ABSTRACT

Computer forensics is the collection and analysis of data from the computers and different digital devices to obtain evidence. In order to describe the circumstances of the particular incident investigators need to build up the timeline events from the past. One of the existing works on digital forensics timeline concentrates on retrieving times from disk image to a respective timeline. However, this method of extracting data can yield numerous events like file modifications or a Registry key update for a single disk. In the proposed system, a prototype has been implemented using Java which reconstructs the sequence of events for data in browsing history. The raw data extracted at the crime scene from digital devices is in the form of a .txt file. These .txt files are then stored into MySQL database in specified fields for retrieving and analyzing purpose in future. The investigator has an option to search desired data from the database for further analysis by obtaining timelines.
# TABLE OF CONTENTS

Abstract ........................................................................................................................................ ii

Table of Contents ......................................................................................................................... iv

List of Figures ............................................................................................................................... v

List of Tables ................................................................................................................................. vi

1. Background and Rationale ........................................................................................................1

   1.1 Introduction .........................................................................................................................1

      1.1.1 Computer Forensics ..................................................................................................1

      1.1.2 Mobile Forensics ......................................................................................................1

      1.1.3 Network Forensics ....................................................................................................2

      1.1.4 Database Forensics ..................................................................................................2

   1.2 Goals of Digital Forensics .................................................................................................2

      1.2.1 Identifying the Digital Evidence .............................................................................3

      1.2.2 Preserving the Digital Evidence .............................................................................3

      1.2.3 Analyzing the Digital Evidence ..............................................................................3

      1.2.4 Forensic Tool Kit .....................................................................................................4
3.4 Use Case Diagram ................................................................. 20

4 Functionality of Application .................................................. 21

4.1 User Interface for Automatic Event Sequence Reconstruction Prototype... 23

4.1.1 Main Form for Automatic Event Sequence generation ................. 23

4.1.2 Uploading the Extracted Data ............................................. 24

4.1.3 View History Of Data Stored in Database ............................. 25

4.1.4 Analyzing The Data From Database ..................................... 26

4.1.5 Searching Data ................................................................. 27

5 Testing and Evaluation ........................................................... 29

5.1 Test Case ........................................................................ 29

6 Conclusion ........................................................................... 35

7 Future Work ........................................................................ 36

Bibliography ........................................................................... 37

Appendix .................................................................................. 38
LIST OF FIGURES

Figure 1.1 Computer Collected as Digital Evidence .................................................6

Figure 3.1 Proposed System Architecture ..............................................................13

Figure 3.2 Flow Chart Diagram of Extracting Data from Computer .........................14

Figure 3.3 Screen Shot of History.txt file ..............................................................15

Figure 3.4 Flow Chart Diagram of Data in Analyzer ..............................................18

Figure 3.5 Flow Chart Diagram For Proposed System .........................................19

Figure 3.6 Showing Input Flow Diagram Of .txt File ..........................................20

Figure 3.7 Use Case .............................................................................................21

Figure 4.1 Sample .txt Source File .....................................................................22

Figure 4.2 Login Page For Prototype ..................................................................23

Figure 4.2 Available Options In Prototype .........................................................24

Figure 4.4 Uploading Data ..................................................................................25

Figure 4.5 Pop-up Window Viewing History Of Stored Database .......................26

Figure 4.6 Analyzing Window asking for Group ID’s .........................................27

Figure 4.7 Pop-up Window for Filtering Data asking Keyword ...........................28
Figure 5.1 Uploading Data From .txt file .................................................................30
Figure 5.2 Viewing Web History From Database .......................................................30
Figure 5.3 Showing Matched and Unmatched Records..............................................31
Figure 5.4 Searching History By Keyword.................................................................32
Figure 5.5 Showing the Sequence of Events Occurred in Particular date ..............33
LIST OF TABLES

Table 1.1 MAC times For Two Files ................................................................................8

Table 1.2 Timeline View ..................................................................................................9

Table 3.1 Sample History Database Table ......................................................................17
1. BACKGROUND AND RATIONALE

1.1 Introduction

Computers became an integral part in everyone’s life these days and many important jobs are done using them like paying bills, financial transactions, storing personal data, browsing internet and e-mails etc. At the same time there have been increasing number of cyber-crimes like phishing, hacking and illegal activities in these electronic media [14]. Forensics is the branch of education which involves the study and investigation of crimes scientifically. Computer forensics is also called as Digital Forensics involves the investigation of digital devices recovered from the crime scene. It includes collection, preserving and analyzing the digital devices and presenting the evidence to the court of law. This Digital forensics has a wide variety of branches of forensics depending upon the different digital devices used as a means of committing a particular crime. Some of the other branches in Digital forensics are:

1.1.1 Computer Forensics:

Computer forensics mainly deal with the extracting, preserving, analyzing and explaining the current status digital artifact like computer system, electronic documents, storage mediums and embedded systems. It can deal with other wide range of information related to areas like log files and actual files on the drive [15].

1.1.2 Mobile Forensics:

Mobile Forensics is the branch in which the digital evidence is recovered from the mobile device. This is somewhat different from the digital forensics because the
recovering of evidence is mainly concentrated on the inbuilt communication system (e.g. GSM). An Investigation will be focused mainly on call data, SMS and emails, but not what data is deleted.[16]. Using the information about location using inbuilt GPS or tracking cell site, logs mobile device location can be traced and used in the investigation.

1.1.3 Network Forensics:

This branch of forensics mainly deals with the observation and analyzing of network traffic data. It is done both in local and internet as well for evidence collection or intrusion detection. Generally this traffic is intercepted at the packet level and further it is filtered in real time for a quick solution or stored for later analysis. By using this forensics method, behavior of the network is determined by observing anomalous traffic and intrusions [17].

1.1.4 Database Forensics:

This branch of forensics mainly deals with the investigation of collecting evidence from the metadata and databases. In this database forensics mainly timestamps are associated for evidence collection by applying to the specified rows of the relational databases to check the integrity of data in databases. And the other techniques are also applied in the databases to identify the transactions in the databases to ensure fraud detection [18].

1.2 Goals of Digital Forensics:

Briefly, the goals of Digital Forensics can be described as “Identifying, Preserving and analyzing” the digital evidence in a way such that it is effectively presented to the court of law ensuring the integrity of data.
1.2.1 Identifying the Digital Evidence.

Identifying the digital evidence is the important and foremost step in the forensics. This phase describes how the data is stored and where it was stored on a particular device. The Investigator can use different methodologies and appropriate procedures to get evidence from the device which was used to commit crime. Now-a-days people are using a numerous number of digital devices in their daily life. So, there is an increased number of chances to commit crime. Hence investigator should be able to differentiate between different devices based upon the type of crime committed on that spot. The computer forensic examiner should be able to use appropriate technology for data extraction based upon the type of information stored in a particular digital device [19].

1.2.2 Preserving the Digital Evidence:

Preserving digital evidence is the one of other important phases in Digital Forensics. The important identified evidence of the previous phase should be stored in most secure and safe place in order to avoid any accidental loss or damage that may happen to evidence. The collected evidence should be stored in another copy which is called “master copy” from which other duplicate copies are replicated for further storage. The Investigator should also take care about not having any physical changes of digital device as well [19].

1.2.3 Analyzing the Digital Evidence:

Analyzing phase is the most important phase. Because the evidence that was extracted from the crime scene is processed in this phase. Several analyzing techniques
and knowledge were applied on the data extracted from the evidence. Generally, one should create a mirror image to the original data as analyzing cannot be done on original data directly, because there may be a chance of accidental data loss. It is a good phenomenon to maintain data integrity as well [19].

In order to perform different forensic techniques, there are numerous commercial and freeware tools that are being utilized currently. Some of them are discussed below.

1.2.4 Forensic tool kit:

The Forensic tool kit is used to find the lost information from computer during computer forensics. To guarantee the accuracy of computer forensic analysis this software tool was approved by the court of digital investigations. This tool is also designed to decrypt the software and crack the database. This tool has a user friendly interface with speed and high performance analytics for enterprise class.

1.2.5 ProDiscover Forensics tool:

The ProDiscover forensics tool creates an evidence report for legal proceedings and also helps investigators to identify and recover the deleted files, examine slack from the digital device. It also helps investigator to extract the data from the digital device without changing the sensitive information like when the file was accessed last time, changes made to it. With the hash comparison capability ProDiscover forensics tool allows an investigator to search for a word anywhere on the disk and slack space to find desired data.
1.2.6 The Sleuth Kit:

This forensic tool was designed as platform independent. The tool runs on both windows and UNIX system, so as to analyze the disk images and files recovered from digital device. This tool is used to process victim’s computer through command line files for computer disk forensic analysis.

Using this tool with command line interface has many difficulties, it was later combined with an easy graphical user interface which made the investigation much easier when compared to the previous interface. It gives investigator to search keywords, image integrity and other operations automatically.

Now-a-days a huge amount of data is collected during investigation and analyzing this data is the main problem in digital forensics. Due to the increase in the number of digital storage devices of the individual using daily, there is a need of implementing the new automated tools to assist the investigators to extract data in efficient way [1]. Many existing tools are efficient in extracting the raw data from the digital devices during the investigation. The internet Evidence finder is a tool which finds, analyze and presents recovered data from the internet and correlates to certain internet usage like Facebook chat artifacts. Automated extraction tools help to provide data to investigators in the understandable form. Since, we need a solution to address the problem of relating the past digital events [5] which is implementing automatic reconstruction of sequence of events.
1.3 Real life Case:

Shown in Figure 1.1 there was a case in Minnesota in which two 13-year-old girls missing, after some days of the investigation, detectives able to find out the girls in the basement of 23-year-old young man’s house named Casey Lee Chinn, who is charged with kidnapping and felony criminal sexual conduct of a child. In this case detectives used the missing girl’s cell phones, tablets and other digital devices for clues such as call logs and tweets and other interactions in social networks. They collected huge amount of data and looked it for several days in order to get relevant data for their investigation.

Figure 1.1 Computer collected as digital evidence [13]
In real life crime investigation cases, evidence is taken from the digital devices from which the accused persons were used to commit crime. There will be a huge amount of raw data which is available for investigator to analyze further in that particular case. Analyzing such an amount of data manually is very complex and time consuming for investigators. Hence, proposing a solution for automating this raw data evidence from digital devices will aid the investigator to investigate case quickly and he can concentrate on some other work on the case which is very important.

1.4 Related Work

Basically timelines approach is divided into different types as listed below

1.4.1 Creating Timelines using File System Times: (Carrier and Bunt)

In general, most of the current digital forensic tools will reconstruct the entire file system of a disk and allow investigator to access the file metadata. This metadata will have the information regarding the file like Modified, Accessed, and Created (MAC) depending upon the different file system. There are other tools which they can capable of transforming different associations of the file system into the timeline. For example, Carrier by using a sleuth kit [4] explains how to create a timeline for file activity, and using Encase [2] Bunt describes how to create and view the timelines as graphical view. The main problem of the timelines constructed based on the file metadata will not accept the times those available by seeing the contents of the file. Hence, for effective analyses of data, sequence of events to be reconstructed.
In order to see what happened to the times of files from certain date, table 1.1 must be sorted by each and every column respectively like column C, column A, and by column M and again by column B. If the extracted data is in large sets, then the examiner should sort all the data manually based upon constraints like, time or date and sometimes he may forget to sort some columns. So, the data in table should be represented in more clearly.

Table 1.1 MAC Times for two files [20]

<table>
<thead>
<tr>
<th>Filename</th>
<th>Created @</th>
<th>Accessed (A)</th>
<th>Written (M)</th>
<th>Entry Modified (B)</th>
</tr>
</thead>
</table>

Representation of a timeline view which is tabulated in table 1.2 will give a clear idea about the details of a file like, modified, accessed, and created times of different files in the database. When compared to the previously stated table in table 1.2 has a columns sorted based upon the date and time. The remaining two columns are filename and actions performed on that particular file. The last column shows actions preformed on the respective files for different data and times clearly as Modified (M), Accessed (A), Created (C) and Entry modified (B) in metadata. From table 1.2 one can understand that both Autoexec.bat and Config.sys were Created at 15:26:30 and Modified at 10:26:30. Such a timeline view of data will help the examiner to analyze data in huge sets of data.
1.4.2 Creating Timelines by considering Times from Inside Files: (Olsson, Bolt and Guðjónsson)

After conducting research Olsson and Bolt [11] proposed tool which timelines based upon the metadata with Cyber Forensic Time Lab (CFTL). This tool can process the file systems like FAT16, FAT32 and NTFS to recover times from files. It will not only list the files, but also process and finds the other different data formats like browser history and E-mail. In addition to that, it will also obtain their respective timestamps in the form of index list and provides to the investigator. The graphical representation of results is also included in this tool.

Log2timeline [7] with timeline scanner enhancement can automatically look at files and directories. So, if we have a proper input module of a file, we can obtain times of that file easily and then added to its respective timeline effectively. One of the references in [7] indicates that whenever the performance and ability of event table in SQLite output is described, there is a possibility of grouping similar events together. Like this we have vivid description and review of timeline software tools is provided in [3].
1.4.3 Visualizations for extracting data: (Lowman, Buchholz and Falk)

Zeitline is a GUI provided forensic tool designed by Buchholz and Falk [1]. This enables us to get the times of the file from different sources. Filtering and sorting options of events are made available. This system also has a concept of atomic and complex events in which atomic events are imported directly from system and complex events are comprised of single events or other multiple complicated events. This tool has the flexibility for investigator to combine atomic and complex events.

Netherlands Forensic Institute (NFI Labs) [10] has developed a tool using java called Aftertime. Using this application different operations can be performed on the disk space and as a result enhanced timeline generation can be obtained in the form histogram. The further visualization of data from investigator becomes easy.

Lowman [9] describes several techniques for visualizing data from different sources like web history. It also comprises of heat maps, bar charts, timeline activity view. Even though the previously described visualizations are very effective, but it is difficult to differentiate and analyze if all events are kept together at the same time.

So, based upon the all visualizations described previously it is very difficult to analyze and visualize the large number of events which are obtained from the file system and metadata. So, an approach that would be very helpful and effective, if some of the patterns with similar kind of activity are highlighted as interested parts in timeline. And most importantly, after getting data from source it should be stored in a safe place.
2 NARRATIVE

2.1 Problem Statement:

Now-a-days due to advent of technologies, there are so many electronic storage devices in use by people everywhere. In order to determine the circumstances of particular incident investigators need to reconstruct the events that occurred in the past. Hence, this reconstruction of past data events becomes very complex. Because the data is spread in large extent in a crime scene. The reconstruction of events is almost impossible and increase cognitive load due to the huge quantity of events extracted. Hence there is a lot of tedious work and complex to analyze, here there is a question of developing such automated tools for reconstructing timeline events for furthermore investigation.

2.2 Motivation:

Previously, many techniques were proposed in the field of digital forensics for investigation. Most of them are mainly concentrated and explained how to extract data from digital devices obtained from the crime scene. After extracting data from digital devices investigator is provided with a huge amount of raw data for further processing and it is very difficult. Due to this reason, there is a need of proposing a solution which can reduce the complexity of analyzing huge amounts of data and help the investigator in furthermore investigation.
2.3 Project Objective:

The main scope of this project is to extract the data from digital devices developed extractor tool. The extracted data is stored in MySQL database in the form of rows and columns for easy analysis. This extracted data helps the investigator in analyzing the huge data extracted from digital devices manually and reduce the burden on the investigator. So, that he can use the time to concentrate more on further investigation.
3 PROPOSED SYSTEM DESIGN

3.1 System Architecture.

Main components of the proposed software are Extractor, MySQL Database and Analyzer. This prototype has a good GUI and allows investigator to access application easily.

The different components in Figure 3.1 are explained briefly:

1) Extractor
2) MySQL database
3) Analyzer

Figure 3.1 Proposed System Architecture
3.1.1 Extractor

In the extractor module all the raw (or) unstructured data from different digital devices is extracted using an extractor tool. For explaining the working of this prototype an extractor tool was developed. For every computer, browsing history of the particular browser is stored in specified places called SQLite files in the computer. Now, identify the SQLite file which has Browsing history data. These SQLite files can be opened using SQLite Manager Plugin to see the data structure in which data is stored. Now the respective SQLite database browser file location is identified and required data is extracted. This data extraction tool was developed by writing a java code to query the data from browser history file. The extracted data is stored in the form of .txt file in another location of the computer or thumb drive.

Once, after having the all .txt source files, the data from these .txt files are stored into the respective fields that are created in the MySQL database for further analysis.

Figure 3.2 describes how the data is extracted from the digital devices using extractor tool.

**Figure 3.2 Flow Chart Diagram Of Extracting Data From Computer**
Figure 3.3 showing the sample screenshot of History.txt file having different details of browsing history of Firefox browser.

![Sample Screenshot of History.txt File](image)

Figure 3.3 Screen Shot Of History.Txt File.

### 3.1.2 MySQL Database

The implemented prototype uses MySQL database for storing extracted data from the digital evidence. Generally in earlier times extracted data is stored in the memory but, later on the extracted data tend to increase its size of storage causing
problems like disk paging and resulting in poor performance issues. Many tools have implemented a variety of different means of storing timelines for the extracted raw data. Storing the timelines of the extracted raw data has the flexibility of executing commands like grep to query for specific events [7].

Multiple queries are necessary to execute on database. Hence the use of MySQL has been found to have performance benefits as the investigator may have limited amount of knowledge to perform queries and searches for specific events in it.

The database design consists of different tables like, history table and search table. In each and every table the respective data that was present in the form of .txt file by using an extractor tool is imported to the respective fields.

For e.g. The History table shown in Table 3.1 consists of fields like URL, Tittle, Visit Time, Visit Count, Visited From, Web browser, User Profile. After importing data from the .txt file the respective data for each field will get stored. Once after storing the values in the history table the required actions are performed in the analysis phase.

Table 3.1 Sample History Database Table

<table>
<thead>
<tr>
<th>URL</th>
<th><a href="https://www.google.com">https://www.google.com</a></th>
</tr>
</thead>
<tbody>
<tr>
<td>tittle</td>
<td>Tom and jerry</td>
</tr>
<tr>
<td>Visit Time</td>
<td>29-03-2015 PM 12:59:14</td>
</tr>
<tr>
<td>Visit Count</td>
<td>2</td>
</tr>
<tr>
<td>Visit From</td>
<td><a href="https://www.google.com">https://www.google.com</a></td>
</tr>
<tr>
<td>Web Browser</td>
<td>Chrome</td>
</tr>
<tr>
<td>User Profile</td>
<td>Avinash</td>
</tr>
</tbody>
</table>
Another database table shown in Table 3.2 is a Search Database table in which it contains the fields like, Search Text, Search Engine, Search Type, Search time, Web Browser, Hits, and URL. In this each field after importing data from extractor tool the corresponding values for each data field is stored in this Database table.

Table 3.2 Sample Search database table.

<table>
<thead>
<tr>
<th>Search text</th>
<th>Taj mahal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Search engine</td>
<td>Google</td>
</tr>
<tr>
<td>Search Type</td>
<td>General</td>
</tr>
<tr>
<td>Search Time</td>
<td>01-04-2015 PM 12:17:50</td>
</tr>
<tr>
<td>Web Browser</td>
<td>Chrome</td>
</tr>
<tr>
<td>Hits</td>
<td>1</td>
</tr>
<tr>
<td>URL</td>
<td><a href="https://www.google.com">https://www.google.com</a></td>
</tr>
</tbody>
</table>

3.1.3 Analyzer

The analyzer selects data from different records to compare and analyze, specified by the investigator. By comparing the data between two different cases investigator may have the chance of correlating similar data between them, based upon the nature of case investigating in which the compared data will help him in further analysis of case.

And one more option in which this prototype includes is searching keyword from the given database table. This search function enables investigator to search and filter the required data from the huge amount of data from database. Search function is the most important functionality through which the manual process of analyzing huge data in the database is done more quickly, reducing the burden on investigator by reconstructing the timelines of the data. This searching and sorting is performed by using Collections framework package in Java. Depending upon the scenario of the particular case,
investigator will search for the related evidence that he may find from database. He can enter any keyword relating Browser information or related URL’s and date. Based upon the timeline created and the angle of investigation in which investigator is proceeding, he will draw the conclusions about accused person correlating timeline of events and nature of case.

The input text may vary anything from related keyword to the times and dates of the particular event happened on or before the time of crime scene. The result field is a generated report of particular key word consisting of various related times, date, URL visited, visited count and user name. Hence investigator can easily analyze the data based upon timeline that was created by this prototype.

![Flow Chart Diagram Of Data In Analyzer](image)

**Figure 3.4 Flow Chart Diagram Of Data In Analyzer**
3.2 Flow chart Diagram of Proposed system.

Figure 3.3 will represent the flow of control of execution of the application.

Figure 3.5 Flow Chart Diagram For Proposed System.

In figure 3.5 initially, data is given as input in different the form of .txt files, like browsing and search history. After taking this as an input, the data is stored into the database. This stored data is used for the further processing like comparing between two different records and searching among the given data base for required pattern or the times of the different events happened.
3.3 Input and output Flow Diagram for Given .txt file:

Figure 3.6 showing the flow of input and output of the given .txt file to the database

![Flow Diagram](image)

Figure 3.6 Showing Input Flow Diagram Of .Txt File.

3.4 Use Case Diagram:

Use case diagram is the graphical representation of functionalities of the system in the terms of actors. Here in the use case, actor is admin and functionalities are like extracting data and storing data in database and processing it for final results. Figure 3.7 depicts the functionalities of actor in the system.
Figure 3.7 Use Case
4. FUNCTIONALITY OF APPLICATION

The important functionality of this prototype is to reconstruct the sequence of events of the raw data that was extracted from the digital devices collected in crime investigation scene. By which the investigator can easily analyze the huge amount of data without much complexity. The main source from which the developed prototype can take the input is .txt source file shown in the figure 4.1.

![Sample Source File](image)

Figure 4.1 Sample “.Txt” Source File.
4.1 User interface for Automatic Event Sequence reconstruction prototype

Figure 4.2 showing the login form of the prototype consisting username field and password field.

![Login Page For Prototype](image)

4.1.1 Main Form for automatic Event Sequence generation

In Figure 4.3 the Main form of the prototype will have these options like

1) Upload
2) View History
3) Analyze
4) Search
5) Exit
4.1.2 Uploading the extracted data

In the figure 4.4 after clicking Upload button, there comes another two different options from which the data can be uploaded from the .txt file. While uploading each record, user must enter be unique record ID.

The different upload buttons that appear in window are

1) Upload browsing history : browse and upload history.txt file
2) Upload Search history   : browse and upload Search history.txt file
4.1.3 View history of data stored in database.

View history button was used to retrieve the stored data from the database. In this form investigator has to give the ID of the particular record in the database to view the data like browsing history, search history shown in Figure 4.5.
4.1.4 Analyzing the data from databases.

Figure 4.6 shows the analyzing window, in which data from two records in a database is compared using the string comparison method. Data is stored in the form of rows consisting the details like, URL name, browser name, visited date and time respectively are kept in the single string for comparison. So, by considering only those fields in two data records, comparison method is selected for implementing prototype. For that two database ID values should be given. Once, after hitting compare button the results will have the number of unmatched records with other database and vice versa. It also gives the total no of rows compared and total number of unmatched records from both given databases.
4.1.5 Searching data

In this Figure 4.7, designed prototype has one field where user can search the desired key from the database. Here, only one search field is kept because the prototype is developed for retrieving the similar data from the database based upon given keyword. After that investigator need to verify the related data to further investigation. Here this not completely automated process but helps the investigator to analyze huge data by providing similar data based upon given keyword. So, whenever the investigator gives a particular related keyword based upon case he is investigating, prototype will provide all details containing that particular given word.
Figure 4.7 Pop-Up Window For Filtering Data Asking Key Word
The project consists of a prototype which will accept extracted data in the form of .txt file as an input file and data in it is stored in rows and columns. Now, investigator can view the history of data stored in the database. And he can also compare and analyze the data between two different database records and get the count of unmatched records.

The other functionality of this prototype is to search a particular keyword from the database and it will return the every detail accompanied with it for the quick reference.

5.1 Test Cases

Test case: Investigating browsing history of a banking fraud

Step 1: uploading history.txt file to the database:

Figure 5.1 showing the upload form of the prototype in which history.txt file is uploaded to the database. For each new data record a unique ID is entered to store it in the database.
Figure 5.1 Uploading Data From .Txt File

Step 2: view history from database

Figure 5.2 Viewing Web History From Database
By observing data from Figure 5.2 prototype shows the history of data stored in the database. Each column has different details regarding the browsing data of the banking fraud. It includes the list of URL’s, title, visited time, visited date and browser name respectively.

**Step 3: Analyzing data from the database**

![Cyber Investigation Application](image)

**Figure 5.3 Showing The Matched And Unmatched Records**

Figure 5.3 Shows the results obtained from comparing the two records from database. It gives the number of unmatched and matched records from corresponding database given. So, that the investigator can analyze the previous database records for quick reference to solve the corresponding case.
Step 5: Searching for the particular data from database using keyword.

![Image of Cyber Investigation Application]

**Figure 5.4 Searching History By Key Word**

Here in Figure 5.4 by giving the keyword date of the event and selecting the source of viewing data from history, it shows a table with column details like browser address, URL visited, visited time, visited from, and title for regarding User name was retrieved. So, here the data was filtered according to the given search word from huge database.
Step 6: Observing the results from sequence of events in particular date.

<table>
<thead>
<tr>
<th>URL</th>
<th>TITLE</th>
<th>VISIT TIME</th>
<th>VISIT</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://www.google.com/">https://www.google.com/</a></td>
<td>Google</td>
<td>05/01/2015 02:38</td>
<td>2</td>
</tr>
<tr>
<td><a href="https://www.google.com/search?q=b">https://www.google.com/search?q=b</a>...</td>
<td>Best phishing techniques...</td>
<td>05/01/2015 02:38</td>
<td>1</td>
</tr>
<tr>
<td><a href="https://www.google.com/url?q=s">https://www.google.com/url?q=s</a>...</td>
<td>null</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="https://www.pentestgeek.com/2013/0">https://www.pentestgeek.com/2013/0</a>...</td>
<td>How do I phish? - Advanced...</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.google.com/url?q=s">http://www.google.com/url?q=s</a>...</td>
<td>null</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.google.com/url?q=s">http://www.google.com/url?q=s</a>...</td>
<td>null</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.hacking-tutorial.com/hacki">http://www.hacking-tutorial.com/hacki</a>...</td>
<td>Tutorial Hacking Facebook...</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://resources.infosecinstitute.com/">http://resources.infosecinstitute.com/</a>...</td>
<td>Phishing Techniques S...</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.hacking-tutorial.com/hacki">http://www.hacking-tutorial.com/hacki</a>...</td>
<td>Tutorial Hacking Facebook...</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.google.com/url?q=s">http://www.google.com/url?q=s</a>...</td>
<td>null</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.pcworld.com/article/1352">http://www.pcworld.com/article/1352</a>...</td>
<td>Types of Phishing Att...</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.google.com/url?q=s">http://www.google.com/url?q=s</a>...</td>
<td>null</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.infoscrip.com/english/arti">http://www.infoscrip.com/english/arti</a>...</td>
<td>InfoSec - Phishing Con...</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.google.com/url?q=s">http://www.google.com/url?q=s</a>...</td>
<td>null</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://www.makeuseof.com/tag/phish">http://www.makeuseof.com/tag/phish</a>...</td>
<td>What Exactly Is Phishing...</td>
<td>05/01/2015 02:39</td>
<td>1</td>
</tr>
<tr>
<td><a href="https://github.com/pentestgeek/jigsaw">https://github.com/pentestgeek/jigsaw</a>...</td>
<td>Page not found - Github...</td>
<td>05/01/2015 02:40</td>
<td>1</td>
</tr>
<tr>
<td><a href="https://www.google.com/search?q=h">https://www.google.com/search?q=h</a>...</td>
<td>how to hack bank acco...</td>
<td>05/01/2015 02:41</td>
<td>2</td>
</tr>
<tr>
<td><a href="http://www.google.com/url?q=s">http://www.google.com/url?q=s</a>...</td>
<td>null</td>
<td>05/01/2015 02:40</td>
<td>1</td>
</tr>
<tr>
<td><a href="http://avenuemcomputers.com/newsli">http://avenuemcomputers.com/newsli</a>...</td>
<td>7 steps to hack into so...</td>
<td>05/01/2015 02:40</td>
<td>1</td>
</tr>
<tr>
<td><a href="https://www.google.com/url?q=s">https://www.google.com/url?q=s</a>...</td>
<td>null</td>
<td>05/01/2015 02:40</td>
<td>1</td>
</tr>
</tbody>
</table>

Figure 5.5 Showing the Sequence Of Events Occurred On Particular Date

Figure 5.5 showing the sequence of the events happened on the particular date. Here, the search key word is given by the investigator. The highlighted portion depicts the title of the particular topic browsed, the date and time of event.

Hence, the amount of data that was displayed in the results has only details similar to the keyword searched by the investigator. Here, without sorting all of the huge trivial data, this prototype is providing a sequence of events for required data in which investigator was looking.
6 CONCLUSION

The proposed prototype shows the implementation of reconstruction of a sequence of events from the extracted raw data. The extracted data was in unstructured format which is later stored in structured format i.e., in rows and columns in the database. It also provides the option for investigator to compare the selected database records and show the matched and unmatched records. So, that investigator can analyze furthermore in any case by checking with the previous records. Most importantly, it has an option to search the required data for investigation by giving desired keyword and observing the constructed sequence of events. The constructed sequence of events by using this application will reduce the burden on investigator to analyze huge amount of data obtained from the digital devices and allows to analyze the interested areas of the raw data in less time.
However, this is not complete automated solution for analysis of data without experienced forensic analyst but, aids the analyst to speed up the investigation. In this prototype source of the extracted raw data is in the form of .txt file. In future the implemented prototype should also compatible to accept the all data formats. In this prototype raw data was extracted from web browser and future work involves extracting data from hard drives from computer and analyzing the times of files. There is also a scope to extract and add system related information into this prototype. And one more thing that can be added in the future work is to reduce time deviance as this prototype was considered in one time zone only. But the data can be extracted from devices which are collected from different places for investigation.
BIBLIOGRAPHY


[16] D. Yadav, M. Mishra, S. Prakash. Mobile Forensics Challenges and Admissibility of Electronic Evidences in India 2013


APPENDIX

Class diagram:
Java code Snippets:

Code for data extraction tool:

```java
package com.test.sqlite;
import java.sql.Connection;
import java.sql.Statement;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.text.SimpleDateFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileWriter;

public class ConnectSQLiteDB {
    public static void main(String[] args) throws Exception {
        // register the driver
        String sDriverName = "org.sqlite.JDBC";
        Class.forName(sDriverName);
        String sJdbc = "jdbc:sqlite:"

        // which will produce a legitimate URL for SQLite JDBC:
        int iTimeout = 30;
        String firefoxQuery = "SELECT url,title,visits_count,visit_date FROM moz_places";
        String fireFoxpath = "C:\\Users\\Avinash\\AppData\\Roaming\\Mozilla\\Firefox\\Profiles\\nqhx0ew.default\\places.sqlite";
        String fireFoxURL = sJdbc + ":" + fireFoxpath;
```
String chromeQuery = "SELECT url, title, visit_count, last_visit_time FROM urls";
String chromePath = "C:\Users\[username]\AppData\Local\Google\Chrome\User Data\Default\History";
String chromeURL = chromePath + ":\" + chromePath;
// create a database connection

//Write File Location
File file = new File("C:\Users\[username]\Desktop\MASTER'S GRAD PROJECT\mozilla_history\history.txt");
FileWriter fw = new FileWriter(file.getAbsolutePath()); // returns file object containing absolute path of current
BufferedWriter bw = new BufferedWriter(fw);

// if file doesn't exist, then create it
if (!file.exists())
    file.createNewFile();

try{
    Connection conn = DriverManager.getConnection(chromeURL);
    try{
        Statement stmt = conn.createStatement();
        try{
            stmt.setQueryTimeout(10000);
            ResultSet rs = stmt.executeQuery(chromeQuery);
            try{
                while(rs.next())
                    {
                        String URL = rs.getString("url");
                        System.out.print(URL + ", ");
                        String title = rs.getString("title");
                        System.out.print(title + ", ");
                        String visit_count = rs.getString("visit_count");
                        System.out.print(visit_count + ", ");
                        String last_visit_date_Epoch = rs.getString("last_visit_date");
                        String last_visit_date = null;
                        if(last_visit_date_Epoch !=null)
                            {
                                Date date = new Date(Long.parseLong(last_visit_date_Epoch).substring(0, 19));
                                SimpleDateFormat format = new SimpleDateFormat("MM/dd/yyyy HH:mm:ss aa");
                                last_visit_date = format.format(date);
                            }
                
                System.out.println(last_visit_date_Epoch);
                bw.newLine();
                bw.write("URL:" + URL);
                bw.newLine();
                bw.write("Title:" + title);
                bw.newLine();
                bw.write("Visit Time:" + last_visit_date);
                bw.newLine();
            }
        }
    }
}
while (rs.next())
{
    String URL = rs.getString("url");
    System.out.println(URL+"\n");
    String title = rs.getString("title");
    System.out.println(title+"\n");
    String visit_count = rs.getString("visit_count");
    System.out.println(visit_count+"\n");
    String last_visit_date_Epoch = rs.getString("last_visit_time");
    String last_visit_date = null;
    if (last_visit_date_Epoch !=null && last_visit_date_Epoch.equals("0"))
    {
        Long epodate = Long.parseLong(last_visit_date_Epoch) - Long.parseLong("1164447360000000000");
        String epodatetst = epodate.tostring();
        epodatetst = epodatetst.substring(0,13);
        Date date = new Date(Long.parseLong(epodatetst));
        DateFormat format = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss a");
        last_visit_date = format.format(date);
    }
    System.out.println(last_visit_date);
    bw.write("-------------------------------------------------------\n");
    bw.newLine();
    bw.write("URL :"+URL);
    bw.newLine();
    bw.write("Title :"+title);
    bw.newLine();
    bw.write("Visit Time :"+last_visit_date);
    bw.newLine();
    bw.write("Visit Count :"+visit_count);
    bw.newLine();
    bw.write("Visited From :");
    bw.newLine();
    bw.write("Web Browser :Chrome");
    bw.newLine();
    bw.write("User Profile :Avinash");
    bw.newLine();
    bw.write("-------------------------------------------------------\n");
    bw.newLine();
    bw.newLine();
    bw.newLine();
    bw.newLine();
Code for Searching records from database:

```java
public Vector<HistoryBean> getBrowserHistory(String id, String key)
{
    Vector<HistoryBean> obj = new Vector<HistoryBean>();
    try
    {
        String query = "select " + escapeData("id") + " id, " + escapeData("url") + " url, " + escapeData("title") + " title, " + escapeData("visitime") + " visitime, " + escapeData("visitcount") + " visitcount, " + escapeData("visitedform") + " visitedform, " + escapeData("webbrowser") + " webbrowser, " + escapeData("userprofile") + " userprofile from history where id in (" + escapeData("id") + " like \%'\%id\%' or " + escapeData("id") + " like \%'\%key\%' or \\n        " + escapeData("id") + " like \%'\%key\%' or " + escapeData("url") + " like \%'\%key\%' or " + escapeData("title") + " like \%'\%key\%' or " + escapeData("visitime") + " like \%'\%key\%' or " + escapeData("visitcount") + " like \%'\%key\%' or " + escapeData("visitedform") + " like \%'\%key\%' or " + escapeData("webbrowser") + " like \%'\%key\%' or " + escapeData("userprofile") + " like \%'\%key\%' )"
            
        //SELECT cast(url as char),replace(cast(quote(url) as char),'\'', '') FROM history
    
    public Vector<SearchBean> getSearchHistory(String id, String key)
    {
        Vector<SearchBean> obj = new Vector<SearchBean>();
        Statement st = null;
        ResultSet rs = null;
        String sql = "select searchtext, searchengin, searchtype, " + id + " from searches where id in (" + escapeData("id") + " like \%'\%key\%' or searchtext like \%'\%key\%' or searchengin like \%'\%key\%' or searchtype like \%'\%key\%' )"
            
        try
        {
            st = conn.createStatement();
            rs = st.executeQuery(sql);
            while (rs.next())
            {
                SearchBean sb = new SearchBean();
                /* create table searches */
```
Code for Analyzing data:

```
public void run()
{
try
{
    // while(i!=14)
    // {
        STATUS.append("Reading Group 1 Browser History...\n");
        Vector<HistoryBean> hb1 = dbc.getBrowserHistory(id1);
        i++; // 1
        System.out.println("I:"+i);
        STATUS.append("Reading Group 2 Browser History...\n");
        Vector<HistoryBean> hb2 = dbc.getBrowserHistory(id2);
        i++; // 2
        System.out.println("I:"+i);
        // STATUS.append("Reading Group 1 Passwords History...\n");
        Vector<PasswordBean> pb1 = dbc.getPasswordHistory(id1); /*
        i++; // 3
        System.out.println("I:"+i);
        // STATUS.append("Reading Group 2 Password History...\n");
        Vector<PasswordBean> pb2 = dbc.getPasswordHistory(id2); /*
        i++; // 4
        System.out.println("I:"+i);
        STATUS.append("Reading Group 1 System History...\n");
        Vector<SearchBean> sb1 = dbc.getSearchHistory(id1);
        i++; // 5
        System.out.println("I:"+i);
        STATUS.append("Reading Group 2 System History...\n");
        Vector<SearchBean> sb2 = dbc.getSearchHistory(id2);
        i++; // 6
        System.out.println("I:"+i);
        STATUS.append("Gather Group 1 Fields Data...\n");
        List<String> ob1 = getCompressObject(hb1,new Vector<>(),sb1);
        i++; // 7
        System.out.println("I:"+i);
        STATUS.append("Gather Group 2 Fields Data...\n");
        List<String> ob2 = getCompressObject(hb2,new Vector<>(),sb2);
        i++; // 8
        System.out.println("I:"+i);
        RemoveDust rd = new RemoveDust();
        List<String> obj1_1 = rd.remove(ob1);
        List<String> obj2_1 = rd.remove(ob2);
        STATUS.append("Pattern Matching Stated...\n");
        int cList1 = obj1_1.size();
        int cList2 = obj2_1.size();
        int value[]=compare(obj1_1, obj2_1);
        int nList1 = obj1_1.size();
        int nList2 = obj2_1.size();
        group1Un.setText(""+value[0]);
        group2Un.setText(""+value[1]);
        totUn.setText(""+(value[0]+value[1]));
        int mRec = ((cList1+cList2-value[0]-value[1])/2); // get the count
        float perc_ = ((mRec)*100)/((cList1+cList2)/2);
```