rip!" that starts the whole process of analyzing. During the process you will be informed about the plugins, which are being run against the hive file and if they ended successfully. In the same path as yours .txt report file will also be a log file, with possible error messages. Partial example of the output file can be seen at the figure 7.

In terms of functionality, the RegRipper is really suitable for forensic analysis because it analyzes individual keys and values accurately and in detail. However, an even bigger advantage is that the tool can work completely automatically,

Forensic tool name	Advantages	Disadvantages	Score
Registry Editor	+ quick and easy way to view Windows Registry + availability on Windows OS + finding keys or values by name	- no timestamps - cannot be automated - values or keys can be modified	4
Registry Explorer	+ timestamps + bookmarks + hex editor + viewing deleted records	- not automated - too much information on one screen	9.5
RegRipper	+ easy to use + automated + simple GUI	- cannot do live analysis - no hierarchical tree/structure	8.5

TABLE V
COMPARISON OF PRESENTED TOOLS

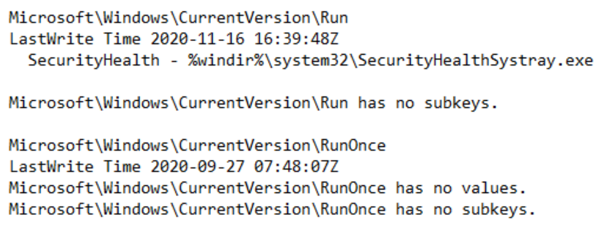
which can save a great portion of time, when dealing with multiple compromised machines.

There are also a few disadvantages to this program like there is no way that you can analyze hive files without copying them, because this tool cannot analyze live/running system. Another disadvantage is, that there are no key paths, view of the hierarchical tree or the binary structure. But on the other side H. Carvey wrote an amazing book with all the explanations, which are necessary when facing Windows Registry.

RegRipper is one of the more suited tools for Windows Registry forensic analysis, but as it works automatically and has a higher level of abstraction, there is a clear disadvantage when using it and not knowing basics about Windows Registry.

VI. COMPARISON OF PRESENTED TOOLS

In this part we will summarize everything about presented tools and evaluate those tools based on five usability components by J. Nielsen (learnability, efficiency, memorability, errors, satisfaction). Table V summarizes the most important aspects from the previous part and adds a new metric, which is the numerical evaluation (1-10, where number 1 means the worst grade and number 10 means the best) of individual items in the table. When evaluating the item ”Score”, we considered the advantages, disadvantages and we also included the outputs from the chapter **Further comparison and evaluation**. Please, keep in mind that attribute ”Score” is subjective that means it is mostly based on our opinion.



```
Microsoft\Windows\CurrentVersion\Run
LastWrite Time 2020-11-16 16:39:48Z
  SecurityHealth - %windir%\system32\SecurityHealthSystray.exe

Microsoft\Windows\CurrentVersion\Run has no subkeys.

Microsoft\Windows\CurrentVersion\RunOnce
LastWrite Time 2020-09-27 07:48:07Z
Microsoft\Windows\CurrentVersion\RunOnce has no values.
Microsoft\Windows\CurrentVersion\RunOnce has no subkeys.
```

Fig. 7. Output from RegRipper (key Run and RunOnce)

VII. FURTHER COMPARISON AND EVALUATION

Because all analyzed forensic tools are based on GUI, we can use 5 components of usability by J. Nielsen to evaluate these tools. For each of the five components, we will give a brief explanation and a forensic tool that is best suited for that component.
List of each component of usability (adapted to the topic of forensic analysis):

- *Learnability* or how easy it is for an analyst to complete basic tasks in a given tool - In this case the winner is RegRipper as it has a really simple user interface and the output file are also in easily readable format.
- *Efficiency* or how quickly analysts can complete basic tasks once they are familiar with the forensic tool - There is no clear winner for this component, because RegRipper is fast and searching for data in the output format can take some time. If an analyst knew how to use the full potential of Registry Explorer, he would be able to analyze individual hive files much faster than RegRipper.
- *Memorability* or how much time does the analyst need to relearn the basics of the forensic tool - Registry Editor and Registry Explorer are much harder to learn than RegRipper, so in this case the winner is again RegRipper, as it has simple GUI and there are not a lot of things that can make the process of relearning more difficult.
- *Errors* or how many error do analysts make, how severe are these errors and how easily can they recover from these errors - There is no winner among these tools, although perhaps the worst case is Registry Editor, because just one miss-click can modify the contents of the Windows Registry, which is absolutely terrible in case of forensic analysis.
- *Satisfaction* or how pleasant it is the forensic tool to use - This component is very subjective, but based on our decision the winners are Registry Editor and Registry Explorer. They have a truly simple yet amazing user interface.

Based on the five components of usability the overall winner is RegRipper, but of course that does not mean, that the other tools are to be discarded. This whole evaluation was very subjective and the best way to compare these tools, is to try them all and judge for yourself. If you do not have any specific use-case in mind, try the practical part in this document.

VIII. PRACTICAL PART

Before performing the laboratory exercise itself, it would be advisable to have a SIFT virtual machine, an NTFS disk image from the github repository and possibly another virtual machine with OS Windows to run forensic tools. Unfortunatly you will need to perform actions on the windows registry from the Windows virtual machine and everything else in the SIFT workstation. So be careful when moving hive files, do not forget to use hash functions and check each step. More about the instructions and everything about practical part you can find on this following github link (https://github.com/57972887/LaboratoryExerciseWR).

IX. Conclusion

In this document we talked about digital forensics as a science, then described Windows Registry, introduced forensic tools like Registry Editor, Registry Explorer and RegRipper. We compared these tools and finally we created a practical lab to test analyzed forensic tools, show you how the Windows Registry works and what artifacts we can find in it. This final part was also a challenge for you as a reader and a more practical view to get away from the reading.

There are several lessons to be learned from this document. Firstly, you should now know, what is digital forensic, the basics of digital forensic model and mainly the term forensic analysis. Secondly, what is Windows Registry, binary structure, tree structure (hives, keys, values), description of individual hive files and of course which keys and values belong to which hives. Lastly, you should have a basic overview of the forensic tools for analyzing Windows Registry, you should know the advantages and disadvantages of each presented tool and you should also test them in your own interest in order to use the forensic tool that is best suited for you.

X. Acknowledgement

Firstly we would like to thank to every used source of information, this document would not exist without all the knowledge of Windows Registry. Secondly, a big thanks to the forensic team at FIIT STU for helping to work on this document and for helping to better understand the topic of forensics. And lastly, thanks also go to all the readers, who have come this far. I hope you enjoyed reading this document.

References

[1] S. Anson and S. Bunting. *Mastering Windows network forensics and investigation*. John Wiley & Sons, 2007.

[2] B. Carrier et al. Defining digital forensic examination and analysis tools using abstraction layers. *International Journal of digital evidence*, 1(4):1–12, 2003.

[3] H. Carvey. *Windows Registry Forensics: Advanced Digital Forensic Analysis of the Windows Registry*. Syngress Publishing, 2011.

[4] K. Kent, S. Chevalier, T. Grance, and H. Dang. Guide to integrating forensic techniques into incident response. *NIST Special Publication*, 10(14):800–86, 2006.

[5] Y. Khatri. Amcache.hve in windows 8 - goldmine for malware hunters. http://www.swiftforensics.com/2013/12/amcachehve-in-windows-8-goldmine-for.html. Published: 2013-12-04.

[6] Lifars. Amcache and shimcache forensics. https://lifars.com/wp-content/uploads/2017/03/Technical_tool_Amcache_Shimcache.pdf. Published: 2017-03.

[7] Microsoft. Windows registry information for advanced users. https://docs.microsoft.com/troubleshoot/windows-server/performance/windows-registry-advanced-users. Published: 2020-09-08.

[8] J. Risto. Wireless networks and the windows registry - just where has your computer been? https://www.sans.org/reading-room/whitepapers/forensics/wireless-networks-windows-registry-computer-been-33659. Published: 2011-05-06.

[9] J. Tan. Forensic readiness. *Cambridge, MA:@ Stake*, 1, 2001.

[10] H. Xie, K. Jiang, X. Yuan, and H. Zeng. Forensic analysis of windows registry against intrusion. *International Journal of Network Security & Its Applications*, 4, 03 2012.

[11] E. Zimmerman. Registry explorer. https://www.oit.va.gov/Services/TRM/files/RegistryExplorerManual.pdf. Published: 2017-05-19.