Tracking User Activity on Personal Computers

Anthony Keane[1] and Stephen O'Shaughnessy[2]

Network Security & Computer Forensics Research Group
Institute of Technology Blanchardstown, Dublin 15, Ireland
anthony.keane@itb.ie, stephen.oshaughnessy71@gmail.com

Abstract. Combining low cost digital storage with the tendency for the average computer user to keep computer files long after they have become useful has created such large stores of data on computer systems that the cost and time to conduct even a preliminary examination has created new technical and operational challenges for forensics investigations. Popular operating systems for personal computers do not inherently provide services that allow the tracking of the user's activity that would allow a simple personal audit of their computers to be carried out so the user can see what they were doing, when they did it and how long they spent on each activity. Such audit trails would assist in forensics investigations in building timelines of activity so suspects could be quickly eliminated (or not) from an investigation. This paper gives some insight to the advantages of having a user activity tracking system and explores the difficulties in developing a generic third party solution.

Keywords: Information Security, Computer Forensics, FTK, Timeline Analysis.

1 Introduction

The most popular operating systems in use today are Windows provided by the Microsoft Corporation, MAC OS provided by Apple Inc. and a host of Linux operating systems provided by different companies and open source groups [1]. None of the operating systems provide the user with a user activity tracking service so it is left to third party companies to offer software tools for providing such a service and yet, none of these tools are 100% accurate [2]. At first glance, the problem of tracking a user's activity looks trivial enough, after all the computer processor is responding to instructions of code so, as nothing can happen on the computer that isn't coded, surely monitoring the execution of code would reveal all? The difficulty is finding a way to capture the instructions so as to interrupt and understand what they are doing and then to extend this to the many different operating systems in use. Alternatively, we look at the activities of the computer system software, the user and other applications that are active and for each of these activities, look for some evidence of their execution that was left behind on the disk. The challenge to this latter approach requires that we understand the operating systems activities and all the programs that run on the systems. We need to know how they work so we can see what trail of evidence they leave behind after execution. Do they generate log data,

P. Gladyshev and M.K. Rogers (Eds.): ICDF2C 2011, LNICST 88, pp. 188–196, 2012.

write to the Registry, call on other programs, and so on? This paper discusses the development of a prototype application for user activity tracking that focuses on the Microsoft XP and Vista Operating Systems for now with the hope of expanding to other operating systems in the future.

2 Data and Information Storage

For the Microsoft Windows Operating Systems, the primary source of information on the system and its components is the Registry [3]. The Registry is essentially a database for configuration data that is stored in a hierarchical structure. The System, Users, Applications and Hardware drivers make constant use of the Registry during the operation of the computer.

As well as the Registry, individual applications and programs can have their own storage areas separate to the registry where data can be logged. These logs can be in the form of basic ASCII text files or binary priority structured files. The former are easy to read while the latter require knowledge of the binary structure in order to parse any meaningful information. Approaches to reading binary files can take the form of trial and error until the correct structure is obtained.

The information or data stored in system and application log files can vary in detail and its usefulness to determine the actions of a user on the system. The location of the log files can also vary and depends entirely on the application generating the files. The question as to why this data is stored, and what the original purpose was for the data very often remains known only to the creator of the program.

Key applications of interest are web browsers, email clients and other applications that take part in communications and social networking. These types of applications tend to leave a lot of information on the system to maintain continuity between sessions of activity. Most of this data has little to contribute in building a profile of a user's movements other than there was some activity and the quantity or duration of the activity. However there is always some data that can be used to establish time points of activity like when the application was run and what it was used for.

Automating the log collection and parsing of data into a database for further analysis can be problematic mainly due to different versions of system and application software storing the logs in different locations and with possibility different formats of the log file.

3 Developing the System

This paper introduces a *work in progress* tool called the User Activity Tracker (UAT). The aim of UAT is to provide a rapid and flexible solution in the step between the Acquisition and Discovery stages of a forensic investigation. The UAT will quickly analyse the data storage of acquired suspect’s computer device, consolidate the retrieved data into a common format and from this produce an accurate timeline of user activity. This can aid in the investigative process by giving

the investigator a quick overview of user activity on the system, thus indicating if a full investigation is warranted or not and an indication of the cost of proceeding with the investigation.

The main difference between UAT and other currently available forensics tools is UAT's ability to automate the process of user activity timeline retrieval. Most of the current tools lack the ability to extract usable information from within files or correlate timestamps into a timeline. There exists previous work on frameworks for timeline creation, such as Ex-Tip[4], System Combo Timeline [5] and most notably, Log2timeline [6]. Log2timeline is a timeline framework, capable of parsing many different log files and other artefacts from a system, creating what the author calls super timelines. According to the author, the super timeline often contains too many events for the investigator to fully analyze, making data reduction an easier method of examining the timeline essential. The UAT differs from these tools by concentrating on user activity data only, providing a much simpler, easily-read timeline.

UAT works by taking in user-defined parameters, such as time frame and location, scanning the various parts of the computer (file system, Registry, user logs, Internet history etc.), gathering all relevant data and storing it in a temporary data table structure. Before storing, each item of data is converted to an evidence object, which is essentially a data structure that allows data and their attributes, normally stored in different forms, to be expressed in a universal format. This is essential if the data is to be stored and searched in an efficient manner.

This is a work in progress report and the development of the automation and reporting processes are still being undertaken. The automation process will involve taking the data gathered from the system and based on the file types, locations and metadata (MAC times etc.), will construct a timeline of the user's activity within the user-specified time frame. The reporting mechanism will present this information in a user-friendly readable format such as pdf or html.

This report is based on developing the prototype system to test the software's ability to rapidly analyse the system to decide if further investigation is needed. The project surrounding the development of UAT is based on retrieving user activity data from Microsoft Windows operating systems and as such, the programming language chosen as the development platform had to be one that was compatible with the Windows environment in order to make best use of the available DLL libraries and supporting tools. Windows operating systems and applications are developed in Microsoft's own programming languages, C++ and C#, so ultimately C# was chosen for its ease of use and GUI-building capabilities.

It is intended that other operating systems will be explored to see how they differ and what changes to UAT would be required to allow universal access to all operating systems and their applications (see figure 1). Sections of the code will be optimised to take advantage of the different operating systems.

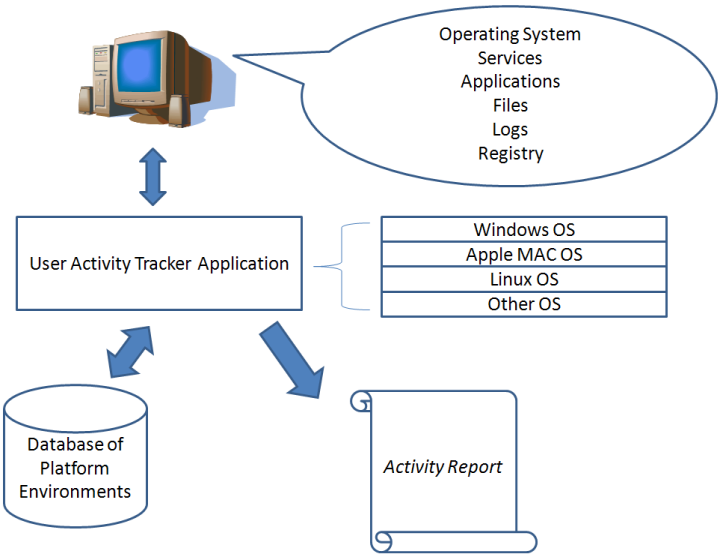

Fig. 1. Architecture of Computer System Independent UAT

Figure 1 illustrates the UAT, structured as a universal interface for multiple operating systems. The UAT sits on top of a database containing an array of operating systems and services, such as registry data, applications, their versions, name, structure and location of relevant log files etc. This would allow the UAT tool to be platform independent, capable of retrieving user activity information across different operating system platforms.

The main challenges to the development of system are:

- Identifying, finding and getting access to the log files.
- Extracting data and information from the registry.
- Access to user profiles, i.e., registry configuration for each user on the system.
- Parsing of custom binary files such as Internet Explorer (IE) index.dat files
- Automation of the process of constructing timelines of user activity, based on the information retrieved from the system.
- Generation of understandable, user-friendly reports of user activity.

4 The Timeline Data

To rapidly build the timeline, we focus on the areas of the system used for storing data of user activity thus reducing the large amount of data that is essentially irrelevant to our cause. UAT extracts and processes timeline data from the Registry, file system, event logs, Internet history and shadow copies or restore points.

The Registry is the first place to search for evidence of user activity since there are numerous keys that are altered by a user's actions. Each key has a *Lastwrite* time,

which indicates when the key was last written to, and is useful when creating activity timelines. Keys of particular interest in the Registry are the *UserAssist* key, which tracks the use of applications, shortcuts and other items by frequency of use and the last time accessed; the *MRU* keys, which list files that have been most recently accessed; the *TypedURLs* key maintains a list of the URLs that the user types into the Address bar in Internet Explorer and when combined with the Temporary Internet Files will show which Web sites were visited by clicking a link and those that the user typed in by hand; the *MountedDevices* key stores information about all devices and volumes mounted to the system and the USBSTOR key stores specific information about all USB devices plugged into the system. These keys are all located in the user's profile configuration, within the NTUSER.DAT file.

The file system contains many folders and files that can be used to give information on the user's actions. Among the more useful are the various log files, such as the Windows event logs, firewall logs and application logs. The Windows Event Logs store information that is gathered from various parts of the system, such as user login information or when a services starts and stops. Event Logs are therefore useful in the correlation of events during forensic analysis [7].

Also of interest in the file system are prefetch and shortcut file types. Prefetch files contain information on the applications that have been executed on the system, such as the number of times they have been executed and the timestamp of their last execution. Shortcut files are recognised by their .lnk extension. When a user accesses a document on their hard drive, a removable storage device, or a network share, a shortcut is created on the system in the Recent folder. Shortcut files can prove useful during an investigation as they can provide information on files accessed and devices attached to the system by the user.

Internet history files from Web browsers such as Microsoft's Internet Explorer or Mozilla's Firefox retain a record of the browsing history of users. Investigators can use this information to develop an understanding of the user's Web browsing activities.

Shadow Copies were introduced in Windows Vista and are similar to the restore points found in Windows XP. These files essentially contain a snapshot of the system and so previous shadow copies can be examined to determine any changes that may have occurred in the system since the date of the shadow copy's creation.

5 Case Study Analysis

A test case scenario was devised whereby a user gains unauthorised access to the Administrators account and performs certain actions. The aim here was to test the effectiveness of the UAT and to verify by duplicating the analysis with a popular commercial forensics tool, the Forensic Toolkit (FTK) by Access Data.

In the scenario, an administrator discovers a critical accounts file, *ClientListx10045.xlsx*, is missing from the system, in this case a PC running Vista and 160GB hard disk. The administrator had been away from the system from 29/05/ 2011 until 30/05/2011. What was the activity on the system between these two dates?

Another user gained unauthorised access to the administrator's account and performing a series of actions, including taking a file and creating another user account with administrator privileges.

5.1 Methodology

This section gives a brief overview of the methods employed by both UAT and FTK to investigate the incident in order to build the timeline of activity. An image of the system was taken using FTK Imager and this image was used for the investigations carried out by both tools.

The following questions were based on the scenario and both UAT and FTK were tested for their ability to find the evidence to answer the questions.

Question A: Had the missing file been accessed on the administrators account, and if yes, at what time?

- **FTK:** A simple file search for the missing file returned 3 hits in the index.dat (Internet History) file. The file was accessed through the URL*: C:\Users\ Administrator\Documents* at 09:34:56 on 30/05/2011 under the Administrators account.

- **UAT:** Using UAT's string search facility, a search was conducted for the missing file. Similarly to FTK, the search yielded a match found in the index.dat file (Internet Explorer history file) for the *URL C:\Users\Administrator\Documents*at 09:34:56 under the Administrators account.

Question B: Name any other users that were logged into the system around this time.

- **FTK:** To find what users logged into the system, the SAM Registry file was exported and then viewed in FTK's Registry Viewer. This showed (using the last write time for each user listed in the SAM file) that *Stephen_2* last logged in at 08:33:48 on 30/05/2011 and the Administrator account was last accessed at 09:54:37 on 30/05/2011. Investigation of the System Volume Information folder (restore points folder) on Windows Vista showed a shadow volume was created on 30/05/2011. An image of this restore point volume was taken and opened in FTK. A search of the Users folder on the C drive revealed another user, John Smyth. Loading the NTUSER.DAT file for this user into the Registry Viewer revealed that this user was last logged on at 9:53:21.

- **UAT:** Using the file type filter on UAT, a refined search was conducted for all security event log entries around this date. The search showed that two other users were logged in around the time of the incident: *Stephen_2* and *John Smyth*. Stephen_2 is a registered user on the system but John Smyth is not. A scan of the file system showed no user account existed for John Smyth.

Question C: Determine the length of time they were logged in.

- **FTK:** There were issues with the reading of the event logs through FTK, so the login times for each user could not be determined. The last access times were recorded as in the previous question.

- **UAT:** The logon/logoff times were determined by correlating the security log event 4624 (logon), with event 4634 (logoff). It was found user *Stephen_2* was logged in from 08:05:33 until 08:33:48;*John Smyth* was logged in from 09:39:39 until 9:53:21. The Administrator was logged on from 09:34:08 to 09:39:29 and also from 09:53:33 to 09:54:37.

Question D: Were there any peripheral devices attached to the system? If so, what were they?

- **FTK:** FTK's *Registry Viewer* application was used to determine if any USB devices were attached to the system. The System hive from the image was loaded into the Registry Viewer and the following keys were inspected:

  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Enum\USBSTOR
  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\DeviceClasses

  The USBSTOR key creates a sub key for each USB device that is plugged into the computer. Each device is given a unique ID for identification purposes, similar to a MAC address on a computer. Also stored in this sub key are the names of the USB devices. Under the DeviceClasses key, two sub keys were examined:

  {53f56307-b6bf-11d0-94f2-00a0c91efb8b}
  {53f5630d-b6bf-11d0-94f2-00a0c91efb8b}

  These two sub keys are globally unique identifiers (GUIDs) that contain a list of the USB devices as in the USBSTOR key. Each USB device entry has a unique *ParentIdPrefix* number that corresponds to the unique ID of the device in the USBSTOR key. Correlating these two unique identifiers gives the name of the device (from the USBSTOR key) and the last time it was plugged into the system (from the last write time of the corresponding *DeviceClasses* sub key).

  Using this method, it was discovered that a device called *Kingston Data Traveler G3 USB Device* attached to the system on 30/05/2011 at 09:43:21.

- **UAT:** UAT searches through each of keys mentioned above and automatically correlates the unique instance identifier for each device with the ParentIdPrefix for the devices under the two DeviceClasses sub keys. Similarly, UAT found the device, *Kingston Data Traveler G3 USB Device* attached to the system at 09:43:21on 30/05/2011.

Question E: Is there any evidence of each of the user's activities during their logged in period?

- **FTK:** The ntuser.dat file for user John Smyth, from the restore point image, was loaded into Registry Viewer. A search of the recent documents sub key showed the ClientListx10045.xlsx file was accessed at 09:42:24 on 30/05/2011. A search of the UserAssist key, which tracks application use, shortcuts and other items by frequency of use and the last time accessed by a user, showed that that user John Smyth accessed Excel.exe (the executable for Microsoft's Excel application) at 09:41:33 on 30/05/201. Examining the prefetch folder, correlated the execution time of the EXCEL.EXE file with the timestamp found in the UserAssist registry key. No Internet usage was recorded for this user.

  On examining the index.dat file for Stephen_2 around the times of logon, revealed the user accessed a Website, www.oxid.it at 08:07:22. An inspection of this URL revealed it to be the download site for the password cracking tool Cain and Abel. At 08:20:57, user Stephen_2 accessed the URL: http://hackaday.com/ 2009/09/cain-and-abel-windows-password-recovery-utility. A search of this URL revealed it to be a tutorial on how to crack Windows passwords using Cain and Abel.The ntuser.dat for user Stephen_2 was loaded into Registry Viewer. Examination of the Recent Documents key revealed no documents had been accessed. Examining the UserAssist key, it was found that Stephen_2 executed CAIN.EXE at 09:19:04 0n 30/05/2011. This was correlated with the prefetch file, CAIN.EXE-578E80AC.pf, with the same last access timestamp. Furthermore, CAIN-SETUP-563409AF.pf was accessed at 09:17:25, indicating that the software was installed on the system.

- **UAT:** Since there is no trace of the user John Smyth, no evidence could be found in the file system. A search using the Internet history filters revealed no activity for this user between the dates specified. Using the prefetch files filter, an entry, EXCEL.EXE-53A22446.pf, was found with a last access timestamp of 09:41:33 on 30/05/201, indicating that user John Smyth accessed Microsoft Excel as he was logged in at this time. No other useful information could be found for this user.

  Similarly to FTK, evidence was found in the index.dat file, revealing Stephen_2 accessed www.oxid.it on 30/05/2011 08:07:22 and http://hackaday.com/2009/09/ cain-and-abel-windows-password-recovery-utility at 08:20:57.

  The same prefetch files as those found by FTK, namely CAIN.EXE-578E80AC.pf and CAIN-SETUP-563409AF.pf were also found using the prefetch filter search of UAT.

6 Results and Conclusions

The investigation carried out using the scenario image revealed that both UAT and FTK found most of the information that was required to build a timeline of activity, but there were some differences. FTK was much slower in identifying the files than

UAT and required much more user interaction. FTK had issues reading the event log (.evtx) files and so no proper login/logout timeline could be determined. UAT had problems identifying any activity for the user John Smyth, only discovering that Excel was executed (by examining the prefetch files) by the user. This was due to the tools inability to search through restore points. However, it is the intention of the authors to include this facility in the fully-developed tool.

Overall, UAT demonstrated its usefulness as a rapid analysis tool, working well when profiled with FTK in this simplified scenario. UAT's approach in targeting the essential data for the timeline discovery gives it a distinct advantage over general purpose forensics tools in speed and relevance of reported content.

Further work on this project will involve developing the remote interfaces for other operating systems and mobile devices. More elaborate test scenarios will be used to test the effectiveness of the UAT tool before trialling it in some forensics companies on real-world cases.

Acknowledgments. The authors wish to thank the Department of Informatics in the Institute of Technology Blanchardstown and Rits Ltd for their continuing support. Funding for this project was provided by the IOTI Stand-1 scheme.

References

1. W3Counter, http://www.w3counter.com/globalstats.php (accessed June 14, 2011)
2. Olsson, J., Boldt, M.: Computer Forensics Timeline Visualisation Tool, Digital Investigations. Digital Investigations 6, S78–S87 (2009)
3. Carvey, H.: Windows Forensic Analysis. Syngress. Elsevier Press (2009)
4. Cloppert, M.: Ex-tip: an extensible timeline analysis framework in perl. SANS Inst. (2008)
5. Weber, D.: System Combo Timeline (2007), http://www.cutawaysecurity.com/blog/system-combo-timeline (accessed July 27, 2011)
6. Guðjónsson, K.: Mastering the Super Timeline With log2timeline. SANS Institute (2010)
7. Microsoft Technical Article technet, http://microsoft.com/en-us/library/cc751049.aspx (accessed June 14, 2011)