symmetry

Article

An Abstraction Based Approach for Reconstruction of TimeLine in Digital Forensics

Sandeepak Bhandari * and Vacius Jusas

Software Engineering Department, Kaunas University of Technology, Studentu St. 50, LT-51368 Kaunas, Lithuania; vacius.jusas@ktu.lt

* Correspondence: sandeepak.bhandari@ktu.edu

Received: 11 December 2019; Accepted: 3 January 2020; Published: 6 January 2020

Abstract: Acquiring a clear perspective of events and artefacts that occur over time is a challenging objective to accomplish in digital forensics. Reconstruction of the timeline of events and artefacts, which enables digital investigators to understand the timeline of digital crime and interpret the conclusion in the form of digital evidence, is one of the most paramount and challenging tasks in digital forensics. This challenging task requires the analysis of immense amounts of events because of the explosive growth of the internet, interconnected devices, and innovative technology nowadays. Various approaches have been developed during the last decade, but most of them are not able to handle huge volumes of data, explore evidence, and enhance the understandability of timelines in a competent way to assist the investigator. For this purpose, we introduce a methodology backed by an abstraction concept and forensic tools that can support investigators during the reconstruction, understanding of the timeline of events and artefacts, and interpretation of evidence by tracing the activities performed by users of the typical computer system. The Java programming language is used to implement the proposed methodology, which is object-oriented and follows the symmetry definition in software. Generally, symmetry in software can be viewed as an invariant change that aims to preserve a specific property of the system, namely its structure, behaviour, regularity, similarity, familiarity and uniformity. Similarly, the abstraction-based methodology also permits us to follow the properties of symmetry. For instance, a uniform structure is stipulated for all the sources at the particular level of abstraction, such as the number of fields to be considered to provide the abstract level of timeline. The primary purpose of this approach is to assist with the analysis of the timeline in an optimum way. This paper illustrates the approach and then focuses on conceptual aspects of the methodology. The performed experiment shows that the proposed approach enhanced the analysis of the timeline.

Keywords: digital forensics; digital evidence; timeline reconstruction; events and artefacts

1. Introduction

Digital forensics has emerged as a new area of study during the last two decades due to the explosive growth of the internet, usage of electronic devices, rapid innovation in technology and growing size of storage devices and high rise in digital or computer crimes. Digital forensics' objective is to find and interpret the origin of an event or artefact found on a computer system. Moreover, digital forensics is concerned with the recovery and investigation of digital evidence; when needed, this is often because of a (cyber) crime. The most common reasons for performing digital forensics are attribution, identifying a leak within an organisation, and assessing the possible damage that occurred during a breach. The field of digital forensics is divided up into several subdivisions, depending on the type of the digital device that is the subject of the investigation. These include computer forensics, network forensics, forensic data analysis, and mobile device forensics. Digital investigators examine data and devices to find out as much as possible about a breach or crime that involved digital devices in the form of digital evidence. Carrier [1] classifies digital forensics into three main phases, namely acquisition, analysis and presentation. The aim of the acquisition phase is to preserve the state of a digital system so that evidence can be retrieved and analysed later in controlled conditions. The next phase, i.e., analysis, is to investigate the acquired data to determine evidence of malicious activity. The presentation phase is based entirely on legal rules and regulations, which change depending on the jurisdiction of the country where the digital evidence is located, and it is thus beyond the scope of this paper. Generally, at the acquisition phase, the first step initiated by the investigator is to acquire images of the storage devices of a seized computer. The next step after the acquisition phase involves determining evidence that may involve the exploration of documents and image files of relevance to the investigation.

In a digital forensic investigation, as in a conventional crime scene reconstruction, it is important to be capable of composing a timeline of the file system activities on a computer system. The timeline is used to reconstruct the suspected crime, and it is beneficial in presenting user access to the target system, execution of certain applications, and identifying system and data files which were accessed, modified or deleted during particular periods of interest to the investigation [2]. To show the evidence collected from a seized computer in a law court, the process by which the evidence was derived is required to be clearly shown to abide by accepted procedures and practices [3]. In legal proceedings, this practice is known as the 'chain of custody'. To prove the authenticity of the evidence, comprehensible chains of reasoning are required to be submitted to the court about the scientific methods adopted in the extraction of evidence. Digital investigators face various challenges, such as high speed and volumes of data, explosions of complexity [4], development of standards, privacy-preserving investigations, and rise of anti-forensics techniques when digital or electronic devices are considered to extract evidence from them in the form of the construction of timeline, and this is the objective of our paper. This paper presents an abstraction-based approach to support digital investigators during an investigation via reconstruction of the timeline of events and artefacts which is readable and understandable for investigators and reduces the time needed to collect digital evidence from digital devices.

The remainder of this paper is structured as follows. Section 2 provides a review of related work. Section 3 discusses the methodology for this research. Section 4 provides implementation, results and its discussion from the use of the developed methodology. At last, we finish with conclusions in Section 5.

2. Literature Review

Prasad & Satish [5] defined digital forensics as the procedure of identifying, collecting, preserving, analysing and presenting digital evidence in a way that is legally accepted by the court. To collect the digital evidence from digital devices efficiently, effectively and within a limited period of time, reconstruction of timeline required. Carvey [6] defined timeline as "a means of identifying or linking a sequence of events in a manner that is easy for people such as incident responders to visualize and understand". Harrell [7] defined "timeline analysis is a great technique to determine the activity that occurred on a system at a particular period of time on a system". So, "creating a timeline of the various events that occurred during an incident is one of the key tasks performed by the digital forensic practitioner" [8]. Timeline analysis should be a key component to any investigation as the timing of events is nearly always relevant. The primary source of timeline information is the file system metadata. File systems' track different time stamps and have nuances that must be considered when performing forensic analysis [9].

Inglot & Liu [10] specified that there are basically two approaches for analysis of the timeline. Firstly, there are applications that have been specifically created for the analysis of the timeline, and they focus on visually presenting the timeline, such as Cyber Forensics Time Lab (CFTL), Zeitline, Encase, Sleuth kit, Forensic toolkit and many others. Secondly, there are a combination of command-line tools and spreadsheet applications which are labour intensive, such as Log2timeline and excel together.

Guðjónsson [11] designed a tool to extract timestamps from various files found on a typical computer system and aggregate them. The tool Log2timeline is a part of a Python-based backend engine plaso. The purpose of plaso was to have the timestamps in a single place for computer forensic analysis. Such a timeline sometimes is called a Super Timeline. Sitompul, Handoko, & Rahmat [12] state that to have a clear view of events that occurred during a time period is challenging to achieve in digital investigation. Event reconstruction, which allows a digital investigator to understand the timeline of a crime, is one of the paramount steps of the digital investigation process [13]. This complex task requires exploration of a large number of events due to the innovation in technology frequently, a heterogeneous, huge quantity of data, and manually-performed event reconstruction process, which is inefficient and expensive [14]. Bang et al. [15] discussed how the creation time, last written time, and last accessed time of a file or folder are important factors that can indicate events that have affected a computer system. They analysed changes in the time information of files and folders for different operations of the FAT and NTFS file systems and attempts to reconstruct the user's actions. Further, they demonstrate the use of time information for digital evidence analysis by presenting a case study. Chabot et al. [16] identified two major challenges (heterogeneity and volume of data) with event reconstruction. To solve these two challenges, they present an approach supported by theoretical concepts that can assist investigators through the whole process, including the construction and interpretation of the events describing the case. The proposed approach is based on a model which integrates knowledge of experts from the digital forensic fields and software development to allow a semantically rich representation of events related to incidents.

Hargreaves & Patterson [17] considered a possible use of the tool Log2timeline for timeline reconstruction; however, they decided to build their own Python-based prototype. The other contribution of the research work was a framework that allows analyses to be written that can produce a high-level event based on the presence of one or more low-level events. The shortcoming of the approach was that the analysers, which were oriented to particular events, have to be written in advance. Brady, Overill & Keppens [18] proposed the use of an ontology, the Digital Evidence Semantic Ontology (DESO), that allows an examiner to quickly discover what artefacts may be available on a device before time-consuming processes are commenced. The DESO is built on the ideas of the Gene Ontology (GO). The general principle behind DESO is two-fold: (1) examiners use some form of classification or tagging system that allowed examiners to readily assess what artefacts were available; (2) once artefacts have been extracted from various sources, DESO provides the means to compare them. The main idea of DESO is to enable comparison of the artefacts extracted from different sources. Brady & Overill [19] continued the development of the ontology DESO. DESO's primary purpose is to act as a repository and a classifier of digital evidence artefacts to allow correlation of extracted data from heterogeneous sources. Investigative objectives are set and fulfilled by asking simple questions based on who, what, when, where, why and how? ('5WH'). Only the "What" subclass has been detailed. The classes "why" and "how" were not discussed at all. An implemented body of DESO was not revealed. It remained behind the scenes. Only the ideas were presented.

Debinski, Breitinger & Mohan [20] state that event reconstruction is a fundamental step for investigators to understand a case where a prominent tool is Log2timeline to generates timelines. While these timelines provide great evidence and assist to understand a case, they are complex and require tools as well as training scenarios. Moreover, they also state some of the major limitations of Log2timeline, such as the fact that there is no easy-to-use tool that beginners/investigators can use to analyse a generated timeline and no free training material that also allows practitioners to learn and improve their familiarity with the Log2timeline as well as visualization tools. To support investigators, the authors developed Timeline2GUI a standalone tool written in Python that supports the analysis of the CSV timeline (output from Log2timeline). The tool is similar to the commonly-used Excel sheet to allow an easy transition for practitioners. In addition, they developed three training cases that are freely available and can be used to improve investigator's timeline analysis skills by either using Timeline2GUI, the Excel sheet or any other tool. The performance of Timeline2GUI can be enhanced by speeding up the process, such as by removing irrelevant fields or combining fields in future work. Soltani, Seno & Yazdi [21] proposed an event reconstruction framework which determines whether an application has been run on a compromised system. The proposed framework has constructed the signature or the TPFSM-A (temporal pattern of file system modification of the application) and the TPFSM-D (temporal pattern of file system modification of the hard disk). Moreover, the framework has presented a distance metric which is used to calculate the distance between the signature of the application and TPFSM-D of the hard disk. Finally, the decision engine of the framework has used the calculated distance to decide whether the application has been run on the compromised system.

Zhao and Coplien [22] stated that the success of symmetry applications in various scientific disciplines motivated them to explore the concept of symmetry in software. Further, the authors explained object-oriented language constructs use the notion of symmetry and explored symmetry in an object-oriented language and also provide some examples of symmetry outside of object-oriented programming to show that symmetry considerations broaden beyond object-orientation to other areas of software design.

Jensen [23] stated that a class is the classification of objects. Classifications establish the class member invariance such that the description of the class is true for all the objects of the class. This view of the class can be explained as symmetry. A class enables the change of the object, but the change must respect the structure and behaviour stipulated by the class.

The literature review shows that numerous tools and techniques have been developed to support digital investigators handling timeline analysis available, but none of them are able to solve the complexity of construction of the timeline of events and artefacts effectively in digital forensics. Some of the primary reasons which are responsible for the ineffective reconstruction of timelines include the immense amount of data, heterogeneity of data, rapid innovation of technology, accelerated growth of the internet, and integrity of the timeline. The purpose of the paper is to develop and implement a methodology for the reconstruction of the timeline of events and artefacts which is able to regenerate the timeline completely and understandably based on the level of abstraction, and is also able to accomplish this even with the existence of above-mentioned issues and can follow the symmetry properties by stipulating specific conditions at each level of abstraction to compose the relevant timeline.

3. Methodology

Various problems are highlighted in the literature review section stem from a huge volume of data when the timeline is extracted from a disk image, particularly with the 'Super Timeline' approach. Our research work is based on the data provided by Log2timeline in the form of plaso file, which is sorted out with the Psort tool as shown in Figure 1.

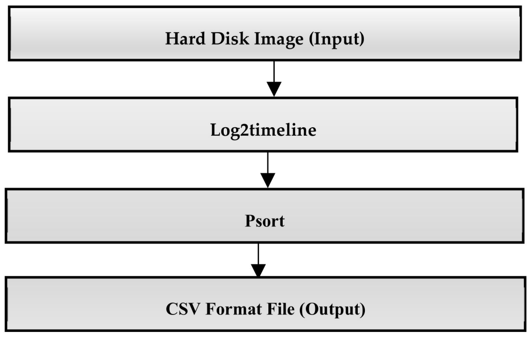


Figure 1. The formation of the timeline.

Additionally, Psort allows converting the plaso file into common file formats such as CSV. The CSV timeline contains 17 fixed fields as listed in Table 1 (SANS, 2011); then, data from the CSV file is imported into an Excel sheet. One of the most common problems in the timeline extracted by Log2timeline is a repetition of the same time unit for numerous times. For instance, time unit "11:49:53" in the field "time" repeated multiple times correspond to various sources of data such as "WEBHIST" and "LNK" in the field "source", which shows that the occurrence of the same event has raised the reflection in different logs and different artefacts, as shown in Table 2. Such scenarios formulate timeline large, complex and difficult to analyse for digital forensic practitioners. So, we extract such scenarios from the timeline and replace using appropriate methods (such as one event per time unit to eliminate duplication) to reconstruct a simple, compact, recognisable and structured timeline for digital forensics practitioners.

Table 1. Fields in the CSV File by Psort tool.

Field	Description
date	date of when the event occurred
time	time of when the event occurred
timezone	timezone that was used to call the tool with
MACB	Modification, Access, Creation, and Birth
source	source short name such as registry entries are REG
sourcetype	description of the source
type	timestamp type such as last accessed or last written
user	what user name is associated with event if any
host	what hostname is associated with entry is there is one
short	this contains a short description field where text is stored
desc	this is where majority of the information that is parsed is stored
version	gives the version number of the timestamp
inode	gives the inode number of the file being parsed
notes	additional storage location for information for some input modules
format	input module which was used to parse
extra	Parsed information that are joined together and stored here. All these pieces of information make up the super timeline that Log2Timeline creates.

We present a methodology by which it is viable to present the timeline of events and artefacts relevant by using the concept of abstraction to extend the timeline of events and artefacts at four different abstraction levels, i.e., events: High level; events: Low level; artefact location: High level; and artefact location: Low level, as shown in Figure 2 to generate timeline readable and understandable for digital investigators. The ideas behind the breakdown of the timeline provided by Log2timeline into four levels of abstraction is given to present the different kinds of information, and a different structure should be specified for each level along with distinctive levels of details of information in order to reduce complexity of timeline, omitting unwanted details, enforcing the correctness of timeline and presenting only information that will be helpful to recognise and understand particular actions executed by users by analysing different sources and fields.

Table 2. Timeline with repetition (duplicity) of the same time unit.

Date	Time	MACB	Source	Sourcetype	Type
19 November 2017	11:49:53	.A..	WEBHIST	Firefox History	Last Visited Time
19 November 2017	11:49:53	.A.B	LNK	Windows Shortcut	Creation Time; Last Access Time
19 November 2017	11:49:53	MA.B	LNK	Windows Shortcut	Content Modification Time; Creation Time; Last Access Time
19 November 2017	11:49:53	.A.B	LNK	Windows Shortcut	Creation Time; Last Access Time
19 November 2017	11:49:53	.A..	WEBHIST	Firefox History	Last Visited Time
19 November 2017	11:49:53	MA.B	LNK	Windows Shortcut	Content Modification Time; Creation Time; Last Access Time
19 November 2017	11:49:53	MA.B	LNK	Windows Shortcut	Content Modification Time; Creation Time; Last Access Time

In digital forensics, an event can be characterised as an activity performed by users utilising digital devices. On the other side, "artefact" currently does not have a formal definition within the domain of cyber/digital forensics, resulting in a lack of standardised reporting and linguistic understanding between professionals. Generally, the artefact can be defined as a piece of data that may or may not be relevant to the investigation/response. Examples include registry keys, files, timestamps, and event logs. In other words, artefacts can be defined as something observed in a scientific investigation or experiment that is not naturally present but occurs as a result of the preparative or investigative procedure.

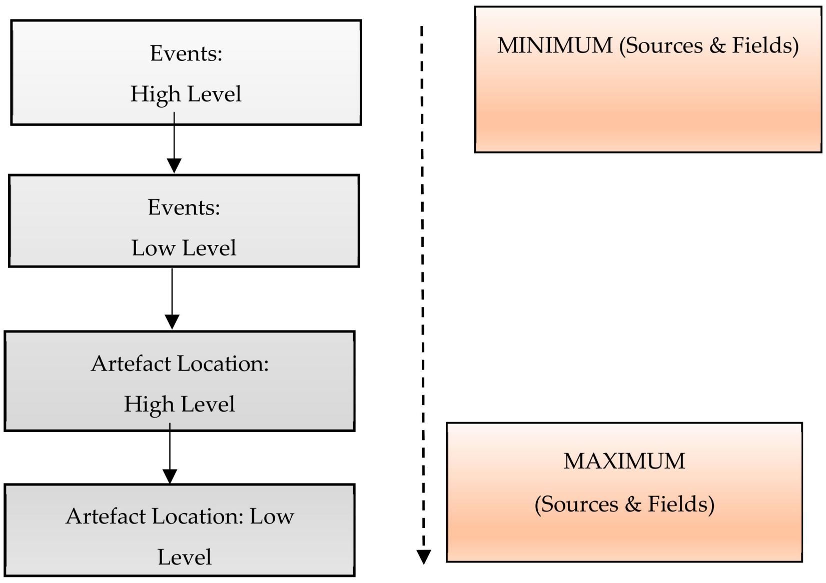


Figure 2. Four abstraction layers of events and artefact.

The selected methodology in the research involves the development of four different modules, with each module corresponding to one of the levels of abstraction of the timeline of events and artefacts. At each level of the methodology, different types of sources, parameters and fields are considered to compose improved structure and representation of the timeline. At events: High level information related to activities carried out by users such as the creation of files, access of web pages and downloads of information in the form of files of various formats (jpeg, pdf and docx) is provided. Such information with higher levels of abstraction is extracted by analysing six different sources, "LNK", "LOG", "META", "OLECF", "PE" and "WEBHIST", and six distinct fields, namely "date", "time", "source", "short", "visit" and "reference", providing required information to recognise particular activities. Moreover, the number of sources and fields considered for analysing depends upon what kind of information it is and its level of detail required at the particular level of the timeline.

To achieve detailed information, various types of operations are performed related to the files, namely modification, access, changes and birth, and web surfing activities that address of specific web pages accessed and the download of files by user. The second level of abstraction events: Low level is reconstructed with additional details as compared to previous events: High level so an additional number of sources and fields are required for analysis. At this level, nine different sources and seven fields with the extension of three new sources, including "FILE", "REG" and "RECBIN", and a new field, "extra", respectively are examined to reconstruct the timeline. The first two levels of abstraction of the methodology can provide the least information related to what type of actions are performed by users, but it is not adequate to reconstruct the complete timeline. To gain a clear view of the timeline, additional information is required. This includes a list of all .exe files run by the user, applications and files that are frequently accessed, authors of files, list of files. Namely, .automaticDestinations-ms provides information about the application used that is pinned on a user's taskbar, timestamps, and the paths to items (documents, webpages, images, etc.) recently accessed by a program pinned on a user's taskbar, while .msp provides updating information related to Windows operating system and other Microsoft programs, and the .msi file contains installation information for a particular installer, such as files pending to be installed and installation locations.

Moreover, detailed information is provided related to various activities performed on the internet. For instance, which application program (web browser) is used to and how (LINK–user clicked a link, GENERATED–selected an entry from the list, RELOAD–user reloaded the page and TYPED–user typed the URL in the URL bar) each specific web page is accessed. This detailed information is provided at artefact: High level, and is reconstructed by examining the nine different sources and ten fields with the addition of three new fields "MACB", "sourcetype" and "desc "(description) with a lower level of abstraction. At different levels of abstraction, we include relevant information and removing unwanted details to reconstruct the relevant timeline by considering a different number of sources and fields. At the end, we reconstruct a timeline with the lowest abstraction level among all levels of abstraction of artefact: Low level by analysing all available nine sources and seventeen fields with the addition of 9 fields, namely "timezone", "type", "user", "host", "version" "filename", "inode", "notes" and "format" to provide complete detailed information related to all activities performed by user.

The proposed methodology is composed of four different abstraction levels of events and artefacts. Each level provides different information related to various activities executed by the user, with different levels of abstraction of information, beginning from minimum to maximum details, to reconstruct the relevant and recognisable timeline for the better understandability of the digital practitioner.

4. Experiment and Discussion

In this section, we carried out the experiments on about 150 Gigabytes of hard disk image data with the NTFS file system in a Windows-based system to demonstrate the capability of the proposed event reconstruction abstraction based novel methodology. The experiments set consists of two main categories of operations executed by the user.

1. Online operations: It includes all types of actions performed on the internet such as access to web pages, downloads of information and compose an email by using any browser.
2. Offline operations: It includes all types of actions performed on the typical computer system such as creation, modification, access, rename and copy of the file by using various applications such as word pad, Notepad and many others. The experiments were conducted by running each application sequentially and together parallelly.

4.1. Case Study of Web History

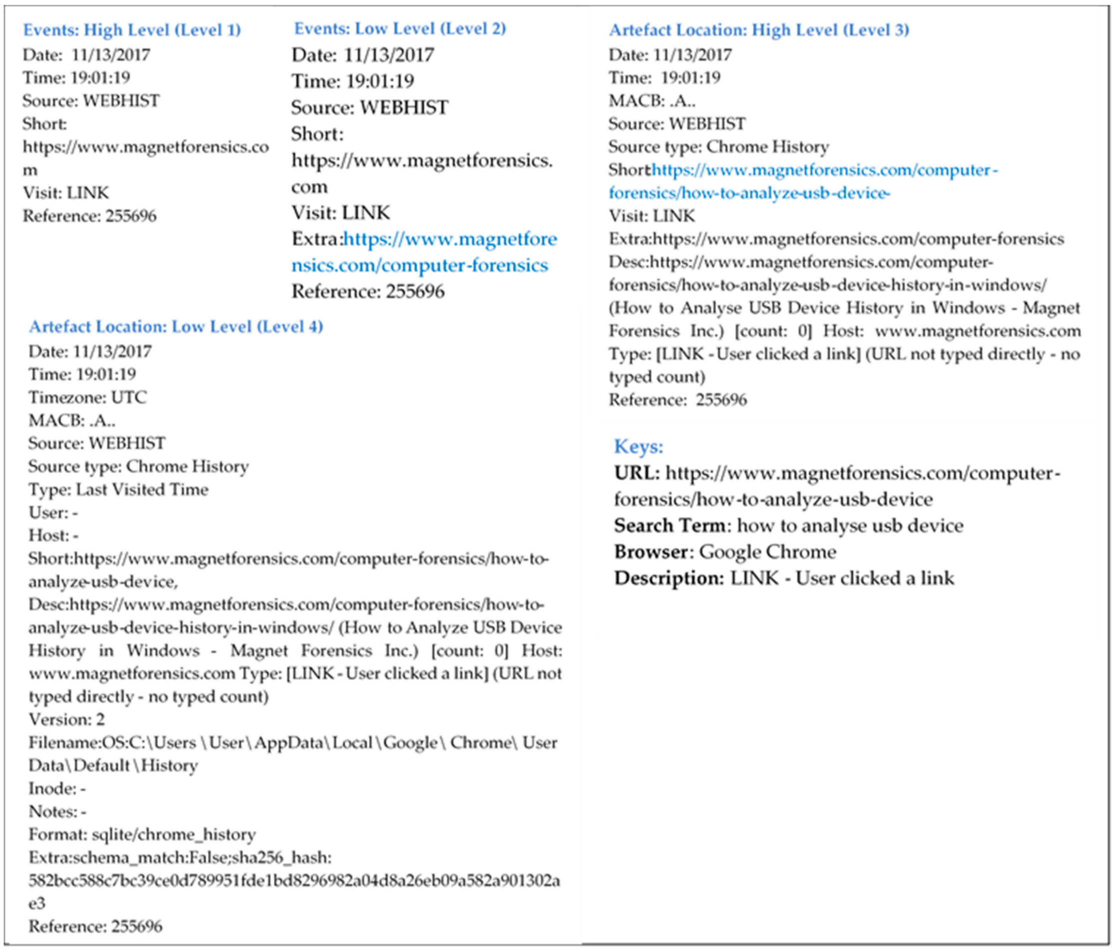
The web history case study presents all types of operations performed on the web by using three different browsers, namely Internet Explorer version 11, Google Chrome version 74.0.3729.169 and Mozilla Firefox version 66.0.5 for the implementation of the first phase of experiments. We conducted multiple trials of various kinds of experiments by running these browsers with a blank home page setting to various levels of depth of numerous websites, surfing and downloading the different files from different websites. The description of the experimental scenario is given in Table 3.

Table 3. Scenario summary (web history case study).

Experiment	Scenario Summary
1.	Flat or Blank execution of Internet Explorer, Google Chrome and Mozilla Firefox, i.e., running without opening/surfing any webpage by typing 'about blank' in the URL address text box. The experiment was conducted by launching all three browsers and then instantly closing their windows. The purpose of flat execution of IE was to check which of the system and temporary files are accessed by browsers when no webpage is loaded.
2.	Launched all three browsers parallelly and sequentially and opened Kaunas University of Technology website 'https://ktu.edu'. Checked personal email and then signed out followed by closing the respected browsers window.
3.	Launched browser and opened various sessions for random surfing of academic, commercial, social and news websites like "www.linkedin.com", "www.forensicswiki.org", "www.sciencedirect.com", "www.elsevier.com", "https://www.seb.lt/", "www.researchgate.net" and many more. Further, more perform some activities such as download required files and entering comments to some portal and closed browser after a session of different times.
4.	Launched browser and open "https://mail.google.com/" websites to check the emails. At this time, not only the emails are checked but also attachments were downloaded and uploaded, then subsequently opened through their associated application programs like MS-Word, Adobe Acrobat Reader, Excel and PowerPoint.

The developed methodology consists of four levels of abstraction for the reconstruction of the timeline. So, there are four different timelines in the form of results. As we mentioned above, we used three different browsers to implement the first phase, but we only show the timeline of specific events and artefacts of the Google Chrome browser due to limited space.

Figure 3 consists of four different timelines presenting outcomes of the web history study case corresponding to the four levels of the abstraction-based approach. Each level depicts different information along with the level of details related to particular activity performed by the user on the web. The first two levels, i.e., events: High and low level, provide the minimum information and overview of what kind of activity is performed. The other two levels, i.e., artefact location high and low levels, provide complete information about a particular activity. For instance, which browser is used by the user to access the particular web page, the complete URL of web page, how the web page is accessed, i.e., by clicking a link, reloading the page or typing the URL in the URL bar and the information search by user as shown in keys section in Figure 3. The main objective is to provide key information in a compact form and is very supportive for the digital investigator to understand the timeline and to retrieve the relevant conclusion.



Events: High Level (Level 1)
Date: 11/13/2017
Time: 19:01:19
Source: WEBHIST
Short:
https://www.magnetforensics.co
m
Visit: LINK
Reference: 255696

Events: Low Level (Level 2)
Date: 11/13/2017
Time: 19:01:19
Source: WEBHIST
Short:
https://www.magnetforensics.
com
Visit: LINK
Extra:https://www.magnetfore
nsics.com/computer-forensics
Reference: 255696

Artefact Location: High Level (Level 3)
Date: 11/13/2017
Time: 19:01:19
MACB: .A..
Source: WEBHIST
Source type: Chrome History
Short:https://www.magnetforensics.com/computer-
forensics/how-to-analyze-usb-device-
Visit: LINK
Extra:https://www.magnetforensics.com/computer-forensics
Desc:https://www.magnetforensics.com/computer-
forensics/how-to-analyze-usb-device-history-in-windows/
(How to Analyse USB Device History in Windows - Magnet
Forensics Inc.) [count: 0] Host: www.magnetforensics.com
Type: [LINK - User clicked a link] (URL not typed directly - no
typed count)
Reference: 255696

Artefact Location: Low Level (Level 4)
Date: 11/13/2017
Time: 19:01:19
Timezone: UTC
MACB: .A..
Source: WEBHIST
Source type: Chrome History
Type: Last Visited Time
User: -
Host: -
Short:https://www.magnetforensics.com/computer-forensics/how-to-
analyze-usb-device,
Desc:https://www.magnetforensics.com/computer-forensics/how-to-
analyze-usb-device-history-in-windows/ (How to Analyze USB Device
History in Windows - Magnet Forensics Inc.) [count: 0] Host:
www.magnetforensics.com Type: [LINK - User clicked a link] (URL not
typed directly - no typed count)
Version: 2
Filename:OS:C:\Users \User\AppData\Local\Google\ Chrome\ User
Data\Default\History
Inode: -
Notes: -
Format: sqlite/chrome_history
Extra:schema_match:False;sha256_hash:
582bcc588c7bc39ce0d789951fde1bd8296982a04d8a26eb09a582a901302a
e3
Reference: 255696

Keys:
URL: https://www.magnetforensics.com/computer-
forensics/how-to-analyze-usb-device
Search Term: how to analyse usb device
Browser: Google Chrome
Description: LINK - User clicked a link

Figure 3. The output of Web history study case.

4.2. Case Study of Multiple Application Programs

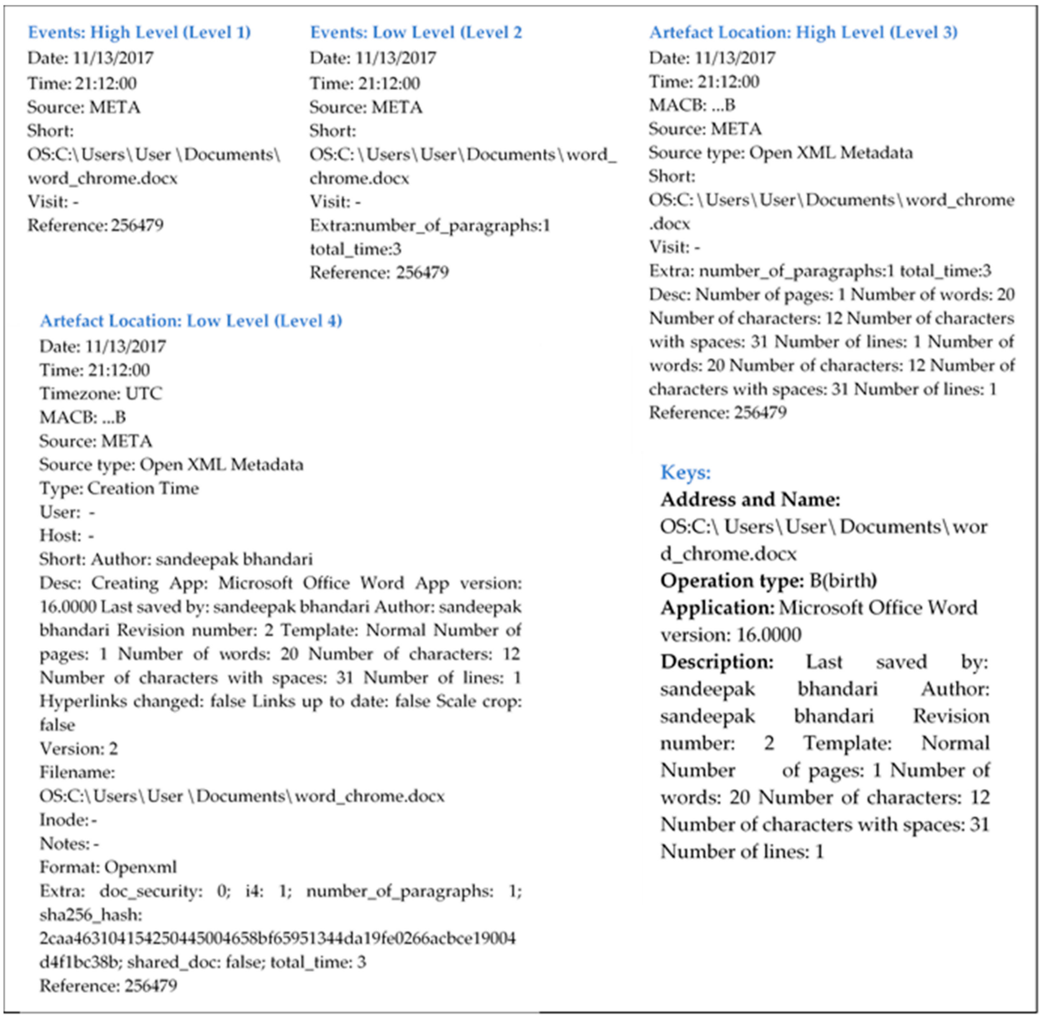
For the second phase of the experiment set, we executed seven different application programs. These included Microsoft Office Word, Microsoft Office Excel, WordPad, and Notepad, and all browsers were also among these applications. Similar to phase one, we conducted multiple trials of experiments from a flat run to executing multiple files in these applications. The experiments were conducted by running each application sequentially and together parallelly. A brief summary of the experimental scenario is given in Table 4.

Table 4. Scenario summary (execution of Multiple application programs case study).

Experiment	Scenario Summary
1.	The following application programs were independently loaded, without opening a single file in these applications, in the order: Internet Explorer, Mozilla Firefox, Google Chrome, WordPad, Notepad, MS-Word, Excel.
2.	The above-mentioned applications were individually run and only a single file was opened for each application followed by closing the application program.
3.	The above-mentioned group of application programs was discretely executed and multiple data files were accessed for each individual application. This time the attempts were made to open the files having an incompatible format to the corresponding application to observe the file system activity patterns of application programs for failed attempts during the loading of inconsistent files.
4.	Multiple applications are executed parallelly and multiple files were opened for each application. The timings of launching these applications were noted down so that file system activity data sets can be linked to the corresponding application programs.
5.	Execute some operations such as modified, accessed, Changed, Birth (file creation time) rename, copy and delete for above mention applications namely WordPad, Notepad, MS-Word, Excel.
6.	Insert USB and create a directory in USB and copy some text files from the computer system to created directory and vice versa. Perform some operations such as modified, accessed, Changed and Birth (file creation time).

In this case, we show the timeline of events and artefacts related to creation (birth) of the .docx file by a user. Figure 4 consists of four different timelines corresponding to the four levels of abstraction based methodology. The objective of this part of the experiment is to provide relevant information such as what kind of operation is performed, an application used by the user to execute that particular operation, and a short description of the operation (address and name of the file), as shown in the key section in Figure 4.

We can observe (see Figures 3 and 4) that our proposed abstraction-based methodology can enhance the relevance of the timeline by reconstructing into four levels of the timeline. Each level of the timeline shows the different depth of information from general to specific to reduce complexity, omitting irrelevant and extracting relevant information in order to provide a clear and understandable view of the timeline to assist digital investigators or practitioners.



Events: High Level (Level 1)
Date: 11/13/2017
Time: 21:12:00
Source: META
Short:
OS:C:\Users\User \Documents\word_chrome.docx
Visit: -
Reference: 256479

Events: Low Level (Level 2
Date: 11/13/2017
Time: 21:12:00
Source: META
Short:
OS:C: \Users\User\Documents\word_chrome.docx
Visit: -
Extra:number_of_paragraphs:1 total_time:3
Reference: 256479

Artefact Location: High Level (Level 3)
Date: 11/13/2017
Time: 21:12:00
MACB: ...B
Source: META
Source type: Open XML Metadata
Short:
OS:C: \Users\User\Documents\word_chrome.docx
Visit: -
Extra: number_of_paragraphs:1 total_time:3
Desc: Number of pages: 1 Number of words: 20 Number of characters: 12 Number of characters with spaces: 31 Number of lines: 1 Number of words: 20 Number of characters: 12 Number of characters with spaces: 31 Number of lines: 1
Reference: 256479

Artefact Location: Low Level (Level 4)
Date: 11/13/2017
Time: 21:12:00
Timezone: UTC
MACB: ...B
Source: META
Source type: Open XML Metadata
Type: Creation Time
User: -
Host: -
Short: Author: sandeepak bhandari
Desc: Creating App: Microsoft Office Word App version: 16.0000 Last saved by: sandeepak bhandari Author: sandeepak bhandari Revision number: 2 Template: Normal Number of pages: 1 Number of words: 20 Number of characters: 12 Number of characters with spaces: 31 Number of lines: 1 Hyperlinks changed: false Links up to date: false Scale crop: false
Version: 2
Filename:
OS:C:\Users\User \Documents\word_chrome.docx
Inode: -
Notes: -
Format: Openxml
Extra: doc_security: 0; i4: 1; number_of_paragraphs: 1; sha256_hash: 2caa463104154250445004658bf65951344da19fe0266acbce19004d4f1bc38b; shared_doc: false; total_time: 3
Reference: 256479

Keys:
Address and Name: OS:C:\ Users\User\ Documents\word_chrome.docx
Operation type: B(birth)
Application: Microsoft Office Word version: 16.0000
Description: Last saved by: sandeepak bhandari Author: sandeepak bhandari Revision number: 2 Template: Normal Number of pages: 1 Number of words: 20 Number of characters: 12 Number of characters with spaces: 31 Number of lines: 1

Figure 4. The output of multiple application programs study case.

5. Conclusions

In digital forensics, one of the primary challenges is the reconstruction and analysis of the timeline of events and artefacts to interpret digital evidence due to the immense quantity and diversity of data. Scientific and technical literature studies show that various approaches have been developed to figure it out, but most of them are not capable of handling this issue precisely.

In this paper, we present a flexible and novel approach based on four different levels of abstraction of timeline of events and artefacts such as events: High level (new entries and web surfing), events: Low level (web surfing, actions of modifying), artefact location: High level (include all .exe files) and artefact location: Low level. The contributions of the proposed methodology are at each level of timeline; a unique structure is followed which gives information related to a particular action performed by the user. Different parameters, fields, and criteria are considered for a noise-free timeline (such as one event per time unit), and distinct relevant patterns of data are considered to exclude the irrelevant data to reconstruct the compact and efficient timeline with a distinct level of abstraction of detail in order to show the structured timeline, which assists the investigator in understanding and analysing the timeline conveniently and rapidly.

Various experiments are executed to implement the proposed approach and test the outcome. The results show that the abstraction-based approach is capable of the reconstruction of the timeline of events and artefacts efficiently, regardless of quantity and heterogeneity of data, as compared existing techniques. Moreover, in this article, only windows-based operating system operations and results are discussed. However, this methodology can be implemented to other operating systems as the Lo2timeline command based tool has the capability to compile other operating systems and generate output in the form of the L2TCSV format, which is used as an input to the methodology. In the future, we will implement a proposed methodology for the android operating system, and based on the results, an abstraction-based ontology will be developed for digital forensics' future work.

Author Contributions: Conceptualization, methodology, V.J. and S.B.; formal analysis, V.J.; writing—original draft preparation, S.B.; writing—review and editing, S.B.; supervision, V.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Carrier, B. Open Source Digital Forensics Tools. Available online: http://www.digitalevidence.org/papers/opensrc_legal.pdf (accessed on 21 May 2019).
2. Olajide, F.; Savage, N.; Ndzi, D.; Al-Sinani, H. Forensic Live Response and Event Reconstruction Methods in Linux Systems. Available online: http://www.cms.livjm.ac.uk/pgnet2009/Proceedings/Papers/2009001.pdf (accessed on 25 May 2019).
3. Thomas, D.S.; Forcht, K.A. Legal methods of using computer forensics techniques for computer crime analysis and investigation. *Issues Inf. Syst.* **2004**, *5*, 692–698.
4. Lillis, D.; Becker, B.A.; Scanlon, M. Current Challenges and Future Research Areas for Digital Forensic Investigation. In Proceedings of the 11th ADFSL Conference on Digital Forensics, Security and Law, Daytona Beach, FL, USA, 13 April 2016; pp. 9–20.
5. Prasad, M.A.R.; Satish, Y.N. Reconstruction of Events in Digital Forensics. *Int. J. Eng. Trends Technol.* **2013**, *4*, 3460–3467. [CrossRef]
6. Carvey, H. *Windows Forensic Analysis DVD Toolkit 2E*; Syngress: Burlington, MA, USA, 2009.
7. Harrell, C. What's a Timeline. Available online: http://journeyintoir.blogspot.com/2011/09/whats-timeline.html (accessed on 28 May 2019).
8. Esposito, S.; Peterso, G. Creating Super Timelines in Windows Investigations. In Proceedings of the 9th International Conference on Digital Forensics, Orlando, FL, USA, 28–30 January 2013; pp. 135–144.
9. James, J.I.; Gladyshev, P. Automated inference of past action instances in digital investigations. *Int. J. Inf. Secur.* **2015**, *14*, 249–261. [CrossRef]
10. Inglot, B.; Liu, L. Enhanced Timeline Analysis for Digital Forensic Investigations. *Inf. Secur. J. Glob. Perspect.* **2014**, *23*, 32–44. [CrossRef]
11. Guðjónsson, K. Mastering the Super Timeline Who Am I ? Available online: https://digital-forensics.sans.org/summit-archives/2010/eu-digital-forensics-incident-response-summit-kristinngudjonsson-mastering-the-super-timeline.pdf (accessed on 22 May 2019).
12. Sitompul, O.S.; Handoko, A.; Rahmat, R.F. File Reconstruction in Digital Forensic. *TELKOMNIKA Indones. J. Electr. Eng.* **2018**, *16*, 776–794. [CrossRef]
13. Cho, G.S. A computer forensic method for detecting timestamp forgery in NTFS. *Comput. Secur.* **2013**, *34*, 36–46. [CrossRef]
14. Kalber, S.; Dewald, A.; Freiling, F.C. Forensic application-fingerprinting based on file system metadata. In Proceedings of the Seventh International Conference on IT Security Incident Management and IT Forensics, Nuremberg, Germany, 12–14 March 2013; pp. 98–112.
15. Bang, J.; Yoo, B.; Kim, J.; Lee, S. Analysis of time information for digital investigation. In Proceedings of the Fifth International Joint Conference on INC, IMS and IDC, Seoul, Korea, 25–27 August 2009; pp. 1858–1864.
16. Chabot, Y.; Bertaux, A.; Nicolle, C.; Kechadi, M.T. A complete formalized knowledge representation model for advanced digital forensics timeline analysis. *Digit. Investig.* **2014**, *11*, 95–105. [CrossRef]
17. Hargreaves, C.; Patterson, J. An automated timeline reconstruction approach for digital forensic investigations. *Digit. Investig.* **2012**, *9*, S69–S79. [CrossRef]
18. Brady, O.; Overill, R. DESO: Addressing volume and variety in large scale criminal cases. *Digit. Investig.* **2015**, *88*, 72–825. [CrossRef]

19. Brady, O.; Overill, R.; Keppens, J. Addressing the increasing volume and variety of digital evidence using an ontology. In Proceedings of the 2014 IEEE Joint Intelligence and Security Informatics Conference, Hague, The Netherlands, 24–26 September 2014; pp. 176–183.
20. Debinski, M.; Breitinger, F.; Mohan, P. Timeline2GUI: A Log2timeline CSV parser and training scenarios. *Digit. Investig.* **2019**, *28*, 34–43. [CrossRef]
21. Soltani, S.; Seno, S.A.H.; Yazdi, H.S. Event reconstruction using temporal pattern of file system modification. *IET Inf. Secur.* **2019**, *13*, 201–212. [CrossRef]
22. Zhao, L.; Coplien, J.O. Understanding symmetry in object-oriented languages. *J. Obect Technol.* **2003**, *2*, 5. [CrossRef]
23. Jensen, W.B. Classification, symmetry and the periodic table. In *Symmetry: Unifying Human Understanding*; Hargittai, I., Ed.; Pergamon Press: Oxford, UK, 1986; ISBN 0-08-033986-7.