3-12-2008

Extraction of User Activity through Comparison of Windows Restore Points

Damir Kahvedžić
University College Dublin

Tahar Kechadi
University College Dublin

Recommended Citation

Kahvedžić, D., & Kechadi, T. (2008). Extraction of User Activity through Comparison of Windows Restore Points. DOI: https://doi.org/10.4225/75/57b2703b40cbc

DOI: 10.4225/75/57b2703b40cbc
6th Australian Digital Forensics Conference, Edith Cowan University, Perth Western Australia, December 3rd 2008.

Extraction of User Activity through Comparison of Windows Restore Points

Damir Kahvedžić
Computer Science and Informatics
University College Dublin,
Ireland
Damir.kahvedzic@ucd.ie

Dr Tahar Kechadi
Computer Science and Informatics
University College Dublin,
Ireland
Tahar.kechadi@ucd.ie

Abstract

The extraction of past user activity is one of the main goals in the analysis of digital evidence. In this paper we present a methodology for extracting this activity by comparing multiple Restore Points found in the Windows XP operating system. We concentrate on comparing the copies of the registry hives found within these points. The registry copies represent a snapshot in time of the state of the system. Differences between them can reveal user activity from one instant to another. This approach is implemented and presented as a tool that is able to compare any set of offline hive files and present the results to the user. Investigative techniques are presented to use the software as efficiently as possible. The techniques range from general analysis, in which areas of high user activity are pinpointed, to specific techniques, where user activity relating to specific files and file types is found.

Keywords

Registry Restore-Points User Activity Reconstruction

INTRODUCTION

System Restore is a process in Microsoft Windows that monitors key system changes on a user's computer. Whenever a change that could jeopardise the system's stability is detected, System Restore copies the core system files and stores them in a hidden directory ("C:\System Volume Information_restore{GUID}") in the files system before allowing the change to take place. If the subsequent change results in an unstable system the user can simply reload the last know good configuration and undo the damaging changes. Typically a large number of these time points, called Restore Points, are found in the system. They present a snapshot of the state of the system at that point in time. Differences between them can highlight significant user activity that could be useful to a criminal investigation.

A number of different files are monitored by System Restore and are archived in the restore point (Microsoft (2007)). The user can specify which file types they want to monitor by modifying a particular system parameter. However, left as default, the system restore archives the Registry, COM+ database, the IIS metabase and other specific file extensions. A log of the changes is also stored.

The most important of these are the registry hive files. The registry is a central hierarchical database used in Microsoft Windows operating systems to store information that is necessary to configure the system for the users, application, and hardware devices. It provides a single location where installed programs, user profiles and settings can be stored and managed. Analysing different values in the registry not only reveals currently installed programs and the state of the operating system but would also give clues to recent opened files, folders and network connection and other user activity.

This paper presents a new methodology for extracting user activity from Windows Restore points. We design and implement a forensic registry analysis tool that is able to compare different Restore Points and hives on a file by file and key by key basis. The results are presented to the investigators to help them in determining how the system was used between the snapshots. A methodology for using this tool in the most efficient way is presented. Initially we focus on extracting the differences between the registry hives however the same technique can be applied to other sets of files stored in the Restore Points.

Restore Point Creation

The number of registry Restore Points stored on the file system varies from computer to computer. The factors that influence their creation include how often new drivers and large programs are installed, if the computer is powered on constantly and whether the user has turned off restore point creation. Regardless of these factors, Restore Points are likely to be found in the majority of the file systems. Once created, they store an exact copy of the active hives of the system registry. Honeycutt, J. (2002) described when Restore Points are created.

- On Schedule: The default is 24 hours.
- On Program Installation: The system may be backed up when a user installs a program that uses a particular type of installer.
- On Update: The system is backed up just before an update to the operating system takes place.
- On System Restore: The system is backed up before a system is restored using one of the Restore Points.
- On Driver Installation: Device drivers affect system stability so the system is backed up for security.
- On User Request: Users can create manual restore points whenever they chose.

The Restore Points are kept on disk for up to 90 days and are deleted after this time. As a result, it is not uncommon to find many different copies of the state of the system in a typical forensic investigation. Although these Restore Points are extremely useful to roll back unwanted system wide changes, they are also an invaluable forensic resource to provide insight into the state of the system at a given time.

Test Setup

Throughout the paper we will use a test file system to illustrate the investigative techniques on a practical example. The system is a typical home computer using the Windows XP operating system. The computer configuration and other relevant data are shown below. All settings dictating the frequency of the restore point creation were left in their default values. Nevertheless, the creation of the Restore Points was not done at regular intervals. The differences between the Restore Point creations varied from a single day to 10 days. In this case study the Restore Points found in the system had a total time range of 2½ months. In the rest of the paper, the Restore Points will be referred to with respect to its creation name (i.e. RP11) and the number of days after the oldest hive they were created (i.e. RP11 (25 days)). In this way a perspective is maintained on the time range between two Restore Points.

Computer Manufacturer	Dell
Operating System	Windows XP SP2
Number of Restore Points	17
Time Range of Restore Points	2 months
Frequency of Use	Light to Medium Use

Table 1: Test system configuration

The above computer system is used to demonstrate how user activity can be recreated with the comparison of the Restore Points. Initially we focus on the registry hives within these points. The validity of the activity found was confirmed by interviewing the owner of the above system. The demonstrations use the SOFTWARE and ntuser.dat hives respectively, since they hold most of the interesting evidence. Other hives can be used in a similar manner.

RPCOMPARE

RPCompare, (*R*estore *P*oint *Comparer*), is a tool developed to address the issues raised in comparing Restore Points. It is designed to compare the points in an offline forensic environment and present the differences to the user. It does not use the WMI interface or any inbuilt Windows functions. The tool includes techniques that can carry out the registry comparisons on a number of different abstractions depending on the time limits of the user. Initially, it includes techniques to find information on which files are stored in the points and in particular can find all the differences between the registry hives throughout these points. It can highlight any of the keys and values that have been deleted, added or modified in the interim. The methodology for using the best technique and the potential advantage of using them is described in subsequent sections.

The rest of this section will detail the comparison function used for comparing registry branches and individual registry keys. The latter scenario is an invariably slower technique but captures all of the changes between the hives and can therefore give a more complete result.

RPCompare takes any number of similar hives and compares either their keys or the values with each other. Any keys or values that are found are extracted and tagged with Added, Modified or Removed with respect to the more recent registry. The results are then presented to the user for further analysis.

Registry Comparison Function

At first, RPCompare uses the naming conventions of the Restore Points to order the points. Each Restore Point is named RP*xx* with *xx* being an integer numbering the Restore Point (Bunting, S. (2008)). To compare the registry keys themselves, RPCompare utilizes the ``Last Written Time" values present in all of the registry keys. The value is updated by Windows whenever an operation to write or modify the key's data is carried out on the data of that key. Each key, including the root contains this information. The time written to the value depends on the system clock, which can be manipulated by the user to show an inaccurate time. For the purposes of this study, we assume that the time is an accurate reflection on the real time, and that the time is consistent throughout the restore points.

The comparison function used in RPCompare is recursive and traverses the length of each hive tree comparing every node's time values. All the relations are with respect to the earlier node. If a node has a different time value to its corresponding node in the next hive then that node is tagged as modified. If it does not exist in the next hive then it has been removed. Finally, if a new node has been found in the new hive then it has been added. Values are compared only for those keys that have been tagged as modified.

Performance

Since RPCompare compares every key to its corresponding key in another hive, the complexity of its execution is (n*m), where n and m are the number of keys in the hive. In the worst case if the key is the root of the hive, the whole hive will be compared. In our tests, comparison of a mature hive, such as a SOFTWARE hive, is very time consuming; as such we present a number of investigative techniques to concentrate on specific branches of the hive or to limit the time range of the comparisons.

The next section details these techniques. They have been developed to give the investigator a progressively detailed view of the data. We classify the techniques into two categories; those that attempt to find large scale differences in the system such as installations/uninstallations of programs and those techniques that attempt to recreate the minute steps of the user. The latter techniques involve the processing of Most Recently Used (MRU) lists and other private attribute information that can highlight how the user used the system. A special processing needs to be carried out on these types of registry entries to understand their meaning and their relationship with the user.

LARGE SCALE HIVE COMPARISON

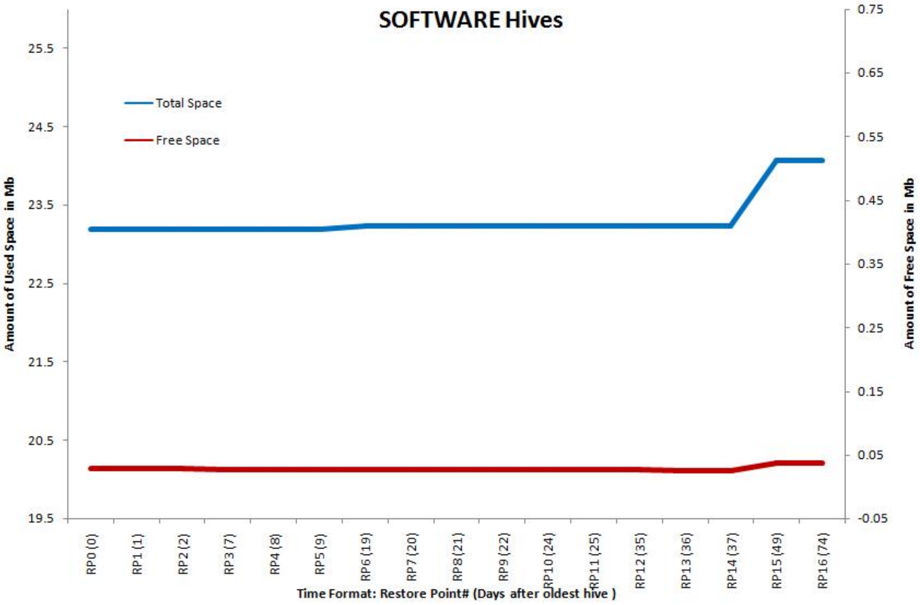
The investigator may need to expose large spots of activity and illustrate what was the general use of the system over a long period of time. User activity such as installation and uninstallation of programs and the addition or deletion of user accounts can be detected by comparing the registries of the system taken before and after the activity. The following technique illustrates a method on how the registries are compared in a progressively detailed manner. The process involves highlighting the areas of large activity by comparing file sizes first and then selectively comparing entire hives, then branches and finally values. The progressively detailed comparisons allow the investigators to streamline the comparison process and avoid spending too much time on complete hive comparisons.

Registry Size Comparisons

The first procedure entails comparing the sizes of the hives against each other. The differences can highlight some important changes that have occurred between time points in a relatively quick manner. Although small changes may not be noticeable, a large change to the hive, such as a program installation, can be easily spotted by comparing the hive sizes. The results can highlight a time point of high activity and bring it to the attention of the investigator for further investigation.

Variations in sizes highlight additions to the registry but it cannot show activity that *removes* keys from the hives. The extraction of unallocated space in the hive can be used to accomplish this aim. Whenever a key or a set of keys is deleted, due to an uninstallation or registry cleaning for example, the space left by the removed keys is marked as empty and is kept for future use (Russinovich, M.). The hives never shrink to compress this space and therefore do not reveal the uninstallation in its file size. In order to highlight this fact, RPCompare calculates the amount of free space in the registry alongside the total amount of used space. Sharp increases in the total amount of unallocated space signify large scale removal of keys.

Figure 1 shows the sizes of the hives in the case study. The size of the SOFTWARE hive increased at the latter part of the graph; between RP14 and RP15 which relate to 37th day and 49th day after the oldest restore point. Similarly, in the ``ntuser.dat'' file, the total space rises sharply between RP5 and RP6 or the 9th and 10th day after the oldest hive. Deallocated space has largely remained constant except between RP0 and RP1 and RP13 and RP14. The investigator can therefore narrow the range of the comparison for closer investigation.



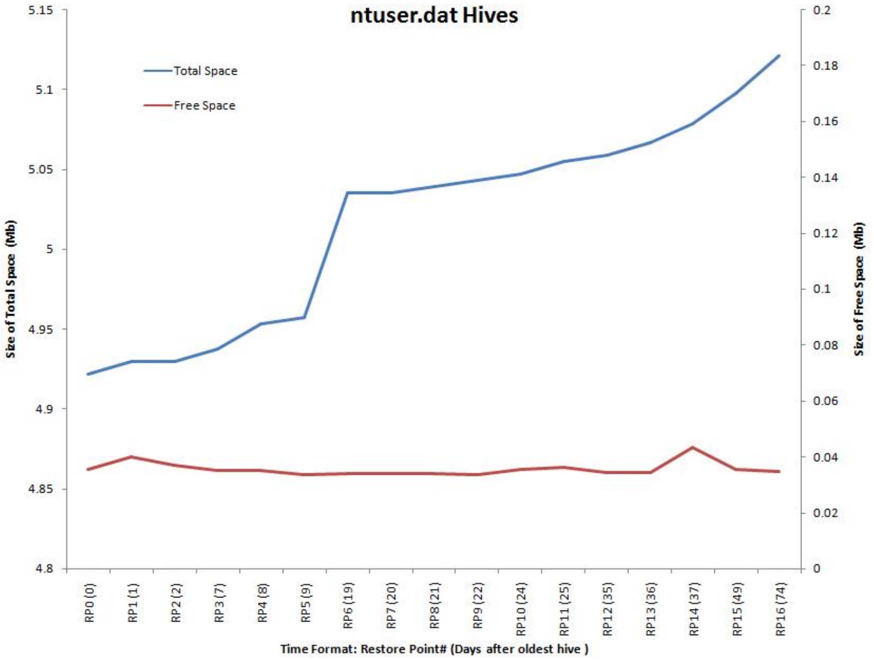
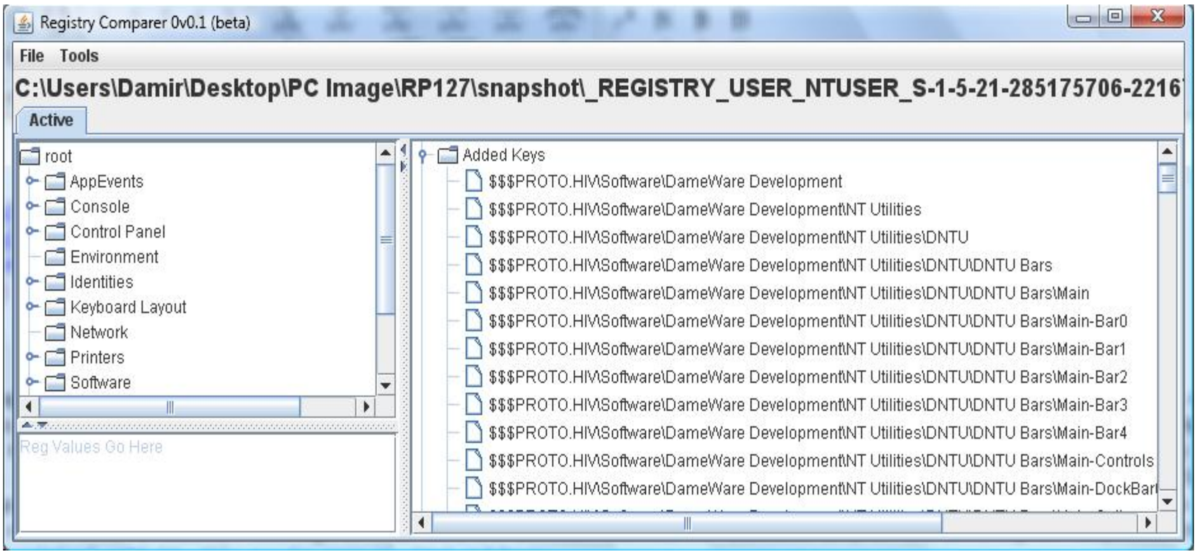


Figure 1: Sizes of the SOFTWARE and ntuser.dat Hives

RPCompare was executed on the SOFTWARE and ``ntuser.dat'' hives to recover the keys that were added at the above times. In the case of the SOFTWARE hive, RPCompare found that keys relating to the installation of the .NetFramework were responsible for the size increase in the 10 day time range between RP14 and RP15.

In the case of the ``ntuser.dat'' hive, the size of the hive began to increase at RP3 (7 days after the first restore point) with a huge size increase at RP6 and a steady increase thereafter. Comparison of RP5 and RP6 resulted in the identification of 143 Added, 111 Modified, 4 Removed keys. This result is shown in the Registry Comparer window of the program as shown in Figure 2. A majority of added keys are related to a DameWare (DameWare 2008) program. Upon further investigation, this program was found to be a PC remote control utility. Although in this case the installation was for innocent use, if the investigator is looking for a particular type of criminal activity this can be seen as vital evidence. The progressive increase in hive size from RP12 was attributed to new keys being added in the ``ShellNoRoam'' branch of the hive. These keys store window positioning preferences for each folder in the file system and are discussed further on in this paper. New ``ShellNoRoam'' keys indicate creation of new folders in the file system.



```
C:\Users\Damir\Desktop\PC Image\RP127\snapshot\_REGISTRY_USER_NTUSER_S-1-5-21-285175706-2216
```

```
$$$PROTO.HIV\Software\DameWare Development
$$$PROTO.HIV\Software\DameWare Development\NT Utilities
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU\DNTU Bars
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU\DNTU Bars\Main
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU\DNTU Bars\Main-Bar0
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU\DNTU Bars\Main-Bar1
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU\DNTU Bars\Main-Bar2
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU\DNTU Bars\Main-Bar3
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU\DNTU Bars\Main-Bar4
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU\DNTU Bars\Main-Controls
$$$PROTO.HIV\Software\DameWare Development\NT Utilities\DNTU\DNTU Bars\Main-DockBar
```

Figure 2: 143 Added, 111 Modified, 4 Removed Keys between Time Points 5 and 6

Registry Branch Comparisons

Once a branch of a hive has been found and suspected to contain evidence, RPCompare can take the root of that branch and compare it with similar branches in other hives. Because comparing a single branch is much faster than comparing the whole tree hierarchy, the investigator can concentrate on particular aspects of the hive at any given time relatively efficiently.

Returning to the DameWare example above, the DameWare Development key was compared to all the hives that were created after its installation. None of the hives reported any differences. This suggests that the program was rarely used. Upon further investigation it was found that the program was installed as a trial and not actively utilized. Progressively detailed key or branch comparisons can also be carried out if the investigator deemed it to be necessary.

USER ACTIVITY EXTRACTION

The registry contains many important locations that can be directly associated to the user and the way that the system was used at the time of the registry snapshot. The ``Most Recently Used'' (MRU) lists store evidence of files names, programs and other information that has been opened by the user in the recent past. They have been particularly highlighted as highly valuable pieces of user activity (Honeycutt, J. (2002), ForensicMatter.com (2008)). These locations are widely known and are actively analysed in most investigations. The investigator may look manually at the MRUs in the Restore Points but this can be extremely laborious. RPCompare can extract an MRU key, compare it across different Restore Points and extract the user activity that the MRUs held. This section elaborates on the MRUs and how they can be processed to gain understanding of user activity.

Registry MRU Management

The MRU key is a standard in Windows that store the most recently used items in the system. Each MRU `listens' for particular user activity and updates its content if this activity occurs. They store two types of values, a value for each of the entries and an index value, the MRU value, which stores a list of the entries in order of most recent.

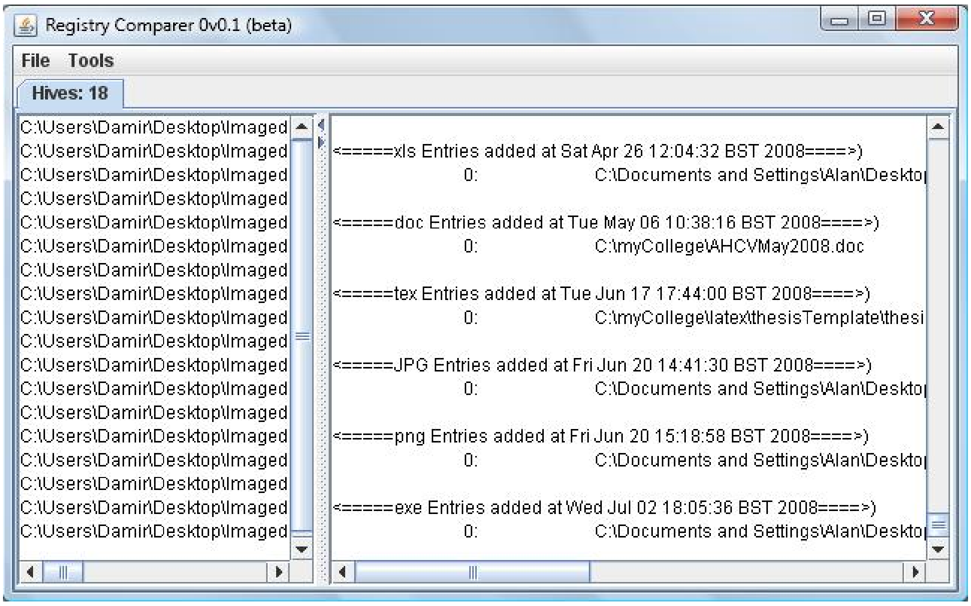
For example, the ``HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\OpenSaveMRU'' key stores the most recent files opened in the Windows open dialogue. The subkeys of the key store entries for specific file extensions. Any new file opened is captured as a value in the extension's MRU with an unused index as its name and the filename as its data. The index is placed at the head of the MRU list signifying that it is the most recent. Only a limited number of indexes exist, so if there are none free, the oldest entry in the list is removed and its index is given to the new entry. If a command has been executed and is already present in the MRU key, then its index is simply upgraded in the MRU list. No new values are created.

RPCompare and the MRU Timeline

In order to compare the MRU keys correctly, RPCompare contains an algorithm to combine the MRU lists and disregard any reoccurring differences. Therefore, if only one new entry is found, only the new command will be highlighted in the report. In this way the analyst can get a clear history of the list without being confused by the other repetitive values.

RPCompare was executed on the `OpenSaveMRU' key to extract the user activity held in this MRU. A timeline of the different MRU's timestamps can be created to illustrate this more easily. Figures 3 and 4 show the different perspectives. Figures 3 show a textual representation (obscured for privacy) while Figure 4 shows a timeline where peaks indicate higher amounts of new MRU entries and therefore more user activity.

Very few new MRU entries are created even for an extended time range. This low user activity indicates that the computer system was used very lightly. This corroborates the stated system specifications in the Test Setup Section. Although only OpenSaveMRU was analysed, RPCompare can aggregate other MRUs from other locations in the registry in a similar manner. The more timestamps that are collected the more accurate the MRU time line becomes.



```
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
C:\Users\Damir\Desktop\Imaged
```

```
<=====xls Entries added at Sat Apr 26 12:04:32 BST 2008====>)
                0:              C:\Documents and Settings\Alan\Deskto

<=====doc Entries added at Tue May 06 10:38:16 BST 2008====>)
                0:              C:\myCollege\AHCVMay2008.doc

<=====tex Entries added at Tue Jun 17 17:44:00 BST 2008====>)
                0:              C:\myCollege\latex\thesisTemplate\thesi

<=====JPG Entries added at Fri Jun 20 14:41:30 BST 2008====>)
                0:              C:\Documents and Settings\Alan\Deskto

<=====png Entries added at Fri Jun 20 15:18:58 BST 2008====>)
                0:              C:\Documents and Settings\Alan\Deskto

<=====exe Entries added at Wed Jul 02 18:05:36 BST 2008====>)
                0:              C:\Documents and Settings\Alan\Deskto
```

Figure 3: OpenSaveMRU textual Time line

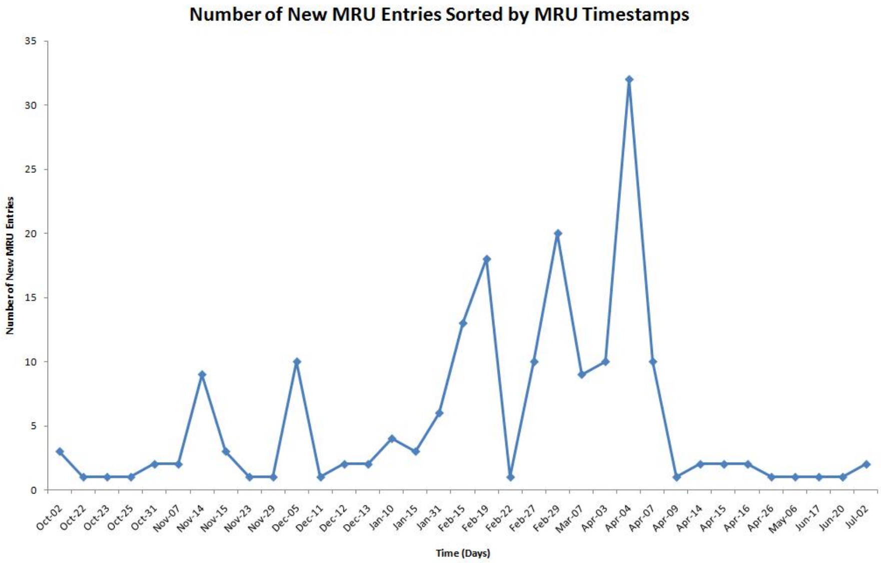
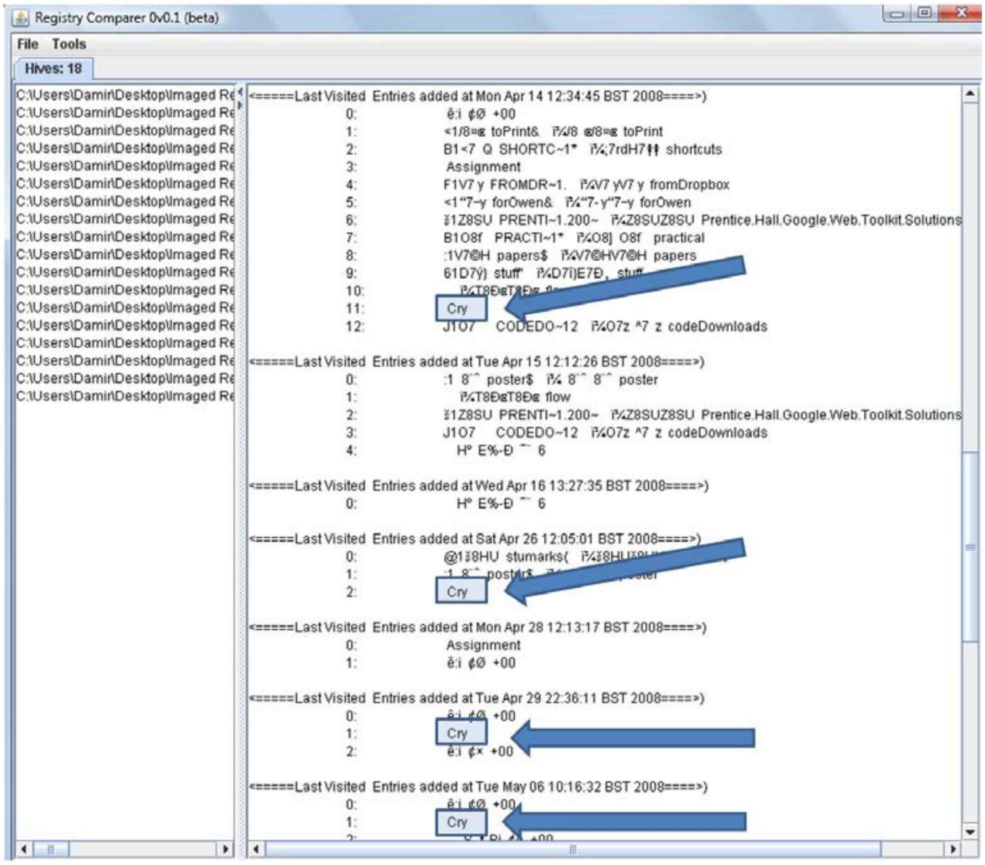


Figure 4: OpenSaveMRU Graphed Timeline



```
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
C:\Users\Damin\Desktop\Imaged Re
```

```
<=====Last Visited  Entries added at Mon Apr 14 12:34:45 BST 2008====>)
            0:              ê:i ¢Ø +00
            1:              <1/8¤ɛ toPrint&   ï¼8 ɐ8¤ɛ toPrint
            2:              B1<7 Q SHORTC~1*   ï¼;7rdH7‡‡ shortcuts
            3:              Assignment
            4:              F1V7 y FROMDR~1.   ï¼V7 yV7 y fromDropbox
            5:              <1"7~y forOwen&   ï¼"7-y"7~y forOwen
            6:              ǯ1Z8SU PRENTI~1.200~   ï¼Z8SUZ8SU Prentice.Hall.Google.Web.Toolkit.Solutions
            7:              B1O8f   PRACTI~1*   ï¼O8] O8f   practical
            8:              :1V7©H papers$   ï¼V7©HV7©H papers
            9:              61D7ŷ) stuff'   ï¼D7ï)E7Ð, stuff
            10:               ï¼T8ÐɛT8Ðɛ flo
            11:             Cry
            12:             J1O7    CODEDO~12   ï¼O7z ^7 z codeDownloads

<=====Last Visited  Entries added at Tue Apr 15 12:12:26 BST 2008====>)
            0:              :1 8¨ˆ poster$   ï¼ 8¨ˆ 8¨ˆ poster
            1:                ï¼T8ÐɛT8Ðɛ flow
            2:              ǯ1Z8SU PRENTI~1.200~   ï¼Z8SUZ8SU Prentice.Hall.Google.Web.Toolkit.Solutions
            3:              J1O7    CODEDO~12   ï¼O7z ^7 z codeDownloads
            4:                H° E%-Ð ¯ 6

<=====Last Visited  Entries added at Wed Apr 16 13:27:35 BST 2008====>)
            0:                H° E%-Ð ¯ 6

<=====Last Visited  Entries added at Sat Apr 26 12:05:01 BST 2008====>)
            0:              @1ǯ8HU stumarks(   ï¼ǯ8HU
            1:              :1 8¨ˆ poster$   ï¼             poster
            2:              Cry

<=====Last Visited  Entries added at Mon Apr 28 12:13:17 BST 2008====>)
            0:              Assignment
            1:              ê:i ¢Ø +00

<=====Last Visited  Entries added at Tue Apr 29 22:36:11 BST 2008====>)
            0:              ê:i ¢Ø +00
            1:              Cry
            2:              ê:i ¢× +00

<=====Last Visited  Entries added at Tue May 06 10:16:32 BST 2008====>)
            0:              ê:i ¢Ø +00
            1:              Cry
```

Figure 5: Subfolder MRU list of the Desktop folder. Highlighting `Cry' folder Access

Shell Bag MRU

It was seen in the Registry Size Comparison Section that a large number of ShellNoRoam keys are continuously created and were responsible for the size increase in `ntuser.dat' hive. These keys store positional information of windows for each folder in the file system. Each folder (bag) has its own MRU storing information on which of its subfolders was accessed most recently. RPCompare is able to process this information and present the user with user activity with respect to folders in a similar manner to the OpenSaveMRU. However, since every folder of the system has its own MRU, these keys contain much more information than the OpenSaveMRUs. It allows the investigator to highlight which folders where opened by the user and which were most widely used over a period of time.

Figure 5 show a sample of the timeline of the `Desktop' folder in the case study. It shows the activity relating to all its subfolders that were opened over a 3 week period. The folder `Cry' is highlighted in 4 out of 6 Restore Points. This indicates that the screen position of the folder `Cry' was changed at least 4 times in that time period. A user must have opened this folder and repositioned its window. This signifies significant user activity. If an investigator is looking for activity relating to a suspiciously named folder, this can be seen as vital incriminating evidence.

MRUs for other folders are found elsewhere in the registry in a similar format as the Desktop folder shown above. Using RPCompare, the investigator can extract any user activity relating to any folder in the system. The investigator can combine the evidence found in the Bag MRUs with those MRUs relating to the files themselves to paint an ever more detailed picture of user activity.

RELATED WORK

Registry comparison softwares have existed for a number of years in the registry analysis fields (RegDiff (Ver3.3), WinDiff (Ver5.1) (2001)). Exact registry specifications have not been published by Microsoft. Therefore, one of the techniques to see what the function is of any key is to take snapshots of the registry before and after known activity and analyse the difference (Microsoft (2008), Honeycutt, J. (2002)). However, these programs are limited in either the focus of the comparison and on what hives they operate on. Most of the registry comparison softwares are limited to comparing .REG files; ASCII versions of the binary registry hives. Each time a snapshot is taken, the relevant hive is exported with the inbuilt Microsoft RegEdit32 tool. This added step is undesirable in the forensic community where the goal is to avoid any modification of the original file. Other tools (RegDiff (Ver3.3)) can compare only the active registries and rely on commands provided by Microsoft Win32 API to extract relevant registry keys. Therefore they are unsuitable for offline registry analysis.

The tools above can only compare two hives at any one time and are not designed for digital forensic investigations. RPCompare differs to these programs since it has a digital forensic focus and aims to extract meaning out of the differences with respect to user activity. It can parse any offline registry hive even when it is extracted from a live system independently from any API.

Investigators have for a long time acknowledged the value of analysing the registry for evidence (Carvey, H. (2005), Carvey, H. (2007)). Guides have been published to explain to the investigators which keys are the most relevant to particular investigations (ForensicMatter.com (2008)). Most current forensic suites, EnCase (Encase (Ver6.8) (2008)) or Forensic Toolkit (FTK (Ver1.62.1) (2008)), contain registry parsers that can parse any registry hive files and present the contents to the investigator for analysis. However, the forensic analysis of the restore points has been treated the same as the analysis of the active registry. Namely, the investigator must open the hive files manually and access the different registry keys.

Research into analysis of the registry with respect to retrieval of deleted data has been done by Morgan, T.D. (2008) and Y. Kim et al (2008). The latter concentrated on retrieving still active keys not deleted by uninstalled programs. These clues are highly dependent on the uninstallation process of the software and may not reveal much information.

Research into Restore Points with respect to forensics has only been tackled recently (Bunting, S. (2008), Carvey, H. (2006), Harms, K. (2006)). Harms, K. 2006 has illustrated how the information stored in the Restore Points can be used to uncover evidence of a system intrusion. However, the author concentrates on the analysis of the ``change.log'' file only. This file is created at every Restore Point and tracks all files saved throughout the restore process. The registry hive files are not analysed.

CONCLUSION:

This paper presented a new approach for extracting user activity in a digital investigation. Namely, it focuses on comparisons of the Restore Points in general and the registry hives stored within them in particular. Differences between them can highlight changes in user activity that can be useful in digital investigation. We introduce a tool, RPCompare; an offline, self contained and integrated environment that can compare Restore Points and registry hives and present the user with the differences in a clear and logical interface. We also present a methodology using this tool to streamline the investigative process. Two techniques were presented in particular. The first focused on the registry in its entirety and attempted to ascertain time points of high user activity. This activity included what software was installed and removed and which keys were added or deleted relating to this activity. The technique, based on comparisons of hive size as well as content, was structured in a series of progressively detailed comparisons which highlighted areas of user activity with progressively higher levels of accuracy. The technique guided the investigator away from time consuming wholesale hive comparison and into much more efficient selective hive and branch comparison.

The second technique focused on the user trail itself and recovered and analysed the Most Recent Used (MRU) keys of the hives. The analysis of the MRUs requires specific processing for them to be investigated properly. User activity was extracted with respect to `file open' MRUs as well as `folder access' MRUs to get a complete user trail. In particular it was shown how RPCompare can reveal which folders and files the user accessed more frequently. Using both the timestamps of the MRUs and the hives, RPCompare presented an informative account of how the system was used by the user in a clear and useful manner to the investigator. This evidence can be extremely useful to any cybercrime investigation.

FUTURE WORK

RPCompare will be further enhanced to streamline the techniques presented above and to add new functionality in the comparison function. In this paper, we have concentrated mainly on the registry hives found within the points. Further work needs to be done to allow the software to utilise other similar files in the Restore Points and gather more information on what has changed between one point and another.

As described in the Large Scale Registry Comparison section, the investigator progressively narrows down their analysis of the registries by focusing on less and less branches. At the start of this process, a large number of differences may be returned that may be irrelevant to the investigation. In the DameWare scenario for example, a large number of HKLM\Software\Microsoft\Windows\ShellNoRoam\Bags\ keys were present. These keys store positional information on windows that the user has opened. Although it may be relevant to the investigator to parse these and extract useful information from them, in finding traces of added \ removed programs, these keys were not relevant. Future development of RPCompare will include filters to remove unwanted keys from the results or mark them as being irrelevant to the case.

Microsoft has enhanced the MRU list standard by storing the values in the MRU key in binary format, MRUListex. This allows the MRUListex keys to contain much more information than a single `Run' command or a filename. The data contained appears to be different for each type of key and is used extensively in Vista. In the HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\LastVisitedMRU key in Windows XP for example, every value stored in the key contains the filename of the document as well as the program that executed it. In Windows Vista, the same key is renamed to LastVisitedPidlMRU and evidently stores much more information. Processing of these MRUListex in Windows Vista has not been implemented in the RPCompare software.

REFERENCES:

Bunting, S. (2008) University of Delaware Police Computer Forensics Lab. *Restore Point Forensics*. URL http://128.175.24.251/forensics/restorepoints.htm, Accessed Sep 2008

Carvey, H. (2005). “The Windows Registry as a Forensic Resource”. *Digital Investigation*, 2(3), pp201-205

Carvey, H. (2006). “Restore Point Forensics”. URL http://windowsir.blogspot.com/2006/10/restore-point-forensics.html, Accessed Oct 2008

Carvey, H. (2007). “Registry Analysis”, in *Windows Forensic Analysis DVD Toolkit*. Syngress Press, pp125-189

DameWare (Ver6.0). (2008) Dame Ware Development. URL http://www.dameware.com. Accessed: Sep 2008

Encase (Ver6.8) (2008) Guidance Software Digital Investigations URL http://www.guidancesoftware.com/, Accessed 28/Mar/2008.

ForensicMatter (2008). *Forensicmatter.com: Registry Hives*. Available at URL http://www.forensicsmatter.com/registry_hives.php, Accessed 19/May/2008.

FTK (Ver1.62.1) (2008) Access Data, URL http://www.accessdata.com/, Accessed 31/Mar/2008.

Harms, K. (2006). “Forensic Analysis of System Restore Points in Microsoft Windows XP”. *Digital Investigation*, 3(3), pp151-158

Honeycutt, J. (2002) *Microsoft Windows XP Registry Guide*. Microsoft Press

Microsoft (2007). “Monitored File Extensions”. URL http://msdn.microsoft.com/en-us/library/aa378870(VS.85).aspx, Accessed Oct 2008

Microsoft (2008). “How to use WinDiff to Compare Registry Files”. URL http://support.microsoft.com/kb/171780, Accessed Sep 2008

Morgan, T.D. (2008) “Recovering Deleted Data from the Windows Registry”. *Proceedings of Digital Forensic Research Workshop 2008*, pp33-42

RegDiff (Ver3.3). Available at URL http://p-nand-q.com/download/regdiff.html, Accessed Sep 2008

Russinovich, M. “Inside the registry”. URL http://technet.microsoft.com/en-gb/library/cc750583.aspx. Accessed: Sep 2008

Y. Kim et al (2008) “Suspects' Data Hiding at Remaining Registry Values of Uninstalled Programs”. *Proc. Of The 1st Int. Conference on Forensic Applications And Techniques In Telecommunications, Information, And Multimedia And Workshop*, 2008.

WinDiff (Ver5.1) (2001). Microsoft, Available at URL http://www.grigsoft.com/download-windiff.htm, Accessed Sep 2008

COPYRIGHT