

WEB HISTORY VISUALISATION FOR FORENSIC INVESTIGATIONS

SARAH LOWMAN

This dissertation was submitted in part fulfilment of requirements for the degree of

MSc Forensic Informatics

DEPT. OF COMPUTER AND INFORMATION SCIENCES

UNIVERSITY OF STRATHCLYDE

September 2010

DECLARATION

This dissertation is submitted in part fulfilment of the requirements for the degree of MSc Forensic Informatics of the University of Strathclyde.

I declare that, in accordance with University Regulation 20.1.20, this dissertation embodies the results of my own work and that it has been composed by myself. Following normal academic conventions, I have made due acknowledgement to the work of others.

I give permission to the University of Strathclyde, Department of Computer and Information Sciences, to provide copies of the dissertation, at cost, to those who may in the future request a copy of the dissertation for private study or research.

I give permission to the University of Strathclyde, Department of Computer and Information Sciences, to place a copy of the dissertation in a publicly available electronic archive.

(please tick) Yes [☐] No [☐]

Signature:

Date

ABSTRACT

Data visualisation is often an area neglected by programmers because it requires more skills and time to do properly, and does not seemingly add any new functionality to software since it just shows the resultant data in a new way. However, good visual interfaces and visualisations can vastly improve the time it takes to complete a task, reduce errors, increase concentration and allow a better knowledge of the data.

This project explores visualisations for forensic investigations, using web history analysis as an example area in much need of visual tools. A web visualisation tool called Webscavator was created and its effectiveness evaluated against a non-visual tool, Net Analysis.

Many of the problems Net Analysis has are common to forensic tools – the dataset is large and cluttered because it is presented in one large table. The program provides very little to help the investigator analyse the results, so a large amount of time is spent filtering the table into a more digestible size. The table format also makes it difficult to explain results to those not familiar with the software, such as police officers.

Three digital forensic investigators took part in usability tests comparing Webscavator to Net Analysis. The results indicated that Webscavator improved the accuracy, speed and confidence of the participant's answers. Questionnaire results indicated that the visualisations would be useful in forensic investigations, and would increase efficiency, reduce errors, allow more focused attention and allow non-technical people to understand the results better. These results can be generalised, furthering the argument that visualisations make data easier to interpret.

ACKNOWLEDGEMENTS

Many thanks to my volunteers Ian, Andrew and Mike who helped me with my user testing. Your input was incredibly valuable.

Thanks to my supervisor Ian Ferguson for your sound advice and Steven for proof reading this document until it made sense.

CONTENTS

Chapter 1: Introduction.....	1
1.1 Web History Analysis.....	2
1.2 Layout of Dissertation.....	3
1.3 Terminology Used and Notes	4
Chapter 2: Research Methodology	5
2.1 Problem Statement.....	5
2.2 Research Questions.....	7
2.3 Application Methodologies	8
2.3.1 Design	8
2.3.2 Implementation	9
2.3.3 Testing	9
2.4 Experimental Design	10
2.4.1 User Testing	10
2.4.2 Scenario Design	10
2.4.3 Output Data	11
Chapter 3: Web Browsers.....	13
3.1 Internet Explorer	14
3.1.1 index.dat file headers.....	15
3.1.2 index.dat file contents.....	16
3.2 Firefox.....	19
3.2.1 places.sqlite	20

3.2.2	formhistory.sqlite	23
3.3	Google Chrome	24
2.3.1	History	24
3.3.2	Web Data	26
3.3.3	History Index Files	26
3.4	Safari	26
3.4.1	History.plist	27
3.4.2	Other significant data	28
3.5	Opera	30
3.5.1	Global_history.dat	30
3.5.2	Typed_history.xml	32
Chapter 4: Forensic Web History Programs		33
4.1	Tools computer forensic investigators use	34
4.2	Pasco	35
4.3	Web Historian	36
4.4	Net Analysis	39
Chapter 5: Visualising Data		43
5.1	Visualising Data	44
5.1.1	Cognitive Benefits	44
5.1.2	Perception Benefits	44
5.1.3	Making Effective Visualisations	47
5.1.4	Evaluating Visualisations	48
5.2	Computer Forensic Visualisations	49
5.2.1	File Visualisations	49

5.2.2	Network and Web Log Visualisations	54
5.2.3	Timeline Visualisations	59
5.3	Visualising Web Browser History	61
5.3.1	Available Visualisation APIs	62
Chapter 6: Webscavator		65
6.1	Filters	67
6.2	Visualisations.....	69
6.2.1	Heat-map.....	69
6.2.2	Timeline	70
6.2.3	Bar Chart.....	74
6.2.4	Word Cloud	75
6.2.5	Tree Structure	78
Chapter 7: Analysis of Results.....		81
7.1	Scenario 1 Results.....	87
7.2	Scenario 2 Results.....	90
7.3	Quantative Data Summary and Conclusions.....	94
7.4	Usability Results.....	96
7.4.1	Questionnaire Results.....	96
Chapter 8: Conclusions		101
8.1	Achievement of Aims	101
8.2	Recommendations.....	103
8.3	Further Work	103
8.3.1	Webscavator Improvements.....	103
8.3.2	Further Testing	104

Appendix.....	i
Bibliography	i

TABLE OF FIGURES

Figure 1 - The median usage share of web browsers.....	14
Figure 2 – A cache file header	16
Figure 3 - A activity record in a cache file	17
Figure 4 - A activity record in a cache file	18
Figure 5 - A activity record in a file	19
Figure 6 - Part of table showing the user clicking on links	22
Figure 7 - Part of table showing the corresponding URLs in Figure 6.....	23
Figure 8 - Sample output from converting	28
Figure 9 - Viewing web history in Safari. The image displayed is a cached thumbnail of the website.....	29
Figure 10 - Example entries in	31
Figure 11 – Same entries as in Figure 10 viewed via Opera's history viewer.....	32
Figure 12 - Example output from Pasco shown in Excel.....	36
Figure 13 – Example Web Historian screenshot	37
Figure 14 - WebHistorian's Pie Chart of 'Website frequencies by domain'	38
Figure 15 - WebHistorian's Timeline for 24 th June 2010.....	39
Figure 16 - Screenshot of Net Analysis with Firefox 3 data	40
Figure 17 - Net Analysis Advanced Filtering options	41
Figure 18 - Find the red dot amongst the blue.....	45
Figure 19 - Find the red dot amongst the blue dots and red squares	46
Figure 20 - The figure at the top is perceived to break down into that of the bottom right not the discontinuous bottom left	47
Figure 21 - We perceive a circle even though the image is of disjointed lines	47
Figure 22 - Files in a directory filtered by file size. Light coloured blocks are large files, and darker coloured are smaller files	50
Figure 23 - Tree map view showing modified file recency	50

Figure 24 - Screenshot of textual visualisation system. Shows Tree-map of the system (left), a tag cloud of the terms in the selected files (top right), and metadata of the selected files (bottom right) (Stamps, Franck, Carver, Jankun-Kelly, Wilson, & Swan, 2009).	52
Figure 25 - SOM visualisation tool. Component maps generated from files (Fei, 2007).....	54
Figure 26 - Network Visualisation Overview (Krasser, Conti, Grizzard, Gribschaw, & Owen, 2005)	55
Figure 27 - Network Visualisation Example (Krasser, Conti, Grizzard, Gribschaw, & Owen, 2005)	56
Figure 28 - Outbound Botnet attack traffic (Krasser, Conti, Grizzard, Gribschaw, & Owen, 2005)	56
Figure 29 - Tudumi visualisation example (Takada & Koike, 2002a).....	58
Figure 30 - Tudumi screenshot (Takada & Koike, 2002a)	59
Figure 31 – CyberForensics TimeLab screenshot (Olsson & Boldt, 2009)	60
Figure 32 - An example word cloud for the author's search terms in the last month made in Wordle	61
Figure 33 - Screenshot of Timeline showing the events leading up to and after the assassination of JFK	63
Figure 34 - Screenshot of the wizard to add a new case	66
Figure 35 - Screenshot of the home page of Webscavator once a case has been loaded.....	67
Figure 36 - Screenshot of the filter bar with 'Social Networking URLs' highlighted and 'News URLs' removed.....	67
Figure 37 - Screenshot of the add filter form.....	68
Figure 38 - Screenshot of a heat-map	70
Figure 39 - Screenshot of the timeline	71
Figure 40 - Screenshot of the timegraph when the mouse hovers over a point	72
Figure 41 - Screenshot of the timegraph zoomed in with highlighting.....	73
Figure 42 - Screenshot of the top 20 domain names.....	75
Figure 43 - Screenshot of the pop-up box when the 'blogspot.com' bar was clicked	75
Figure 44 - Screenshot of the search terms word cloud.....	77

Figure 45 - Screenshot of pop-up when the term 'python' is clicked.....	77
Figure 46 - Screenshot of pie chart and document file structure for the D drive.....	78
Figure 47 - Screenshot of the pop-up when the file 'DSC0281.JPG' is clicked	79
Figure 48 - The end time for the website reddit.com on 22 nd July 2010.....	89
Figure 49 - The start time for the website reddit.com on 22 nd July 2010.....	89
Figure 50 - The same example as above with a highlight filter for reddit.com on 22 nd July 2010	89
Figure 51 - Example improvement for reading off times	90
Figure 52 - regularly occurring websites with evenly spaced entries	91
Figure 53 - Same example as above in Net Analysis. The times of the website are circled in red	92
Figure 54 - Net Analysis results are filtering for 'bing' in the URL.....	93
Figure 55 – Heat-map showing the most common times the suspect was on the internet	94
Figure 56 - Where the results appear on the SUS & adjective scale. Image modified from (Kortum & Bangor, 2009)	97

TABLE OF TABLES

Table 1 - The structure of a activity record in a cache file	18
Table 2 - The structure of a activity record in a file.....	19
Table 3 - Fields in table in	21
Table 4 - Fields in table in	22
Table 5 - Fields in table in	23
Table 6 - Fields in table in	25
Table 7 - Fields in table in	25
Table 8 - Table showing what each line in means	31
Table 9 - Table of results for Scenario 1	82
Table 10 - Table of results for Scenario 2	83
Table 11 - Summary of Scenario Results.....	95
Table 12 - SUS and adjective scale for Net Analysis and Webscavator.....	97
Table 13 - Results of Webscavator Functionality Evaluation.....	99

TABLE OF GRAPHS

Graph 1 - Bar Chart showing the participant's confidence and errors in their answers for Scenario 1	84
Graph 2 - Bar Chart showing the participant's confidence and errors in their answers for Scenario 2	84
Graph 3 - Bar Chart showing the length of time it took to complete each question and the errors made in Scenario 1	85
Graph 4 - Bar Chart showing the length of time it took to complete each question and the errors made in Scenario 2	86

CHAPTER 1: INTRODUCTION

As the prevalence and usage of networked computer systems increases, the chances of these computer systems being involved with criminal behaviour logically also increases. The field of digital forensics has grown rapidly over the last decade due to the rise of the internet and the associated crimes, and although the ideas are well established, the discipline of digital forensics is still new and developing (Fei, 2007).

Today's digital forensic investigator has "hundreds of specific and unique application software packages and hardware devices that could qualify as cyber forensic tools...hundreds of utilities available for the same task" (Marcella & Menendez, 2007). The basic requirement of any computer forensic tool is to convert specific files into a human readable format and leave the analysing to the forensic investigator. As this dissertation will show, the analysis is often quite difficult and time consuming.

Data visualisation is often an area neglected by programmers because it requires more skills and time to do properly, and does not seemingly add any new functions to the software since it just shows the resultant data in a new way (Hitz, Robadey, & Ingold, 2000). However, good visual interfaces and visualisations can vastly improve the problems described.

This project will explore using visualisations for forensic investigations, using web history analysis as an example area in much need of visual tools. This project will create a visual tool called Webscavator, and evaluate its effectiveness against pre existing non-visual tools.

1.1 WEB HISTORY ANALYSIS

Web history analysis is a common and important part of a digital investigation. It involves examining and analysing what users of a computer system have been viewing on the World Wide Web through their internet web browsers. Critical evidence can be found in the suspect's browser history, including websites visited, searches conducted and web-based e-mail. In cases involving crimes predominately carried out using web browsers, such as child pornography, identity theft and credit-card fraud, web history analysis can play a vital role in convicting or exonerating a suspect. In other cases, web history may provide an alibi, a general user profile or behavioural patterns. Web history can also provide crucial evidence in crimes unrelated to computers, such as in the case of Neil Entwhistle who was convicted of murdering his wife and baby daughter after forensic investigators found a Google search for "how to kill with a knife" in his computer's web history (AFP, 2008).

There are many web browsers available, each having different methods of storing a user's web usage history. The most common browsers are Internet Explorer¹, Firefox², Google Chrome³, Safari⁴ and Opera⁵. The history files produced by the browsers are not in a human readable format and parsing them requires external tools. For example, Internet Explorer stores usage history in several `index.dat` files, located in various different places depending on the version of Windows. The latest version of Firefox stores its web history in SQLite databases.

1 <http://www.microsoft.com/windows/Internet-explorer/default.aspx>

2 <http://www.mozilla-europe.org/en/firefox/>

3 <http://www.google.co.uk/chrome>

4 <http://www.apple.com/safari/>

5 <http://www.opera.com/>

There are several forensic tools that currently analyse browser history, some specializing in a particular browser (such as Pasco⁶ for Internet Explorer) and others which analyse a few different browsers (such as Mandiant's WebHistorian⁷ for Internet Explorer, Firefox, Safari and Opera). Complete forensic toolkits such as EnCase⁸ and Forensic Tool Kit⁹ (FTK) also have the ability to analyse many different browsers.

1.2 LAYOUT OF DISSERTATION

This chapter has introduced the area of web history forensics and the motivation for this dissertation. Chapter 2 describes the research questions and the methodology carried out to answer the questions. This includes how usability studies with computer forensic investigators will be conducted and the design, implementation and testing considerations of the application.

The next three chapters are a literature review: Chapter 3 describes the five main web browsers available today and how they store the user's online history; Chapter 4 describes some of the main forensic tools used to extract web browser history and their benefits and limitations and Chapter 5 explores the different types of visualisations that exist and reviews current research in visualisations for computer forensic investigations. Chapter 4 draws upon an analysis of current tools and interviews with forensic investigators and their thoughts on the tools they use.

Chapter 6 describes Webscavator, the visualisation tool developed. The details of the application such as user and developer documentation can be found in the appendix. Chapter 7 evaluates the results of the usability tests. Finally, Chapter 8 concludes with a summary and further work.

6 <http://www.foundstone.com/us/resources/proddesc/pasco.htm>

7 http://download.cnet.com/Web-Historian/3000-2653_4-10373157.html

8 <http://www.guidancesoftware.com/>

9 <http://www.accessdata.com/forensictoolkit.html>

1.3 TERMINOLOGY USED AND NOTES

The phrase **digital forensics** and **computer forensics** are both used interchangeably to mean *“the science of acquiring, retrieving, preserving, and presenting data that has been processed electronically and stored on computer media”* (Kruse & Heiser, 2001).

A **visualisation** is an image, diagram, graphic or animation that is a representation of some data to give a better understanding of the data.

Webscavator is the name of the application developed here. It is an amalgamation of the words website and excavator.

This dissertation will only consider desktop browsers and not mobile phone or PDA browsers. The examples used will be mostly Windows based, however the underlying file formats and descriptions should be applicable to all operating systems that support the browsers in question.

CHAPTER 2: RESEARCH METHODOLOGY

2.1 PROBLEM STATEMENT

The output from most forensic tools is either a structured file or a spreadsheet embedded in the tools' user interface. In most cases, the embedded spreadsheet can also be exported to a structured file. Web history analysis is a good example where most of the tools available behave in this way.

The basic task of a web browser analyser program is to convert browser history files into a human readable version. Some more sophisticated programs (such as Net Analysis¹⁰) can filter, sort and search the resulting data, but most programs do not perform any functions on the resultant data.

It is up to the forensic investigator to make sense of the data and find what they are looking for. There are many problems faced when presented with large amounts of data. The list below

¹⁰ <http://www.digital-detective.co.uk/netanalysis.asp>

explains some of them in relation to web browser history. These problems are by no means unique to this category.

1. The files are very big. The files stores every single website visited since that browser was first used. Even if the user clears their history or temporary internet files from within the browser, their usage history files are not necessarily cleared (`index.dat` files for example). If the suspect is a heavy internet user and has had their computer for some time, the history information can be hundreds of thousands of entries long. For example, using WebHistorian, a dump of the author's Firefox history produced 43,269 entries for the period from July to December 2009. Going through this many entries carefully is time consuming, labour intensive and error-prone.
2. Each entry is a piece of text. The investigator has to read each field to understand its meaning and significance. With such large files, this is a daunting task, especially if the investigator does not have anything specific to search for.
3. Correlations, patterns and anomalies are difficult to spot between lines of text. For example, similar websites (such as different sub domains of a particular website) do not immediately jump out and nor do usage patterns. In a spreadsheet format, columns can be sorted alphabetically, numerically and by time, however they cannot be easily grouped and categorized effectively as the software does not understand the domain of the data. Similarly, showing anomalies is impossible as generic spreadsheet software does not know what to look for.
4. Questions cannot be asked about the data. Spreadsheets offer limited searching and sorting facilities, but cannot query the data without the domain specific knowledge the investigator has. Basic graphs can be made as crude timelines, however Excel has a limit to the amount

of data points it will draw on a 2D graph (32,000¹¹) which is not even enough for the author's example given above.

5. The data is difficult to present to those who are not technically minded. Forensic investigators need to convey their analysis of the data to police officers, lawyers and members of the jury. This may require a lot of additional explanation if the results are in tabular or spreadsheet format.

Forensic investigators can waste significant amounts of time attempting to interpret these large files which are not correlated or meaningful. They also require high levels of concentration, patience and tolerance for error.

2.2 RESEARCH QUESTIONS

The problems above prompted the following research questions to be covered in this dissertation:

1. *How do current forensic investigators use web history information and what are the current problems with the tools available?*

Digital forensic investigators will be asked what tools they used to analyse web history, what they look for when sifting through the data, and what kind of problems they face. Three investigators will be interviewed, two eDiscovery specialists who work for a large financial company and a forensic investigator for the Scottish Crime and Drug Enforcement Agency. Research will be carried out into what web browsers produce and how current tools display the data. Limitations of these tools will be investigated. This will form part of the literature review for this thesis.

2. *What visualisations would be appropriate?*

¹¹ <http://office.microsoft.com/en-us/excel/HP051992911033.aspx>

Following on from the first question, once it is established what investigators use the data for, appropriate visualisations will be researched. Research into visualisations for other digital forensic areas will be investigated and their effectiveness analysed. This will form the other part of the literature review for this thesis.

3. *Are visualisations for web history useful and an improvement on what is currently available? Can this be generalised?*

To evaluate the usefulness of the visualisations, an application will be built that takes in the output from current web history analysers and visualises them. The same digital forensic investigators interviewed previously will be asked to take part in a usability study and compare the application against the tools they currently use.

This will generate quantitative and qualitative data which will be analysed to see if the visualisations improve the investigators ability to complete the tasks. Finally, it will be asked if the results obtained here can be generalised to other areas of computer forensics.

2.3 APPLICATION METHODOLOGIES

The application produced will be called Webscavator, and it is intended to be open source and easy for developers to add new visualisations.

2.3.1 DESIGN

The application will be a web application and will use the Model-View-Controller (MVC) paradigm. MVC is a popular pattern used for web applications which treat web pages (the view) as an interface to data held in a database (the model) via a controller. Due to the separation of code, each part can be modified without causing significant interference to the others. MVC also increases flexibility and reuse of code, and means that testing can be done on each part individually.

A web-based application was chosen over making a standalone application because the user interface is easier to build and web technologies such as JavaScript offers a plethora of toolkits to create visualisations. Forensic investigators may already be familiar with Autopsy¹², a web-based graphical interface for SleuthKit which works in a similar manner.

2.3.2 IMPLEMENTATION

The application backend will be built using Python¹³. Python has its own web server, allowing the application to be run locally on a specified port. No other networked devices will be able to connect to the local server. The database system used will be Sqlite¹⁴. This simple RDBMS will be used because it is lightweight and minimises the installation burden for the application (Sqlite comes with Python). The web application framework used will be WerkZeug¹⁵, a lightweight Python web toolkit. The visualisations will be built using various JavaScript toolkits. The literature review will help select the most appropriate ones.

2.3.3 TESTING

Unit testing will be carried out on each class using PyUnit¹⁶. WerkZeug also has a suite of testing tools that can create fake web requests which will test all the interfaces. Testing of the visualisations will be done manually.

12 <http://www.sleuthkit.org/autopsy/>

13 <http://python.org>

14 <http://sqlite.org>

15 <http://werkzeug.pocoo.org/>

16 <http://pyunit.sourceforge.net/>

2.4 EXPERIMENTAL DESIGN

2.4.1 USER TESTING

To evaluate the visualisations, a set of usability tests will be completed. The participants will be given two scenarios and a set of questions for each to answer. For each scenario they will try to answer the questions using Net Analysis and Webscavator. Which program is used first will be randomised, and to prevent the participants from applying their answer from the first program to the second a different web history dataset will be given. Which dataset is given with each program will also be randomised. There may be an element of learning once the participants are on the second scenario, so it is expected they may do slightly better.

2.4.2 SCENARIO DESIGN

Two different scenarios will be formulated: one based on commercial forensic investigations and one on police forensic investigations. Since commercial and police investigations are very different, this will provide a more thorough test and each participant in the study will have at least one scenario where they are more familiar with. This will be taken into account if the results show a large difference between the scenario users are familiar with and the one they are not.

The commercial scenario will be 3 months of fake web history for an employee working for a fake company. The premise is that their manager is concerned that in a particular week the employee's productivity has dropped and many deadlines have been missed. They want to know if this is due to the employee surfing the web rather than working. The emphasis on this scenario will be on getting an overall picture of the user, comparing web history volume for different days and seeing what websites they have visited.

The police scenario will be 3 months of fake web history for a person who is suspected of being a mastermind behind a failed robbery at a jewellery store. The police want to know if there is

any evidence of this using their web history. The emphasis on this scenario will be on the user's search engine queries and files they have accessed on their computer.

2.4.3 OUTPUT DATA

To compare the use of the two programs some quantitative data will be collected. The questions asked in each scenario will have right and wrong answers and will be given a rating. The length of time it took to complete each task will also be measured. The participants will also be asked to rate each of their answers with a 'confidence' rating out of 5 (5 = very confident, 1 = total guess). This will make it easy to see if correct answers were fluke or not.

Since not much quantitative data can be obtained, more emphasis will be put on retrieving qualitative data. An unstructured interview will take place after the testing which will include a post-task walkthrough, where the participants can highlight any areas of the programs they thought were particularly easy or difficult to use. The 'confidence' rating will provoke questions in the interview if any of the answers are wrong.

Finally, the investigators will be given a survey based on Brooke's *System Usability Scale*, a mature and extensively used usability questionnaire (Brooke, 1996). This will consist of 20-30 statements comparing Webscavator to Net Analysis using a rating scale, concentrating on the usability of the programs.

CHAPTER 3: WEB BROWSERS

In order to understand what is available for visualisation, an investigation was undertaken into what useful information is stored by web browsers. Simply relying on the output of 3rd party tools is not enough, as these do not always output everything that browsers provide. This chapter describes what each of the most popular desktop web browsers retain about user browsing history and how it is stored.

The five most popular desktop web browsers currently in use are Internet Explorer, Firefox, Chrome, Safari and Opera. Together, these have roughly 96.65% of the overall usage share. Figure 1 shows the median usage share of web browsers in May 2010 amalgamated from various sources (Wikipedia, 2010).

Internet Explorer (IE) is the most widely used browser and is the default in all Windows operating systems apart from Windows 7, where the user has a choice of which browsers to include during installation. All versions of IE store internet history in binary `index.dat` files, making it difficult to view the contents without using a hex editor. Firefox and Chrome store detailed web history in easily viewable databases, and are both open source. Safari and Opera are

propriety browsers, and store minimal web history in simple XML or plaintext files. Safari is the default browser for Apple's OS X operating systems.

In addition to storing history, both Chrome and Safari store the text of visited websites. Chrome organises this in a table to make it easy for the user to search through visited pages, whilst Safari puts everything into one file making it difficult to work out which website the text came from. Safari also stores a screenshot of every non-HTTPS web page visited.

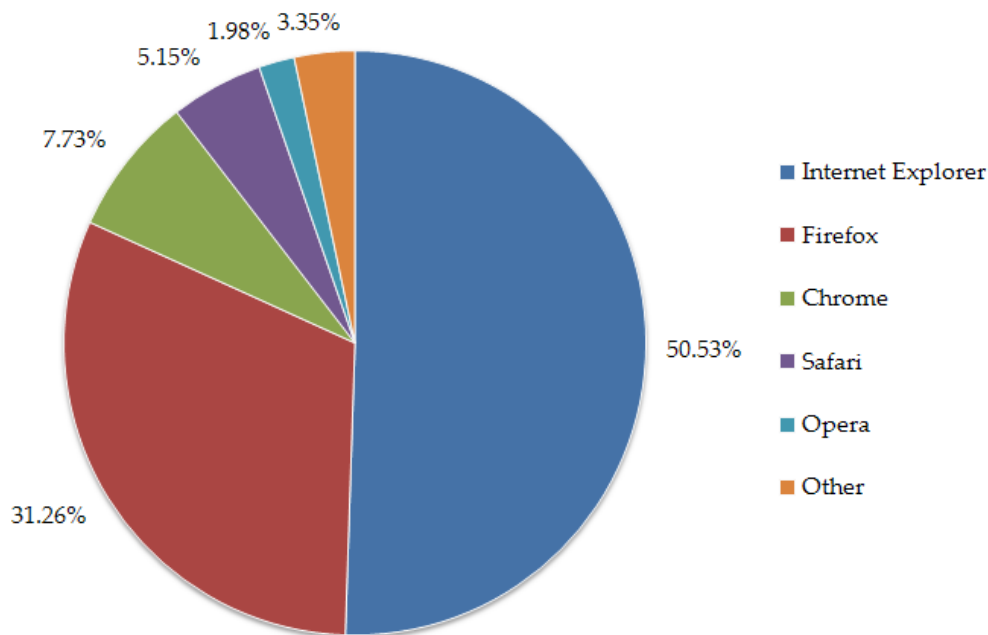


Figure 1 - The median usage share of web browsers

3.1 INTERNET EXPLORER

Internet Explorer stores files downloaded from the internet in a cache called `Temporary Internet Files` (e.g. html pages, images, CSS files). Each cached file is assigned an alphanumeric cache name. Some `index.dat` files serve to map the cached name with the filename and URL it came from. Other `index.dat` files store the user's cookies or web browser history (by default 20 days' worth) (Pereira, 2009). `Index.dat` files are in binary format, and need to be viewed using a hex editor.

There are numerous `index.dat` files kept on Windows machines. Assuming the computer is running Windows XP, the locations of the main `index.dat` files are:

C:\Documents and Settings\<UserName>\Local Settings\History\History.IE5\index.dat

(Older history `index.dat` files can be found in
C:\...\History.IE5\MSHist[18digits])

C:\Documents and Settings\<UserName>\Local Settings\Temporary Internet Files\Content.IE5\index.dat

For Windows Vista and Windows 7, the corresponding paths are:

C:\Users\<UserName>\Local\Microsoft\Windows\History\History.IE5\index.dat

C:\Users\<UserName>\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5\index.dat

The `index.dat` files all have the same format, and comprise of a header followed by a series of records. There are four types of records: `HASH`, `REPR`, `URL` and `LEAK`. `HASH` records are indexes to the other three record types, and can be ignored as they are only used internally by Internet Explorer. `REPR`, `URL` and `LEAK` are called *activity* records, since they each contain information about some sort of online browser activity (Jones, 2003).

There are a few differences between the various `index.dat` files. The one stored in the Temporary Internet Files folder (the “cache `index.dat`” file) is used to relate web files to those cached on the computer, so this additionally stores the names of the cached folders in the file header, and a reference to a corresponding cache folder within each activity record (Mil Incorporated, 2009). Other differences will be explained below.

3.1.1 INDEX.DAT FILE HEADERS

The headers contain a small amount of information about the file and, for a cache `index.dat`, an array of cache folder names.

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f	
00000000																	C
00000010																	lient UrlCache
00000020																	MMF Ver 5.2..@..
00000030	00	3c	d1	7b	01	00	00	00	57	ee	ba	02	00	00	00	00	.P...4..+.....
00000040	fa	41	4d	00	00	00	00	00	04	00	00	00	44	01	00	00	.<N{....Wí°.....
00000050																	úAM.....D...
00000060																	4ZH70LT0C...BZ12
00000070																	5KW0D...\$13CKM30
00000080	01	00	00	00									00	00	00	00	D...A2X32BR5....
00000090	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
000000a0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

Figure 2 – A cache index.dat file header

Figure 2 shows the header of an example cache index.dat file. All index.dat files start with “Client UrlCache MMF” followed by the version number, which is shown in red. Next, in blue, is the size of the file. All numbers are stored little-endian¹⁷. Following on, in yellow, is a pointer to the start of the first record (Jones, 2003). In this example the next part of the header names four subfolders where the cached files are located – shown in green. In non-cache index.dat files, these would be 0x00 (null) values.

3.1.2 INDEX.DAT FILE CONTENTS

There are three types of activity records. These contain URL information and have the following common structure, illustrated by Figure 3:

- **TYPE:** 4 bytes, either URL, LEAK or REDR. Shown in yellow.
- **LENGTH:** 4 bytes, contains the length of the record in 128 byte (0x80) sized blocks.

¹⁷ Little Endian means bytes are stored with the least significant byte first. For example the hexadecimal number 0x001a4000 (1,720,320) would be stored as 0x00 0x40 0x1a 0x00.

- **DATA:** variable length, the data we are interested in. Shown in grey. The end of every record is given by a 0x00 character, which can be seen in blue. The rest of the record is just filled with junk (Jones, 2003).

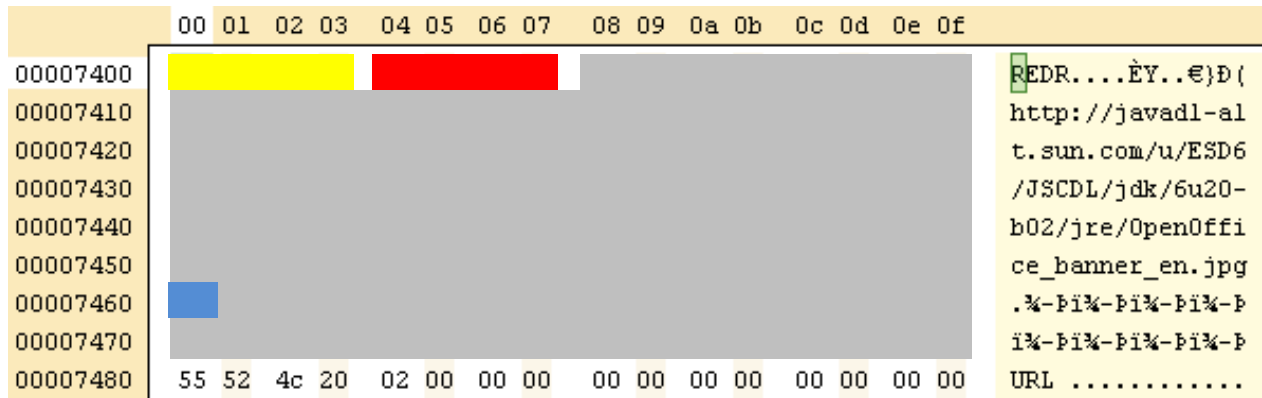


Figure 3 - A REDR activity record in a cache index.dat file

3.1.2.1 REDR ACTIVITY RECORDS

REDR records contain just a URL and indicate a redirect to a different location.

3.1.2.2 URL ACTIVITY RECORDS

These are the important records and an example can be seen in Figure 4. The information held in the **DATA** section is dependent on the type of index.dat file. They all start with the last modified time (in blue) followed by the last accessed time (in green) (Jones, 2003). Time is stored in Windows FILETIME format (100-nanosecond intervals since 1st January, 1601 UTC).

If the index.dat is a cache file, like that of Figure 4, the structure follows that of Table 1. If the index.dat is a history file, the structure follows that of Table 2, and looks like Figure 5.

3.1.2.3 LEAK ACTIVITY RECORDS

LEAK activity records look the same as URL activity records. “A LEAK record can be generated by attempting to delete a URL cache entry when the associated temporary internet file cannot be deleted” (Murr, 2009) or more generally, “A Microsoft term for an error” (Bunting, 2010).

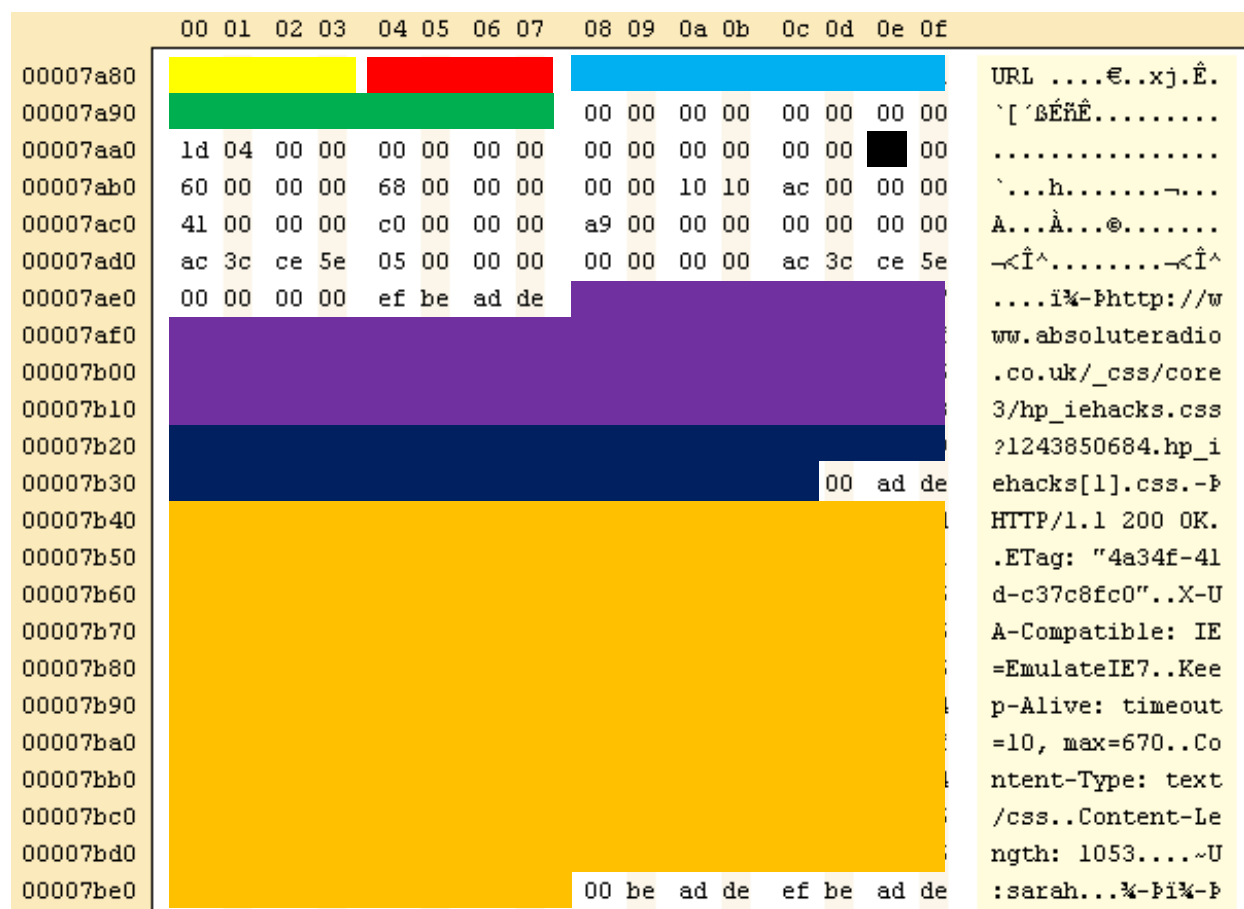


Figure 4 - A URL activity record in a cache index.dat file

Location	Meaning
38 bytes in	Reference to the cache folder the file is located in (Jones, 2003). This is just one byte long and is an index into the array of cache folders given in the file header. Shown in dark grey in Figure 4.
96 bytes in	<p>The URL the file came from (shown in purple). This is followed by the name of the corresponding cached file stored on disk (orange) and finally the HTTP headers (dark blue). Each part starts on a new 16 byte boundary (Jones, 2003).</p> <p>The Windows username is attached to the end of the HTTP headers (Mil Incorporated, 2009).</p>

Table 1 - The DATA structure of a URL activity record in a cache index.dat file

Location	Meaning
96 bytes in	A URL starting with "Visited: <user>@". This is a URL the user with the login name <user> has visited using their Internet Explorer browser (shown in purple in Figure 5).

Table 2 - The DATA structure of a URL activity record in a History.IE5 index.dat file

	00	01	02	03	04	05	06	07	08	09	0a	0b	0c	0d	0e	0f	
00014c00																	URL
00014c10									e4	3c	6c	6e	00	00	00	00	»«...Ë.ä<ln....
00014c20	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
00014c30	60	00	00	00	68	00	00	00	fe	00	10	10	00	00	00	00	`...h...p.....
00014c40	01	00	20	00	b0	00	00	00	14	00	00	00	00	00	00	00
00014c50	c8	3c	6c	6e	01	00	00	00	00	00	00	00	00	00	00	00	È<ln.....
00014c60	00	00	00	00	ef	be	ad	de								i%-PVisited:
00014c70																	Sarah@http://ww
00014c80																	w.hhdsoftware.co
00014c90																	m/dispatch/hex/t
00014ca0													00	be	ad	de	rialpay.html.%-P

Figure 5 - A URL activity record in a History.IE5 index.dat file

3.2 FIREFOX

Firefox version 3 (first released in 2008) employs a different and much richer system for storing browser history than its predecessor Firefox 2. Since only 2.75%¹⁸ of Firefox users still use version 2 or lower, only Firefox 3 will be considered here, and will henceforth be referred to as Firefox. Firefox uses SQLite database files to store browser history, bookmarks, cookies, downloads, form field entries and web site logins. SQLite is "a software library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine" (SQLite, 2010).

18 <http://marketshare.hitslink.com/browser-market-share.aspx?qprid=2&qpmr=40&qpdt=1&qpct=3&qpcal=1&qptimeframe=M&qpsp=136&qpnsp=1>

Assuming the computer is running Windows XP, the default path to the databases is:

C:\Documents and Settings\<user>\Application Data\Mozilla\Firefox\Profiles\<profile folder>

For Windows Vista and Windows 7, the default path is:

C:\Users\<user>\AppData\Roaming\Mozilla\Firefox\Profiles\<profile folder>

The Firefox databases are stored in several different `sqlite` files, which can be viewed using an SQLite viewer such as SQLite Database Browser¹⁹. Several of these files hold important forensic data, the most useful being `places.sqlite` and `formhistory.sqlite`, which respectively store the user's browsing history and web form history.

Firefox 3 stores history for a default of 90 days, which is significantly more than the default of 9 days for Firefox 2 (Pereira, 2009).

3.2.1 PLACES.SQLITE

`places.sqlite` is the main web history database and stores user bookmarks and accessed URLs. The database contains 11 tables. The two most important ones are `moz_places` and `moz_historyvisits`. The fields in `moz_places` are described in Table 3 and the fields of `moz_historyvisits` are described in Table 4.

Field	Meaning
<code>id</code>	The table's primary key. This is used in a lot of other tables to reference rows of this table.
<code>url</code>	Stores a unique visited URL.
<code>title</code>	Stores the URL's page title.
<code>rev_host</code>	Stores the reversed host name.
<code>visit_count</code>	Stores the total visit count for this URL.
<code>hidden</code>	Indicates if the URL will be displayed by the autocomplete function. A

¹⁹ <http://sourceforge.net/projects/sqlitebrowser/>

	value of 1 will keep it hidden and 0 will display it. Examples of hidden URLs are queries and RSS feeds (Pereira, 2009).
typed	Indicates if the URL was typed into the address bar or not. A value of 1 means it was manually entered.
favicon_id	A foreign key to the favicon table which stores the favicon ²⁰ for each URL.
frecency	Frecency is an amalgamation of the words frequency and recency. Frecency is “a score given to each unique URI in Places, encompassing bookmarks, history and tags. This score is determined by the amount of revisitation, the type of those visits, how recent they were, and whether the URI was bookmarked or tagged” (Mozilla Developer Center, 2008a). This value is used by Firefox’s autocomplete. URLs start with a value of -1, and the higher the frecency the higher in the autocomplete the URL will appear. Values of 0 are ignored (and have a value of 1 for hidden).
last_visit_date	Stores the last time the URL was visited. This is a 64bit integer storing number of microseconds since 1 st January 1970 UTC called PRTIME (Mozilla Developer Center, 2008b).

Table 3 - Fields in moz_places table in places.sqlite

Field	Meaning
id	The table’s primary key.
from_visit	Stores the id of the referring URL. If there is no referrer, stores the value 0.
place_id	Stores a foreign key to the moz_places table.
visit_date	Stores the time the URL was visited in PRTIME.
visit_type	Shows how the URL has been accessed. There are seven possible values. The most common are: 1 – the user followed a link and got a new top-level window; 2 – the user typed in the URL or selected it from autocomplete results) and 3 – the user clicked on one of their bookmarks to get to the page (Mozilla Developer Center, 2005).

20 A favicon is an icon associated with a particular web page, and typically shown as a small image in the address bar of a browser viewing the site

session	Stores the session ID that the URL belongs to (Mozilla Developer Center, 2005).
---------	---

Table 4 - Fields in moz_historyvisits table in places.sqlite

Using `from_visit` and `place_id` it is possible to retrace a user's steps and see how they got to a particular page. Using Figure 6 and Figure 7, it can be shown that an example user accessed three additional pages on the website `http://last.fm` (an online music website) after they accessed it for the first time. It can also be seen that in between these visits, the user also searched on Google for lyrics and followed a link. This intermixing can happen because the user was using multiple tabs or had two instances of Firefox open to access multiple websites at the same time. This is confirmed by the `session` being different for both sets of URLs. `moz_places` only stores the unique URLs accessed, but combined with `moz_historyvisits` a full account of the user's online history can be gathered.

	id	from_visit	place_id	visit_date	visit_type	session
13	77889	0	12356	1260532309593000	3	427
14	77891	0	54192	1260532820921000	2	428
15	77892	77891	54193	1260532838000000	1	428
16	77893	77889	12358	1260532856859000	1	427
17	77894	0	54194	1260533169859000	1	429
18	77895	77894	54195	1260533173218000	1	429
19	77896	77895	54196	1260533169843000	6	429
20	77904	77894	54203	1260533212640000	1	429
21	77905	77904	54204	1260533209093000	6	429
22	77906	77905	54205	1260533209453000	6	429
23	77907	77892	54206	1260533230562000	1	428
24	77908	77907	54207	1260533237031000	1	428
25	77909	77889	12358	1260533249078000	1	427
26	77910	77889	54208	1260533296093000	1	427
27	77911	0	36335	1260533395875000	1	430

Figure 6 - Part of moz_historyvisits table showing the user clicking on links

id	url	title	rev	hc
1032	1 54192	http://www.last.fm/music/30+Seconds+to+Mars/_/Night+of+the+Hunter?autos	30 Seconds	mf.tsal
1033	2 54193	http://www.last.fm/search?q=vox+populi&from=ac	Search - Las	mf.tsal
1034	54194	http://www.google.co.uk/search?hl=en-GB&q=30+secodns+to+mars+stranger-	30 secodns	ku.oc.e
1035	54195	http://www.google.co.uk/url?sa=t&source=web&ct=res&cd=1&ved=0CAkQFjAA/	url	ku.oc.e
1036	54196	http://www.lyricskid.com/lyrics/30-seconds-to-mars-lyrics/stranger-in-a-strange-l	Stranger In	moc.dil
1037	54203	http://media.fastclick.net/w/safepop.cgi?cid=194508&mid=365813&sid=50982&	safepop.cgi	ten.kcil
1038	54204	http://rd.apmefb.com/w/safepop.cgi?cid=194508&mid=365813&sid=50982&c=1	safepop.cgi	moc.fb
1039	54205	http://media.fastclick.net/w/safepop.cgi?cid=194508&mid=365813&sid=50982&	safepop.cgi	ten.kcil
1040	3 54206	http://www.last.fm/search?q=stranger+in+a+strange+land+thirty&from=ac	Search - Las	mf.tsal
1041	4 54207	http://www.last.fm/search?q=stranger+in+a+strange+land+&type=all	Search - Las	mf.tsal

Figure 7 - Part of moz_places table showing the corresponding URLs in Figure 6

3.2.2 FORMHISTORY.SQLITE

Form history can provide useful information such as usernames, email addresses, postal addresses and search engine queries. Firefox stores this data in `formhistory.sqlite` which has a single table called `moz_formhistory`, which appears to have no maximum size/time limit (Pereira, 2009). The fields can be found in Table 5.

Field	Meaning
id	The table's primary key.
fieldname	Stores the name of the field on the form.
value	Stores the value the user entered on the form.
timeused	Stores the number of times this value was submitted.
firstused	Stores when the value was submitted for the first time in PRTime.
lastused	Stores when the value was last submitted in PRTime.

Table 5 - Fields in moz_formhistory table in formhistory.sqlite

Google queries appear here with the fieldname as "q". Other possible searches will have fieldnames such as "query", "search" and "search_terms". Some webmail sites use forms to send email, so email subjects and email address recipients may be available too. Usernames and passwords to websites can be found in `signons.sqlite`, but the username and password fields are both stored encrypted, albeit these can be assessed if retrieved via the Firefox browser.

3.3 GOOGLE CHROME

In 2008 Google released most of Chrome's source code as a project called Chromium under a BSD license. Chromium is essentially the same browser as Chrome, but lacks built-in automatic updates and Google branding (Chromium, 2010a).

Assuming the computer is running Windows XP, the Chrome default path to user data is:

C:\Documents and Settings\<User Name>\Local Settings\Application Data\Google\Chrome\User Data

For Windows Vista and Windows 7, the default path is:

C:\Users\<User Name>\AppData\Local\Google\Chrome\User Data

Chrome uses SQLite databases similar to Firefox but with notable differences explained below. All databases are stored in the directory `Default`, with `history` and `web data` being the most important. Chrome does not add extensions to its files; but fortunately SQLite databases have a file header of "SQLite format 3" making them easy to recognise.

Chrome is not consistent with which `datetime` format it uses. Some tables use standard Unix Epoch time (microseconds since 1st January 1970) whilst others use Chrome's own variation of Windows `Filetime` divided by 10 to give the number of microseconds since 1601 (Chromium, 2009a).

2.3.1 HISTORY

Similar to `places.sqlite` in Firefox, `history` contains a table of unique URLs visited called `urls`, and a table for each unique visit called `visits`. Also, like Firefox, it is possible to trace a user's path on the web as Chrome also stores referring URLs and visit types (in a field named `transition`). Table 6 and Table 7 show the fields in `urls` and `visits` respectively.

Field	Meaning
id	The table's primary key. This is used in a lot of other tables to reference rows of this table.
url	Stores a unique visited URL.
title	Stores the URL's page title.
visit_count	Stores the total visit count for this URL.
typed_count	Stores the number of times the URL was typed manually.
last_visit_time	Stores the last time the URL was visited. This is stored in Chrome's time format.
hidden	Indicates if the URL will be displayed by the autocomplete function. A value of 1 will keep it hidden and 0 will display it.
favicon_id	A foreign key to the favicon table which stores the favicon for each URL.

Table 6 - Fields in `urls` table in `history`

Field	Meaning
id	The table's primary key.
url	Stores a foreign key to the <code>urls</code> table.
visit_time	Stores the time the URL was visited in Chrome's time format.
from_visit	Stores the <code>id</code> of the referring URL. If there is no referrer, stores the value 0.
transition	Shows how the URL has been accessed. Unlike Firefox, which just gives a number to represent the type of transition, Chrome has some additional information such as whether this was a client or server redirect and if this URL is the beginning or end of a navigation chain. The transition is a number, and bit-masking is used to extract various specific information (SANS Computer Forensics, 2010). There are 11 page transition types, such as 0 – user followed a link, 1 – user typed in the URL and 6 – this is the user's homepage/startpage (Chromium, 2010b).
segment_id	Stores the segment id. It is not clear what 'segments' are. There are tables called <code>segments</code> and <code>segment_usage</code> also in <code>history</code> , and they appear to store the domain names of accessed URLs along with a total visit count.
is_indexed	Stores whether this URL is indexed or not, a 1 meaning it will be indexed.

Table 7 - Fields in `visits` table in `history`

3.3.2 WEB DATA

Login and autofill data is stored in `web_data`. The table `logins` stores saved usernames and passwords for URLs. The password is stored encrypted, with the username plaintext. The table `autofill` stores values the user has used in forms. This is joined to `autofill_dates` which stores the last used date for the data. There are also tables called `autofill_profiles` and `credit_cards` which contain the user's details and credit card information so they can be automatically added to web forms.

3.3.3 HISTORY INDEX FILES

In the same folder are files named `History-Index-YYYY-MM` which store the textual contents of any page visited. This is used when the user searches through web history with keywords. Over time these files can grow very large, to tens of Gigabytes. This has been issued as a bug in the browser (Chromium, 2009b), but can be of importance to a forensic investigator who can also do keyword searches to get an idea of the contents of the pages the user was visiting.

3.4 SAFARI

Safari has a very simple method of storing browser history compared to Internet Explorer and those that use SQLite databases.

If the computer is running Windows XP, the Safari default path to user data is:

C:\Documents and Settings\<User Name>\Application Data\Apple Computer\Safari

For Windows Vista and Windows 7, the default path is:

C:\Users\<User Name>\AppData\Roaming\Apple Computer\Safari

For Apple OS X computers, the default path is:

/Users/<User Name>/Library/Safari/

3.4.1 HISTORY.PLIST

History is stored in a standard OS X binary property list file, `History.plist`. History is only stored for a month by default, but this can be changed to a year or manual deletion. The data can easily be viewed in a hex editor, but can be converted to a structured text file on Macintosh computers by typing in the following command (Craig & Burke, ?):

```
$ defaults read /Users/<username>/Library/Safari/History > history.txt
```

This produces output like that in Figure 8.

```
{
WebHistoryDates = (
    {
        "" = "http://www.twitter.com/";
        D = (
            1
        );
        lastVisitWasFailure = 1;
        lastVisitedDate = "297678774.0";
        redirectURLs = (
            "http://twitter.com/"
        );
        title = "Twitter / Home";
        visitCount = 2;
    },
    {
        "" = "http://www.apple.com/startpage/";
        D = (
            1,
            3,
            3,
            1,
            1,
            2,
            0
        );
        W = (
            12,
            18,
            15,
            31
        );
        lastVisitedDate = "297678768.7";
        title = "Apple - Start";
        visitCount = 87;
    },

```

```
);  
WebHistoryFileVersion = 1;  
}
```

Figure 8 - Sample output from converting `History.plist`

The format for each entry appears to have the URL visited, followed by a list of “D”, an optional list of “W”, an optional ‘last visit was a failure’, the date this URL was last visited, an optional redirect URL, the URL page title and the total visit count. Safari stores the date as the number of seconds since 1st January 2001 GMT (MacScripter, 2006).

It is not clear what the D and W lists are for, as the only reference to `History.plist` on the internet is to an older version of Safari which stored fewer fields. The D and W values add up to the value in `visitCount`, so may represent previous visits although it is not clear how. Even though `History.plist` only stores the last visit date, when viewing web history in Safari it shows all previous dates (but not times) of visits.

3.4.2 OTHER SIGNIFICANT DATA

Other files that are useful are `Bookmarks.plist` and `Form Values.plist`. `Bookmarks.plist` contains a list of the user’s bookmarks, and `Form Values.plist` the user’s form inputs – however this file appears to store data in an encrypted format (Cunningham, 2007).

Safari also keeps a thumbnail and screenshot of every website visited (apart from secured pages such as HTTPS) in a folder called `Webpage Previews`. This is used for displaying the user’s top 10 websites and web history within the browser, as can be seen in Figure 9. This folder could be immensely valuable to a forensic investigator who has an exact screenshot of what the user was looking at, especially as it may take screenshots of any webmail accessed (Ferox, 2009). However it is difficult to match screenshots with web pages in `History.plist` without going via the Safari history browser as there does not seem to be an obvious correlation, and the screenshots do not include the URL.

Finally in a folder called `History` a file called `_<letter>.cfs` stores the text from websites visited. Both this file and the `Webpage Previews` folder can grow very large if the user uses the web heavily or turns off the default 1 month period of storing screenshots and plaintext.

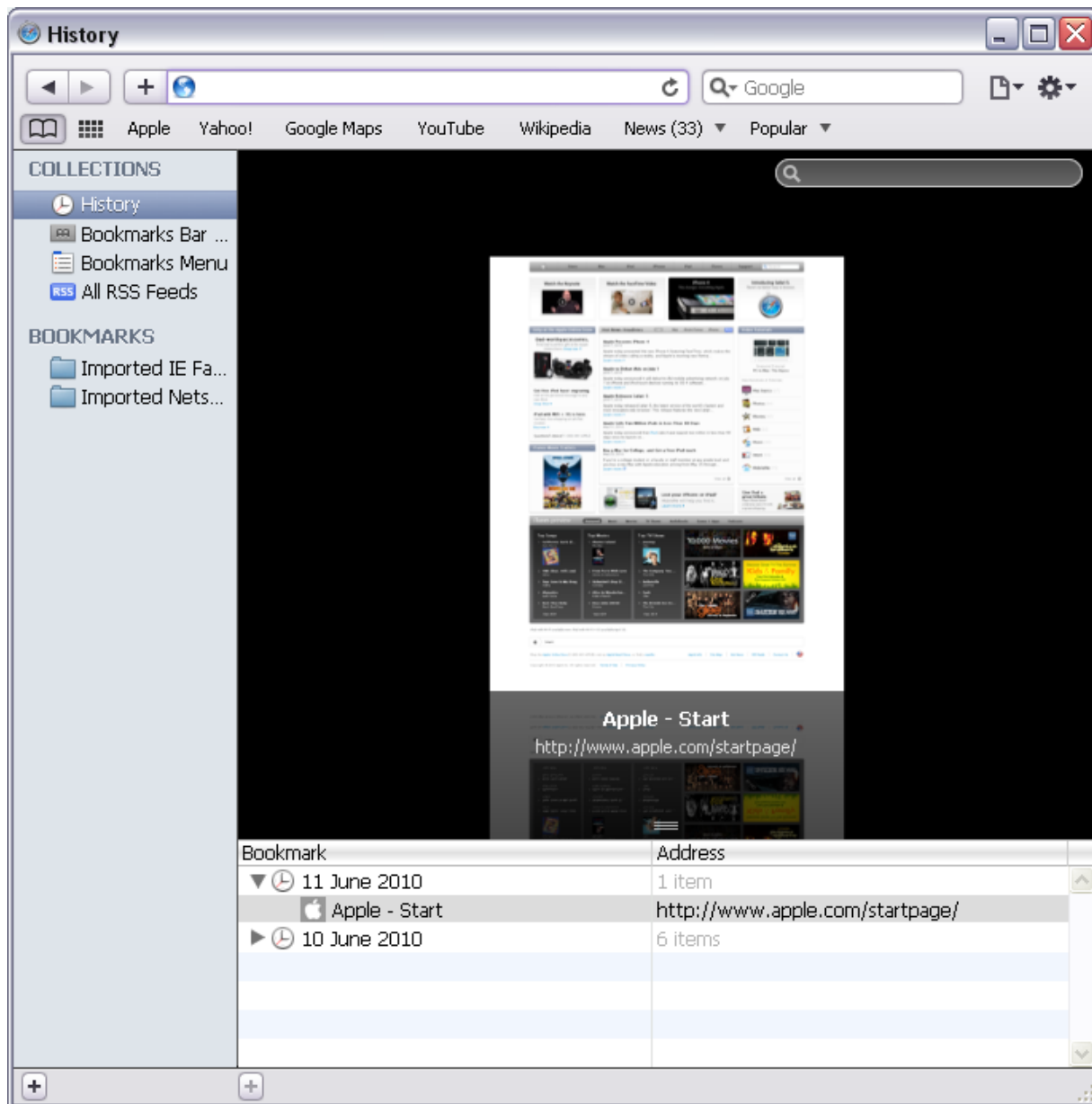


Figure 9 - Viewing web history in Safari. The image displayed is a cached thumbnail of the website.

3.5 OPERA

Out of all the popular browsers, Opera leaves behind the least amount of useful information for investigators. Not only is the data stored in plaintext format, but it does not record every URL visited, only the latest one. Therefore it is impossible to tell how often someone has visited a particular website. Even when viewing web history from within the browser, only the latest entry is shown, giving a false impression of the actual history. For example if someone went to exactly the same websites two days in a row, the first day would have no history associated to it, since each entry would be overridden by the latest visit.

If the computer is running Windows XP, the Opera default path to user data is:

C:\Documents and Settings\<User Name>\Application Data\Opera\Opera

For Windows Vista and Windows 7, the default path is:

C:\Users\<User Name>\AppData\Roaming\Opera\Opera

There are two important files in this directory, `global_history.dat` and `typed_history.xml`.

3.5.1 GLOBAL_HISTORY.DAT

`global_history.dat` is a plaintext file which stores details for each URL visited. Each entry takes up four lines as described in Table 8.

Figure 10 shows some entries in `global_history.dat` and Figure 11 the same entries shown in Opera's history viewer. It is unclear what the popularity field in the file (highlighted in blue) and the viewer means, and how 1936804 corresponds to 2 and -1 to 1. They do not correspond to number of visits.

Line	Meaning
1	The title of the website.
2	Website URL.
3	Time of the last visit. This is the number of seconds after the Unix Epoch UTC.
4	An integer representing the 'popularity' of the website (Gudjonsson, 2010). This number seems to be set to -1 initially, and rises to several million if the website has been visited more than once. It is not clear how this relates to the popularity column when you view web history via the Opera browser.

Table 8 - Table showing what each line in `global_history.dat` means

```
http://cuteoverload.com
```

```
http://cuteoverload.com
```

```
1276255696
```

```
1936804
```

```
BBC NEWS | News Front Page
```

```
http://news.bbc.co.uk
```

```
1276255696
```

```
-1
```

Figure 10 - Example entries in `global_history.dat`

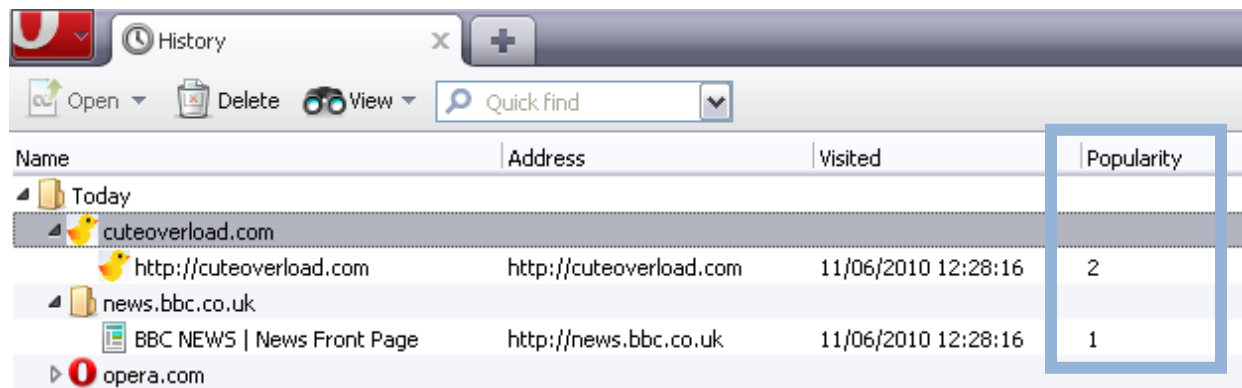


Figure 11 – Same entries as in Figure 10 viewed via Opera's history viewer

3.5.2 TYPED_HISTORY.XML

Typed_history.xml is an XML file that has an entry for each URL that was typed in manually. Each entry is in the format:

```
<typed_history_item content="url" type="type" last_typed="date" />
```

Type is either 'text' or 'selected', where 'selected' means the URL was chosen from Opera's autocomplete and 'text' means it was typed in manually. Again, this only stores the latest URL and not all of them.

CHAPTER 4: FORENSIC WEB HISTORY PROGRAMS

Today's digital forensic investigator has a plethora of tools that take in the web history log files described in Chapter 3 and transform them into human readable data. An investigation into some of these tools was performed in order to discover limitations in how these tools present information that may be addressable with visualisations. The investigation was carried out by examining and using the tools with example browser history files, and interviewing computer forensic investigators to find out what they thought of the tools they used.

The most commonly used tools (Pasco²¹, WebHistorian²² and Net Analysis²³) are described below. Other tools investigated were Firefox 3 Extractor²⁴, FoxAnalysis²⁵, ChromeCacheView²⁶, OperaCacheView²⁷, IECacheView²⁸ and Firefox Forensics²⁹.

21 <http://www.foundstone.com/us/resources/proddesc/pasco.htm>

22 http://www.mandiant.com/products/free_software/web_historian/

23 <http://www.digital-detective.co.uk/netanalysis.asp>

24 <http://www.firefoxforensics.com/f3e.shtml>

25 <http://forensic-software.co.uk/foxanalysis.aspx>

Tools such as Pasco just convert web browser history files into a tab-delimited text file. The results are then viewed using a separate program, usually a spreadsheet application. More advanced tools such as Net Analysis and WebHistorian show their results in a table and allow the user to search, sort and filter the results without resorting to an external application.

There are a few parts of web history files that are not well understood. For example, the tools which process Safari `history.plist` files are none the wiser about the 'W' and 'D' fields, and ignore them. Opera's 'popularity' field in its `global_history.dat` file is also ignored.

4.1 TOOLS COMPUTER FORENSIC INVESTIGATORS USE

Three digital investigators were interviewed to find out what tools they used and the problems they faced with them. One investigator works for the Scottish Crime and Drugs Enforcement Agency (SCDEA) and uses Net Analysis and WebHistorian. The other two investigators work for a large financial company and do not use any specific web history tools, but use EnCase for all their investigations. EnCase is often cumbersome and difficult to use and more often than not the investigators export the output to Excel and use that to analyse the web data.

Several common problems were found:

- Information overload. Net Analysis has 45 columns and shows all the data in one large table, requiring both vertical and horizontal scrolling to see all the data. EnCase and WebHistorian also have many columns.
- It is difficult to show and explain findings to police officers and other non-technical people or those unfamiliar with the software. A lot of explanation is required to highlight the relevant data and explain what it means.

26 http://www.nirsoft.net/utils/chrome_cache_view.html

27 http://www.nirsoft.net/utils/opera_cache_view.html

28 http://www.nirsoft.net/utils/ie_cache_viewer.html

29 http://www.machor-software.com/firefox_forensics

- Current searches within WebHistorian and Net Analysis do not allow for regular expressions or fuzzy searches. It is often easier to search and sort in Excel than in EnCase.
- It is difficult to spot anomalies such as a 'dodgy' website amongst lots of mundane websites. URLs from automated RSS feeds, tickers and adverts obscure genuine browsing history. It is difficult to filter this clutter. A lot of time is spent by the investigators 'cleaning up' the data and making it more relevant and readable.

4.2 PASCO

Pasco is a simple tool that converts Internet Explorer `index.dat` files into tab delimited text files. Pasco is included in the Open Computer Forensics Architecture³⁰ (OCFA), a computer forensics framework built by the Dutch National Police. Pasco is run using the following command:

```
$ pasco index.dat > internet_explorer_history.txt
```

By default the output is a tab-delimited CSV file, which can be imported into a spreadsheet application like Excel. The SCDEA investigator said programs that do nothing but convert web history are not used because the vast quantity of data they produce makes it infeasible to work with the output in external programs such as Excel. However, both the other interviewees found exporting data from EnCase to an Excel spread-sheet made examining and processing the results easier.

³⁰ <http://freshmeat.net/projects/ocfa>

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	TYPE	URL	MODIFIED TIME	ACCESS TIME	FILENAME	DIRECTORY	HTTP HEADERS						
2	URL	http://i.real.com/g/pics/space.gif	02/09/2001 23:44	02/02/2005 15:04	space[1].gif	456NWTMR	HTTP/1.0 200 OK ETag: "4201c6-2b-3a8480eb" Content-Le						
3	URL	http://windowsupdate.microsoft.com/red	10/21/2004 04:14:49	01/24/2005 16:39:59	redirect[1].js	4963KDYV	HTTP/1.0 200 OK Content-Length: 8891 Content-Type: ap						
4	URL	http://v5.windowsupdate.microsoft.com/v	06/15/2004 21:48:24	01/24/2005 16:40:10	toc[1].js	CLABK1AB	HTTP/1.0 200 OK Content-Length: 12378 Content-Type: ap						
5	URL	http://office.microsoft.com/global/images	06/24/2004 03:50:55	01/25/2005 11:16:55	ZA011357811033[1].gif	456NWTMR	HTTP/1.0 200 OK Content-Length: 8189 Content-Type: im						
6	URL	http://v5.windowsupdate.microsoft.com/s	08/04/2004 00:05	01/24/2005 16:15:40	wuident[1].cab	CLABK1AB	HTTP/1.0 200 OK Content-Length: 7314 Content-Type: ap						
7	URL	http://v5.windowsupdate.microsoft.com/v	06/11/2004 02:51	01/24/2005 16:40:18	News_bg_RightTop[1].gif	456NWTMR	HTTP/1.0 200 OK Content-Length: 84 Content-Type: imag						
8	URL	http://office.microsoft.com/en-gb/FX0103	01/25/2005 11:33:38	01/25/2005 11:33:45	FX010354621033[1].htm	OH6F8TA3	HTTP/1.0 200 OK Content-Length: 16739 Content-Type: te						
9	URL	http://realguide.real.com/4plus/main_.js	04/18/2002 20:08:34	02/02/2005 15:04	main_[1].js	OH6F8TA3	HTTP/1.0 200 OK P3P: CP="CAO NON DSP NID ADM DEV C						
10	URL	http://v5.windowsupdate.microsoft.com/v	01/24/2005 16:40:08	01/24/2005 16:40:08	mstoolbar[1].htm	OH6F8TA3	HTTP/1.0 200 OK X-Powered-By: ASP.NET Content-Type:						

Figure 12 - Example output from Pasco shown in Excel

4.3 WEB HISTORIAN

Web Historian is a free tool from Mandiant, an information security company which provides incident response and computer forensics to several US law firms (Mandiant, 2010). Web Historian converts Internet Explorer `index.dat` files, Firefox and Chrome SQLite databases. Web Historian is used by the SCDEA, mainly because their default program, Net Analysis, does not yet support Chrome.

WebHistorian scans the computer for all browser files, and produces a table of results like that of Figure 13. All the browser files found are analysed in one run, as multiple browser history files will exist on a single computer (even when using the default Internet Explorer browser, multiple `index.dat` files exist). There are a total of 18 columns, most of which are relevant to Firefox and Chrome and not Internet Explorer. Viewing all of these requires a lot of horizontal scrolling. WebHistorian only shows 100 rows at a time, with no option to view more or less, which makes it frustrating when having a quick scroll through all the data.

WebHistorian was the only tool to use Firefox's and Chrome's referral URL field and to output the method via which the URL was accessed (for example, from a bookmark or from an HTML link). Even though WebHistorian shows the referring URL, this is not linked to a specific history entry, despite the fact this information is provided by Firefox and Chrome. This makes it difficult to reconstruct the exact path the user used to navigate to a particular URL (although guesses can be made using timestamps). For example, there may be multiple history entries for

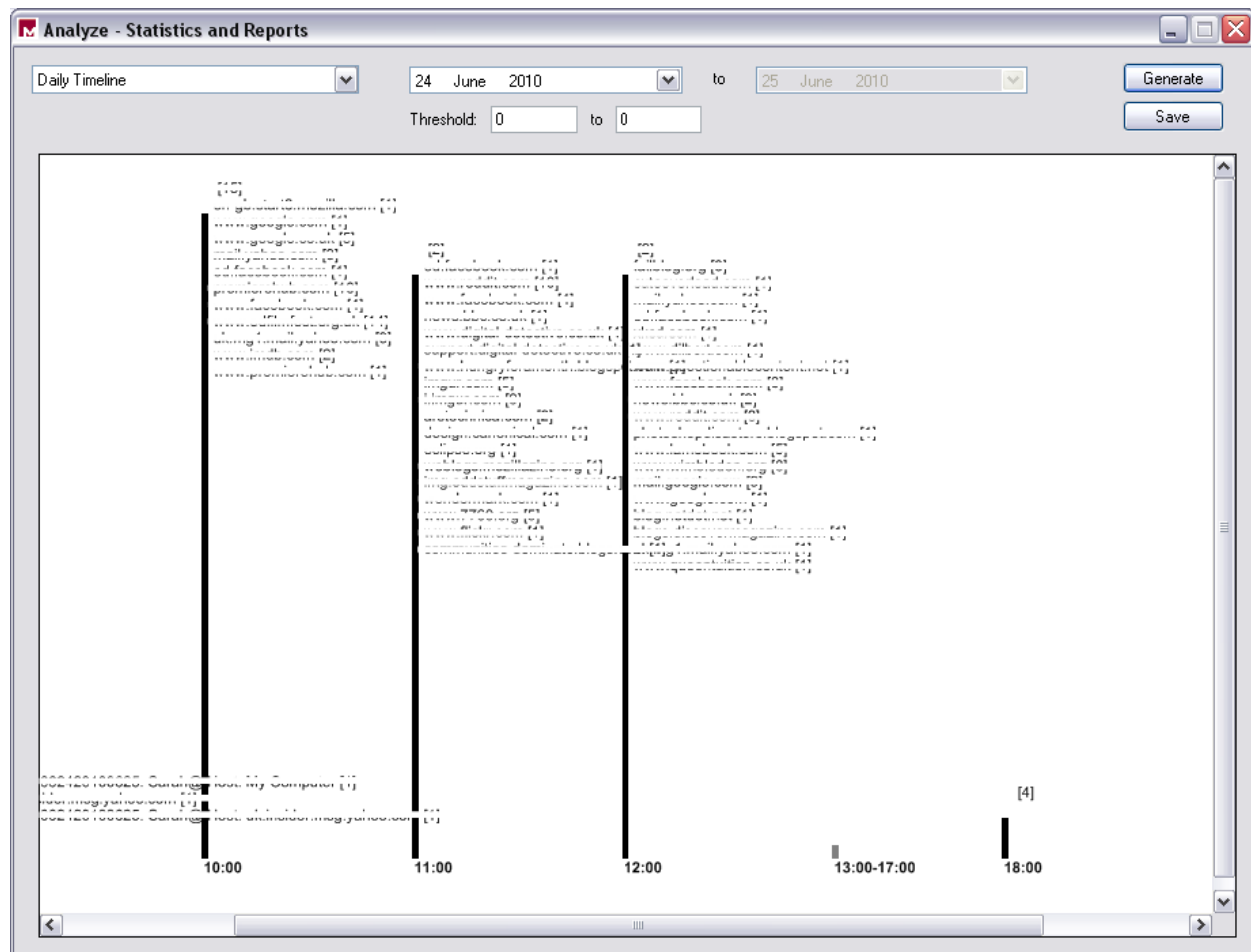


Figure 15 - WebHistorian's Timeline for 24th June 2010

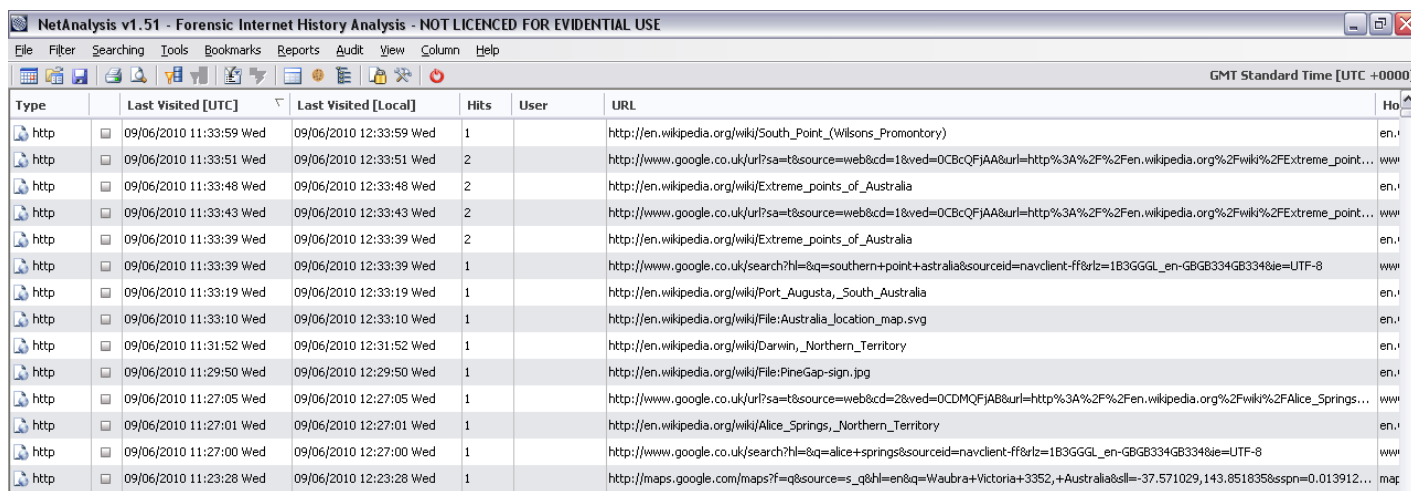
4.4 NET ANALYSIS

Net Analysis is an advanced web browser history tool developed by a digital forensics investigator for the UK police. It *“has become the industry standard software for the recovery and analysis of Internet browser artefacts”* (DigitalDetective, 2010) and is used by many departments including Strathclyde Police, the SCDEA and IBM Incident Response Team.

Net Analysis converts most of the common files produced by web browsers including Internet Explorer (all versions), Firefox (all versions), Opera (versions 9 and 10) and Safari, as well as some more unusual ones such as Yahoo! BT Browser and AOL. Chrome is not yet supported (as of Net Analysis version 1.51).

Net Analysis output consists of a large table with 45 columns, encompassing every field available in all the combined web browsers it supports. Whilst some of the columns will always be populated (e.g. *Last Visited* and *URL*), most are only applicable to certain web browsers (e.g. *Fav Icon URL*). Data that is not well understood (such as the D and W columns in Safari `history.plist`s) are put in the *Status* column. Although there is a column named *referral URL*, this is not populated when using Firefox data, and instead this information is dumped in the *Status* column in an awkward format, eg. "Session: 3876, Placed ID: 127572, Hidden: 0, Freccency: 100".

Due to the large number of columns and entries, a lot of vertical and horizontal scrolling must be done to view all the data. An example can be seen in Figure 16. Using a 1280x1024 screen only the first 7 out of 45 columns can be seen. Net Analysis does have options to hide/show each column, but no quick way of hiding all those which are blank.



Type	Last Visited [UTC]	Last Visited [Local]	Hits	User	URL
http	09/06/2010 11:33:59 Wed	09/06/2010 12:33:59 Wed	1		http://en.wikipedia.org/wiki/South_Point_(Wilsons_Promontory)
http	09/06/2010 11:33:51 Wed	09/06/2010 12:33:51 Wed	2		http://www.google.co.uk/url?sa=t&source=web&cd=1&ved=0CBcQFjAA&url=http%3A%2F%2Fen.wikipedia.org%2Fwiki%2FExtreme_point...
http	09/06/2010 11:33:48 Wed	09/06/2010 12:33:48 Wed	2		http://en.wikipedia.org/wiki/Extreme_points_of_Australia
http	09/06/2010 11:33:43 Wed	09/06/2010 12:33:43 Wed	2		http://www.google.co.uk/url?sa=t&source=web&cd=1&ved=0CBcQFjAA&url=http%3A%2F%2Fen.wikipedia.org%2Fwiki%2FExtreme_point...
http	09/06/2010 11:33:39 Wed	09/06/2010 12:33:39 Wed	2		http://en.wikipedia.org/wiki/Extreme_points_of_Australia
http	09/06/2010 11:33:39 Wed	09/06/2010 12:33:39 Wed	1		http://www.google.co.uk/search?hl=&q=southern+point+australia&sourceid=navclient-ff&rlz=1B3GGGL_en-GBGB334GB334&ie=UTF-8
http	09/06/2010 11:33:19 Wed	09/06/2010 12:33:19 Wed	1		http://en.wikipedia.org/wiki/Port_Augusta,_South_Australia
http	09/06/2010 11:33:10 Wed	09/06/2010 12:33:10 Wed	1		http://en.wikipedia.org/wiki/File:Australia_location_map.svg
http	09/06/2010 11:31:52 Wed	09/06/2010 12:31:52 Wed	1		http://en.wikipedia.org/wiki/Darwin,_Northern_Territory
http	09/06/2010 11:29:50 Wed	09/06/2010 12:29:50 Wed	1		http://en.wikipedia.org/wiki/File:PineGap-sign.jpg
http	09/06/2010 11:27:05 Wed	09/06/2010 12:27:05 Wed	1		http://www.google.co.uk/url?sa=t&source=web&cd=2&ved=0CDMQFjAB&url=http%3A%2F%2Fen.wikipedia.org%2Fwiki%2FAllice_Springs...
http	09/06/2010 11:27:01 Wed	09/06/2010 12:27:01 Wed	1		http://en.wikipedia.org/wiki/Alice_Springs,_Northern_Territory
http	09/06/2010 11:27:00 Wed	09/06/2010 12:27:00 Wed	1		http://www.google.co.uk/search?hl=&q=alice+springs&sourceid=navclient-ff&rlz=1B3GGGL_en-GBGB334GB334&ie=UTF-8
http	09/06/2010 11:23:28 Wed	09/06/2010 12:23:28 Wed	1		http://maps.google.com/maps?f=q&source=s_q&hl=en&q=Waubra+Victoria+3352,+Australia&sl=-37.571029,143.851835&sspn=0.013912...

Figure 16 - Screenshot of Net Analysis with Firefox 3 data

Net Analysis allows the data to be sorted by column and offers searching and filtering of data. Under advanced filtering (Figure 17), Net Analysis provides pre-made filters for search engine queries, possible username and passwords and possible 'indecent images of child abuse'. This search is rather simple and on the author's Firefox history brought up any URL which had

swearing or the words 'pirate', 'illegal' etc and any synonyms of the word 'child' in the URL's page title.

Although the digital investigator interviewed found it very easy and quick to search for key words using Net Analysis, they did not like the huge number of (mostly unnecessary) columns. Net Analysis does not provide fuzzy searching or regular expression searching, and often results were missed due to suspects making spelling mistakes that were not picked up.

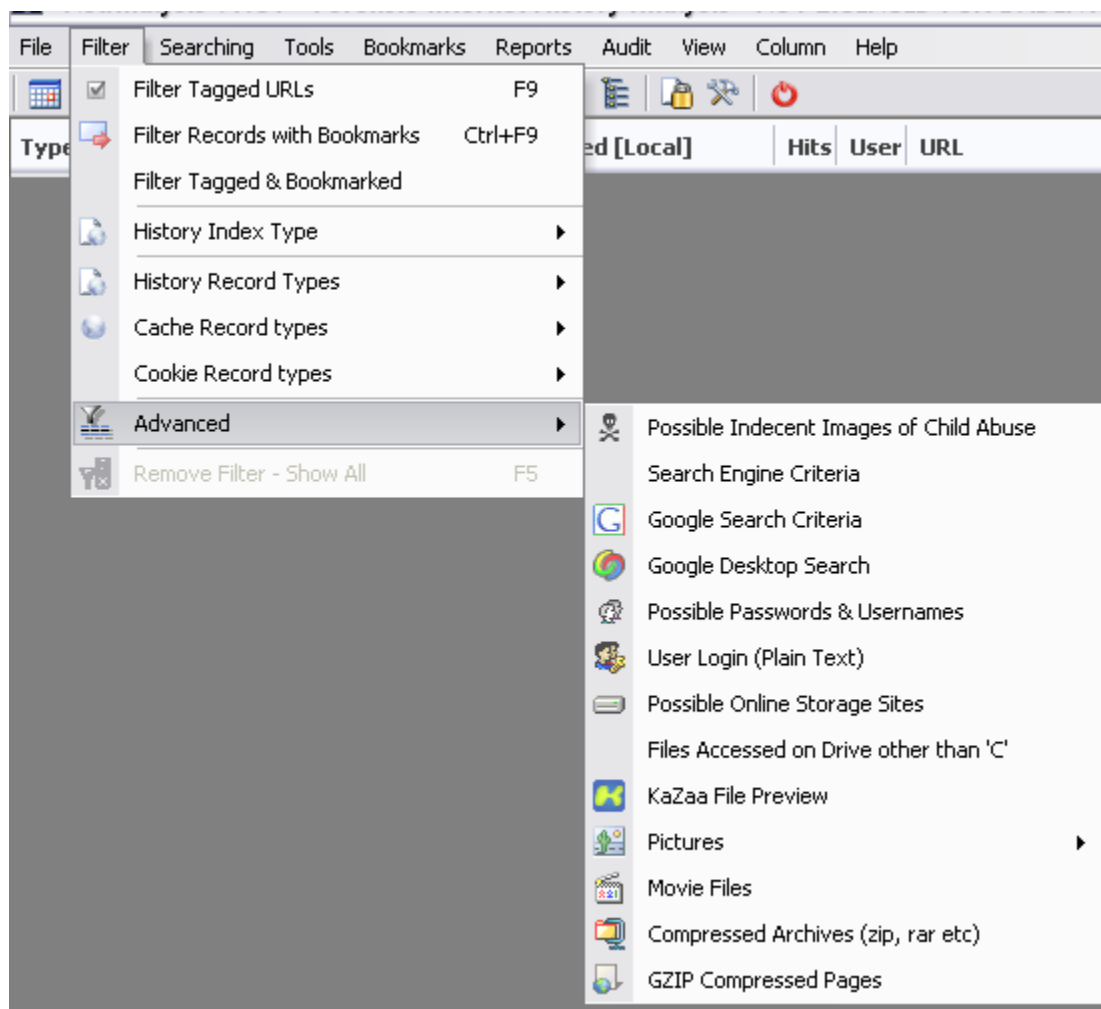


Figure 17 - Net Analysis Advanced Filtering options

CHAPTER 5: VISUALISING DATA

Chapter 4 has shown what tools are currently available for web history analysis and their limitations as perceived by digital investigators. Many of the problems described can be mitigated or even solved by using visualisations. This chapter will explain why visualisations in general improve the ability to understand data.

Commercial providers of digital forensics tools have so far not shown much interest in visualisations. EnCase and FTK – the two major toolkit providers – automate retrieving and processing much of the information from computers to be examined, but do not have any visualisations to help interpret the resulting data. However, there is some academic work on using visualisations to aid forensic investigation of network usage, server logs and file systems. This work is discussed and analysed in this chapter to find out if the visualisations helped forensic investigators perform tasks.

Finally, visualisations that would be most appropriate for web history are investigated. The types of data that web browser log files produce (as seen in Chapter 3) along with the opinions of the interviewed investigators are the main influences on deciding which visualisations are most appropriate.

5.1 VISUALISING DATA

A visualisation is an image, diagram, graphic or animation representing some data that is intended to give a better understanding of that data. There are many different visualisation areas, differing mostly by the domain of the visualised data. Examples include: *mathematical & scientific* visualisations (data from equations and formulas); *product* visualisation & *3D design* (data from images, photos or CAD) and *medical imaging* (data from medical machines such as MRI scanners). *Information* visualisation “is generally applied to the visual representation of large-scale collections of non-numerical information, such as files and lines of code in software systems, library and bibliographic databases, networks of relations on the internet, and so forth” (Friendly, 2009) and is applicable to forensic data.

5.1.1 COGNITIVE BENEFITS

Visual data can act as a temporary storage area for human cognitive processes, so not all the work must be done in the head. By expanding human memory and processing resources, a larger working set for thinking and analysis is provided (North, Stasko, Fekete, & van Wijk, 2008). Graphical representations of mathematical problems, for example, can often increase efficiency in problem-solving by allowing difficult mathematical inferences to be substituted for easier perceptual inferences (Larkin & Simon, 1987).

5.1.2 PERCEPTION BENEFITS

The human visual system is able to perceive graphical information such as pictures, videos and charts in parallel, but text only sequentially (Hendee & Wells, 1997). Images are interpreted much faster than textual descriptions due to the brain processing visual input much earlier than textual input (Teerlink & Erbacher, 2006a).

There are several psychological theories which try to explain why vision is so effective at perceiving shapes and colours. *Preattentive Processing Theory* explains that some visual properties, known as *preattentive properties*, are detected very accurately in less than 250 milliseconds. Since eye movements take at least 200 milliseconds, it is suggested that some

simple visual processing must happen in parallel with this (Healey, 2009). Simple things that can be perceived preattentively include line orientation, length or width, curvature, closure and colour (North, Stasko, Fekete, & van Wijk, 2008). An example can be seen in Figure 18, where it is instantly obvious where the red dot is amongst the blue dots. In Figure 19, where both colour and shape must be distinguished, this takes much longer. Studies have shown that images like Figure 19 cannot be detected preattentively (Healey, 2009).

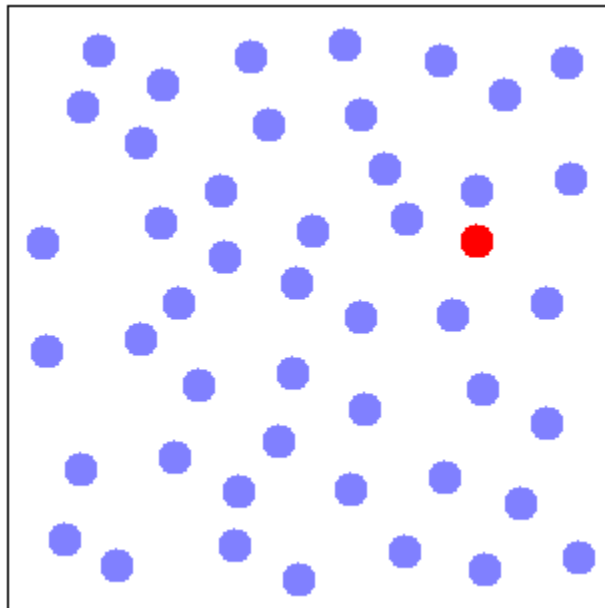


Figure 18 - Find the red dot amongst the blue

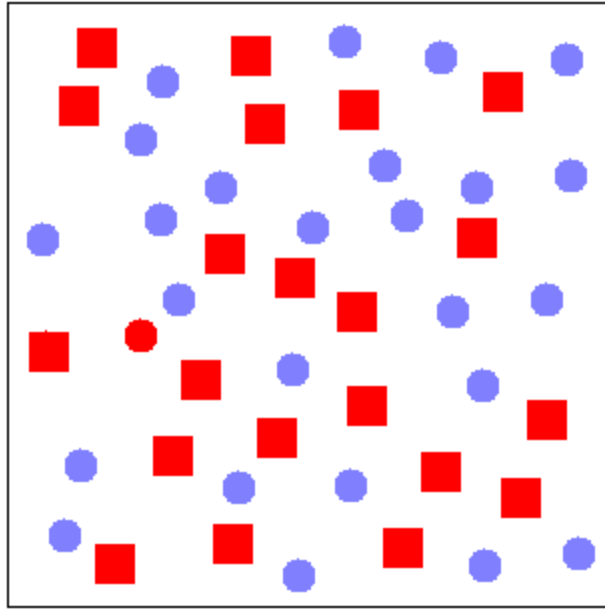


Figure 19 - Find the red dot amongst the blue dots and red squares

Gestalt Theory explains the principles followed by the vision system when it attempts to understand an image, especially groups of objects or patterns. It is based on the following principles (Sternberg & Mio, 2005):

- **Figure ground** – when perceiving a visual field, some things will appear prominent and others will fade into the background
- **Proximity** – objects that appear close to each other will be perceived as a group
- **Similarity** – similar objects will be perceived as a group
- **Continuity** – smoothly flowing / continuous forms will be perceived rather than discontinuous ones (e.g. Figure 20)
- **Closure** – incomplete objects will be perceptually closed up or completed (e.g. Figure 21)
- **Symmetry** – symmetrical objects will be perceived as having a centre and as a whole object

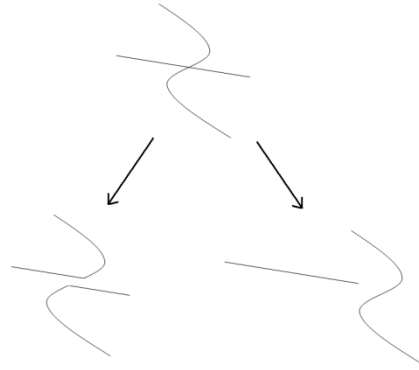


Figure 20 - The figure at the top is perceived to break down into that of the bottom right not the discontinuous bottom left

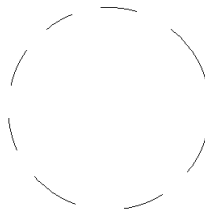


Figure 21 - We perceive a circle even though the image is of disjointed lines

Effective visualisations try to adhere to and take advantage of these principles, e.g. grouping similar items together by colour or icon adheres to the similarity principle (North, Stasko, Fekete, & van Wijk, 2008).

5.1.3 MAKING EFFECTIVE VISUALISATIONS

Visualisations are only effective when the right kind of pictorial representation is chosen and can be manipulated to show useful information. Shneiderman's visual information seeking mantra: *"overview first, zoom and filter, then details-on-demand"* (Shneiderman, 1996) states what every visualisation should do. His mantra is based on the fact that the common goals in information retrieval are to find a narrow subset of items that comply with a query, or to develop an understanding of unexpected patterns within a set of data (Marchionini, 1997).

Shneiderman argues there are only 7 tasks a user will do with information and that visualisations should provide for them:

1. **Overview:** Gain an overview of the data. The visualisation should provide an overall view of the data. If the data is too big, techniques like fish-eye magnification of areas should be used.
2. **Zoom:** Zoom in on items of interest. Users will want to see details and a smooth zooming effect will preserve a sense of context.
3. **Filter:** Filter out uninteresting items. Control widgets are needed to filter out unnecessary information.
4. **Details-on-demand:** Select an item or group and get details when needed. Items or groups should be clickable to reveal more details in an information area or pop-up bubble.
5. **Relate:** View relationships among items. The visualisation should be able to provide tags to group similar items or highlight items that have the same attributes as the one selected.
6. **History:** Keep a history of actions to support undo, replay, and progressive refinement. Users may have to backtrack and so the ability to undo and redo refinements is useful.
7. **Extract:** Allow extraction of sub-collections and of the query parameters. Once the user has got what they were looking for, they should be able to save the information to a file (Shneiderman, 1996).

5.1.4 EVALUATING VISUALISATIONS

Information visualisations are best applied to exploratory tasks, especially to those which contain large datasets, since they can rapidly narrow down the search space. Therefore users of visualisations may not have any specific questions to ask of the data and instead want to get an overall picture. Gaining insight into the general appearance of the data means more meaningful questions can be asked of it later (North, Stasko, Fekete, & van Wijk, 2008).

This may make it difficult to evaluate the effectiveness of visualisations, since the tasks of browsing, exploration and gaining insight do not generate quantifiable results (Plaisant C. , 2004). Quantifiable results are achieved by making users answer specific questions – which will generate a length of time it took to answer the query, number of mistakes made and answer accuracy. These can be easily evaluated against benchmarks or other tools.

The evaluation of visualisations is more qualitative – for example – can the user explain the overall structure of the data better than if presented in textual format or can ‘similar’ objects be identified? It has been suggested that in-depth long-term case studies prove more useful to gain an idea of when the visualisation is most useful and if it can be classified as a successful tool in the professional world (Phan, 2007).

5.2 COMPUTER FORENSIC VISUALISATIONS

5.2.1 FILE VISUALISATIONS

A visual representation of file information was developed by Teerlink (Teerlink, 2004). The user is able to look at a specific directory and see statistics such as file size, file type, and access and modification dates. Teerlink uses two different visualisations – non-hierarchical and hierarchical. In the non-hierarchical visualisation, each file is represented as a square block without regard to where it is in the system. Each square block has a colour and intensity representing a file attribute, such as file type (Teerlink & Erbacher, 2006b). Figure 22 is an example of this with file size being shown in shades of grey. Information about a particular file can be seen in more detail by clicking on the coloured block the file represents.

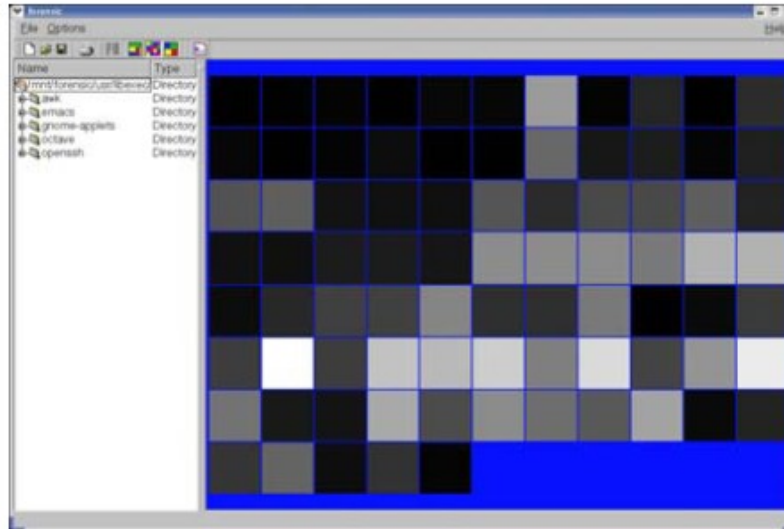


Figure 22 - Files in a directory filtered by file size. Light coloured blocks are large files, and darker coloured are smaller files

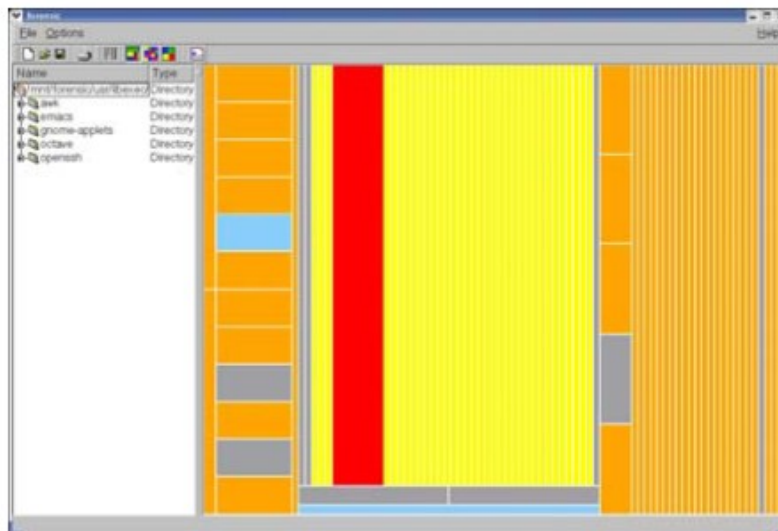


Figure 23 - Tree map view showing modified file recency

The second visualisation is a Tree-map, which respects the directory and file hierarchy of the file system. An example can be seen in Figure 23. This visualisation increases awareness of where the file is in relation to others (Teerlink & Erbacher, 2006a). Each file is a coloured block, the size of the block determined by the percentage of the directory the file occupies.

Subdirectories are represented by subdividing the block until each individual file has its own inner block. Teerlink and Erbacher argue their visual Tree-map greatly enhances the ability to see what files are in what folders, and reduces the time to locate large files in directories that are several layers deep (Teerlink & Erbacher, 2006b).

Their arguments are backed up by their user study results. Six users with knowledge of forensics were asked to investigate a hard drive to find the three altered and hidden files using Linux command line techniques and by using their visualisation tool. Three used the visualisation tool first, and the other three used the command line tools first. Their results indicated that 53% more files were found using the visualisation with a 35% reduction in time taken to find files. Locating the first hidden file was 57% faster using the visualisation (Teerlink & Erbacher, 2006b). There did not seem to be a vast difference between those that used the visualisation first or the command line tools first. Whilst the results seem impressive there were only six users, which is a lot less than the recommended 20 for a usability study (Nielsen, 2006). No qualitative results were obtained which may have given more fruitful evidence that the visualisation was preferred over the command line tools. Finally, it is unrealistic that investigators will exclusively use only command line tools, and comparing a visualisation against its antithesis will of course produce good results. A study against GUI-based file system tools would provide more credible results.

A second file visualisation was proposed by Stamps et al (Stamps, Franck, Carver, Jankun-Kelly, Wilson, & Swan, 2009). Instead of concentrating on the location and attributes of the file, this visualisation focuses on the words found in the file contents. By visually showing the importance of particular words and their relation to other information, investigators may be able to find useful information with much less effort (Stamps, Franck, Carver, Jankun-Kelly, Wilson, & Swan, 2009).

The authors also employed a different methodology, conducting a pre-design task analysis to discover what current investigators use and how they find textual data. This involved asking investigators to “do what they would normally” given an example hard drive to investigate,

whilst talking through what they were doing. The authors discovered that searching words related to those already found took significant searching, and this was an important part of the investigation process.

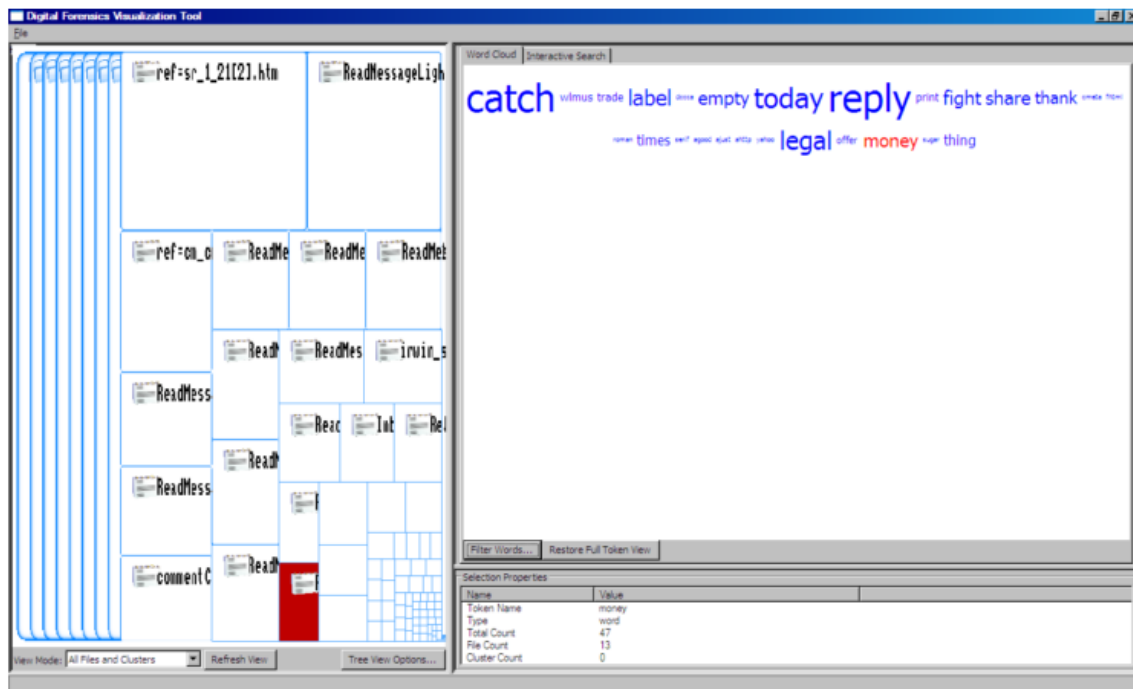


Figure 24 - Screenshot of textual visualisation system. Shows Tree-map of the system (left), a tag cloud of the terms in the selected files (top right), and metadata of the selected files (bottom right) (Stamps, Franck, Carver, Jankun-Kelly, Wilson, & Swan, 2009).

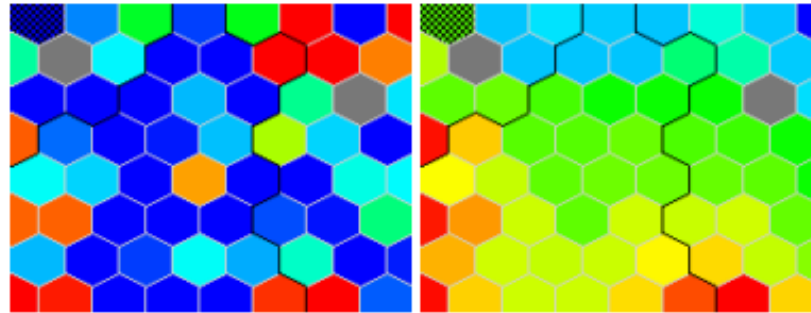
The visualisation is divided into three parts, which can be seen in Figure 24. A search-sensitive file hierarchy is on the left in the form of a Tree-map. The size of the block representing a file is based on the number of words, so images and non-textual files are not included in the Tree-map. On the top right is a tag cloud with the words in the selected files. The larger the word the more often it appears in the files. Particular words searched for will appear in red. Filters can be enabled to get rid of textual “noise” such as HTML tags. On the bottom right is metadata from the selected files (Stamps, Franck, Carver, Jankun-Kelly, Wilson, & Swan, 2009).

The authors did not complete a user study afterwards, but present a case study and show how easy it is to find related terms and relevant information. It is not compared to any existing tools

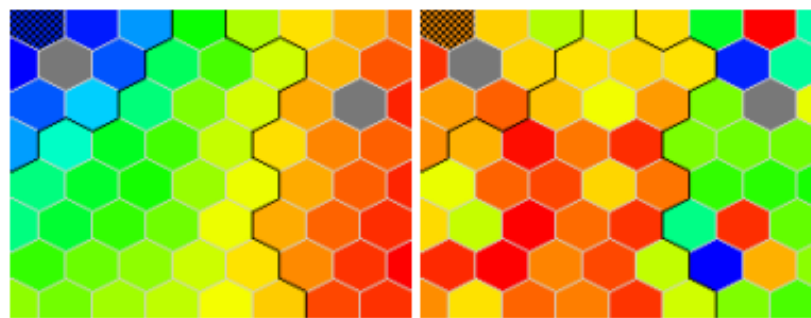
such as Autopsy (which was used in the pre-design task analysis), so it is difficult to judge how useful the visualisations are without user testing.

Fei uses a self-organising map (SOM) to visualise attributes or contents of files (Fei, 2007). A SOM is a neural network that maps high dimensional data onto a low-dimensional space whilst preserving the topology of the data. This essentially means that attributes of a file can be displayed as a colour, with red and blue being the two extremes (grey means no data is present). 'Component maps' represent a certain attribute and an example can be seen in Figure 25. In (b), blue represents older creation dates and red represents newer creation dates. By clicking on the hexagon more information about that file is revealed. Fei suggests patterns and correlations can be found by comparing the different maps of the same files (Fei, 2007). Anomalies may be very easy to spot, such as only one hexagon being a different colour to the rest.

Fei does not do any user testing, and like Stamps et al presents a couple of case studies that show the SOM working effectively to reveal anomalies and patterns. Unlike the previous two visualisations, Fei presents his visualisations without any context, so at first glance it is difficult to work out what they mean. The component maps are similar to Teerlink's square block approach, however Teerlink has inadvertently catered for the colour blind by just using one colour and using different shades. Teerlink's approach gives more context to the files with the added Tree-map view, whilst it is unclear with Fei's SOM model if you can investigate files in directories and sub-directories or see where these files are in relation to each other. It is difficult to tell whether the SOM would be a useful forensic tool without user testing, but it looks helpful for finding anomalies quickly.



(a) Component map (File type); (b) Component map (File creation date).



Component map (File creation time); (d) Component map (File creation day).

Figure 25 - SOM visualisation tool. Component maps generated from files (Fei, 2007).

5.2.2 NETWORK AND WEB LOG VISUALISATIONS

Krasser et al have produced a visualisation for network activity (Krasser, Conti, Grizzard, Gribshaw, & Owen, 2005) . Their aim was to make it possible for the investigator to see what was happening on the network at a glance, but to also provide more details when required. Their unique visualisation can be seen in Figure 26, with an example in Figure 27.

The left hand side depicts the source IP address, and the right hand side the destination port. The size of the packet is shown as the starting and ending vertical bar, the larger the packet the longer the line. The further away the vertical line is from the centre the older the packet. The colour of the lines depicts the protocol used (green is TCP, blue is UDP) and the brightness of the colour also shows age (faded lines are older). The user can zoom in and out and can click on

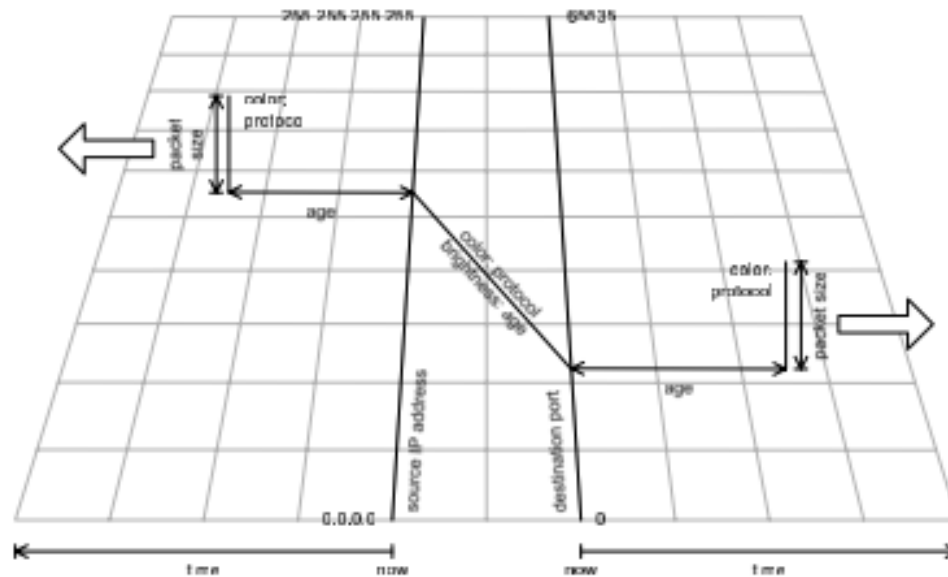


Figure 26 - Network Visualisation Overview (Krasser, Conti, Grizzard, Gribschaw, & Owen, 2005)

The authors do not provide any user studies but do give multiple screenshots of networks in various conditions such as under botnet attack (Figure 28) and Slammer worm attack. The screenshots are very clear and show immediately what is happening. They conclude that “*in this straightforward visualization, distinctive patterns can easily be detected by the human observer*” (Krasser, Conti, Grizzard, Gribschaw, & Owen, 2005).

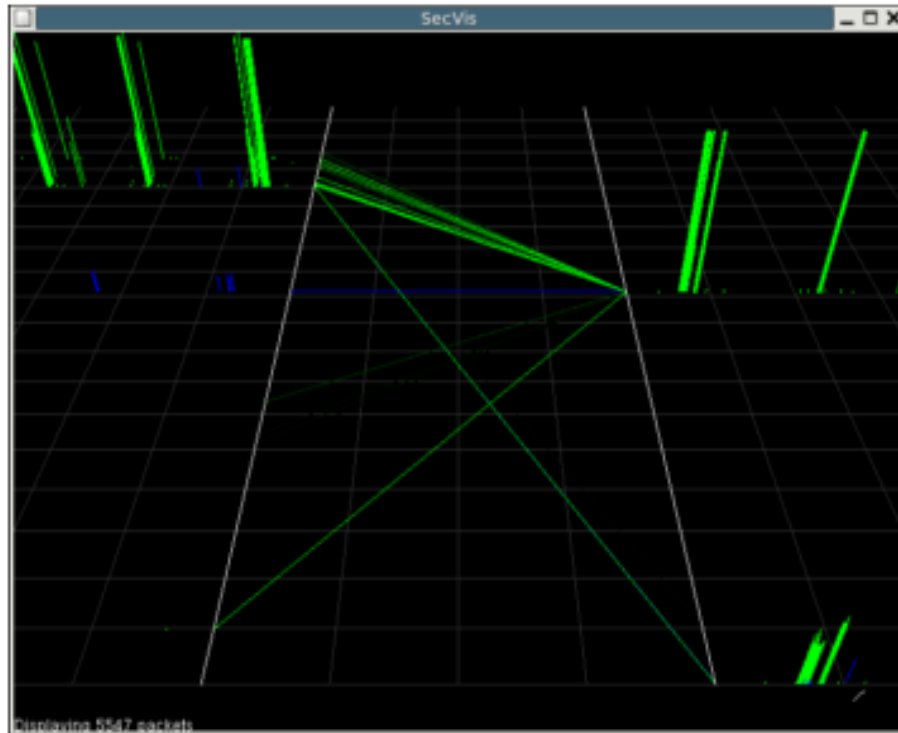


Figure 27 - Network Visualisation Example (Krasser, Conti, Grizzard, Gribschaw, & Owen, 2005)

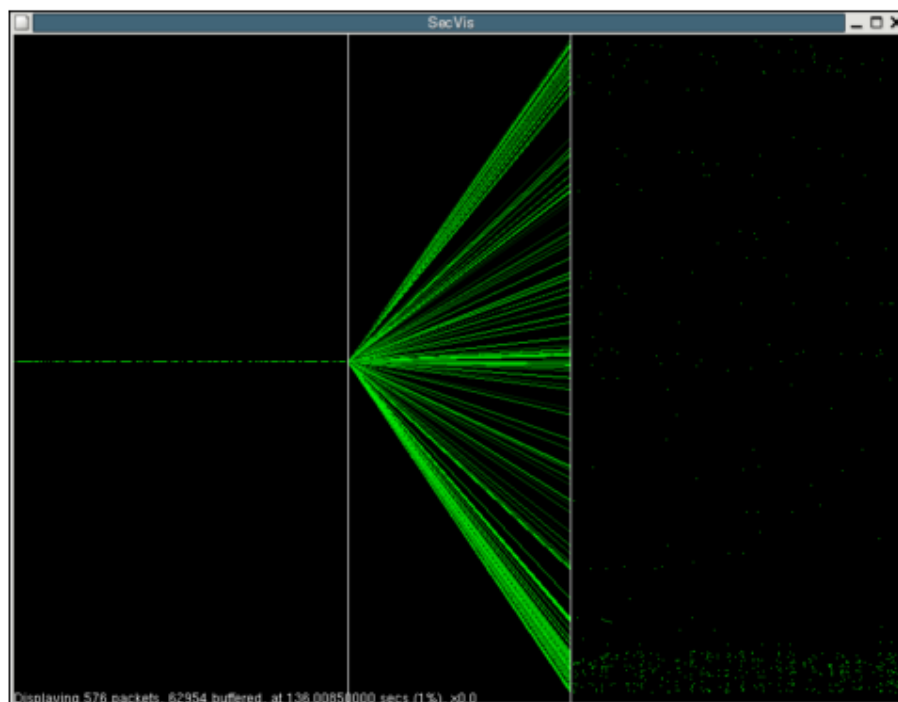


Figure 28 - Outbound Botnet attack traffic (Krasser, Conti, Grizzard, Gribschaw, & Owen, 2005)

Tudumi is a visualisation tool for monitoring and auditing user behaviours on a server, by visualising server logs (Takada & Koike, 2002a). It focuses on three activities: access to the server from other computers, logins to the server and identity changes from a user to another user (i.e. ^{su} on Linux), which the authors call *substitution*. Tudumi works by applying rules to server log files to extract behaviours, and changes the visual representation depending on the applied rules.

An example Tudumi screenshot can be seen in Figure 29. The visualisation consists of layers of two circles. The bottom layer represents user substitution, and the other layers represent network access and login information. Each layer is a rule the investigator has applied to the logs, e.g. all hosts from a certain country. Hosts that a user connects from are represented by spheres on the outer circle, and logins are represented by cubes in the inner circle, which are joined by a line to the host they connected from. Coarse dashed lines mean the connection was via a terminal, and fine dashed lines mean files were transferred. Solid lines mean both happened. Some logins are represented by red cones, which means they have tried to login as a different user. User substitutions are represented on the bottom layer as two users (outer circle spheres) connected by an arrow. The arrow head is coloured to indicate the success rate of the substitution – red for success and blue for failure. If user substitution has occurred more than once, the colour is a proportional mixture of successes/failures (Takada & Koike, 2002a).

The cubes can be assigned a texture, so that particular users or groups of users can be easily identified. If a sphere/cube is clicked on, Tudumi will only show visualisations of that particular host/user (Takada & Koike, 2002a).

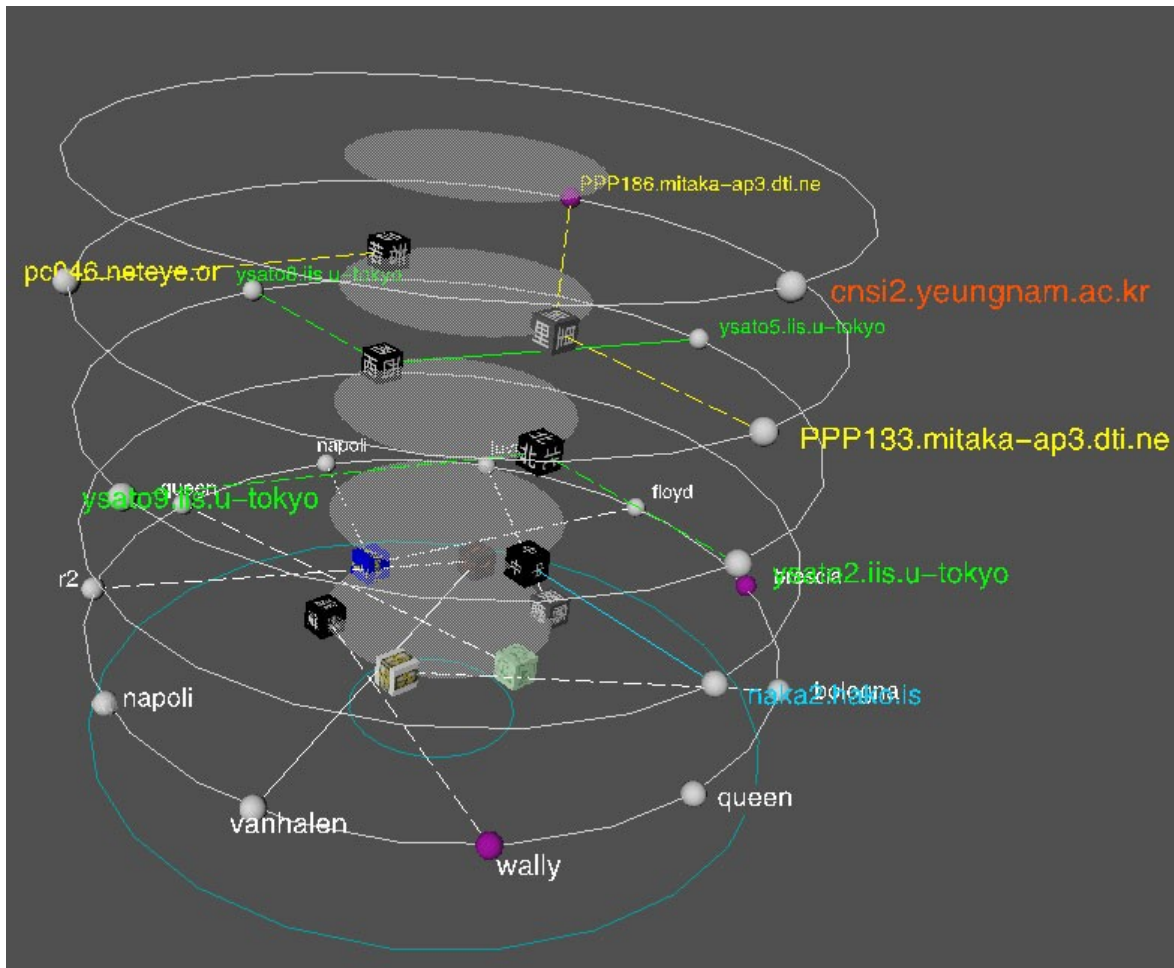


Figure 29 - Tudumi visualisation example (Takada & Koike, 2002a)

Even though layers and certain hosts/logins can be filtered, Tudumi is by default a cluttered visualisation, as can be seen in Figure 30, making it arguably just as difficult to understand as textual log files. The authors do not carry out any user studies, nor provide any evidence for why layered circles are the best visualisation for server logs. It is unclear whether more information can be gathered from the visualisation, such as what time period it covers or at what time user substitutions happened. Despite the clutter, with suitable filtering and texturing of cubes and a trained eye, Tudumi may be valuable in spotting anomalies and unusual behaviour that would not stand out just by examining log files.

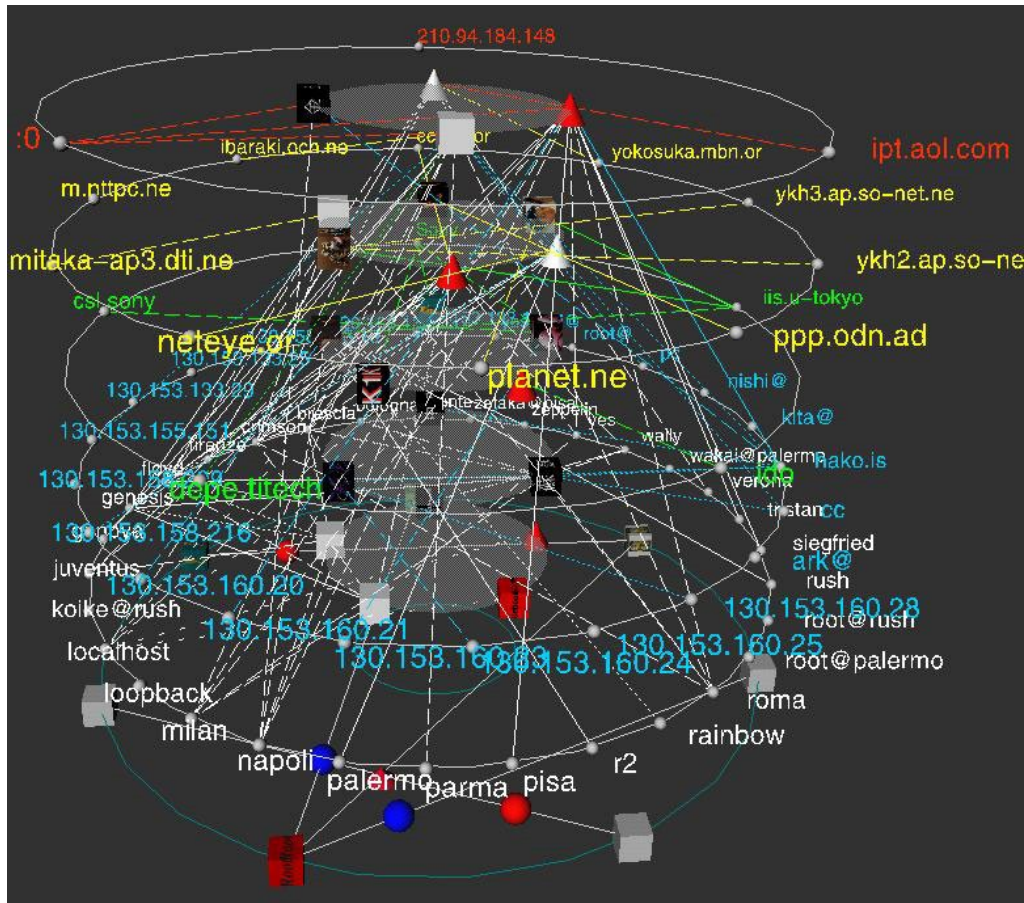


Figure 30 - Tudumi screenshot (Takada & Koike, 2002a)

5.2.3 TIMELINE VISUALISATIONS

Whilst there are currently no web browser log file visualisations available, there are a couple of timeline-based visualisations which can take in web log files. CyberForensic TimeLab (CFTL) is a timeline-based forensic tool which finds and plots all forensic data based on timestamps (Olsson & Boldt, 2009). Since time is one of the common attributes between all files on a computer system, Olsson and Boldt believe timelines are an effective way of organising and browsing evidence. CFTL allows the examiner to see a full timeline of events and then choose a time span to concentrate on. Different file types are given a new line on the timeline, with histograms representing the number of events at a specific time. An example can be seen in Figure 31. Below the timeline is a detail viewer panel, where the timestamps are listed in

chronological order accompanied by further details (Olsson & Boldt, 2009). The investigator is able to zoom in to see more details.

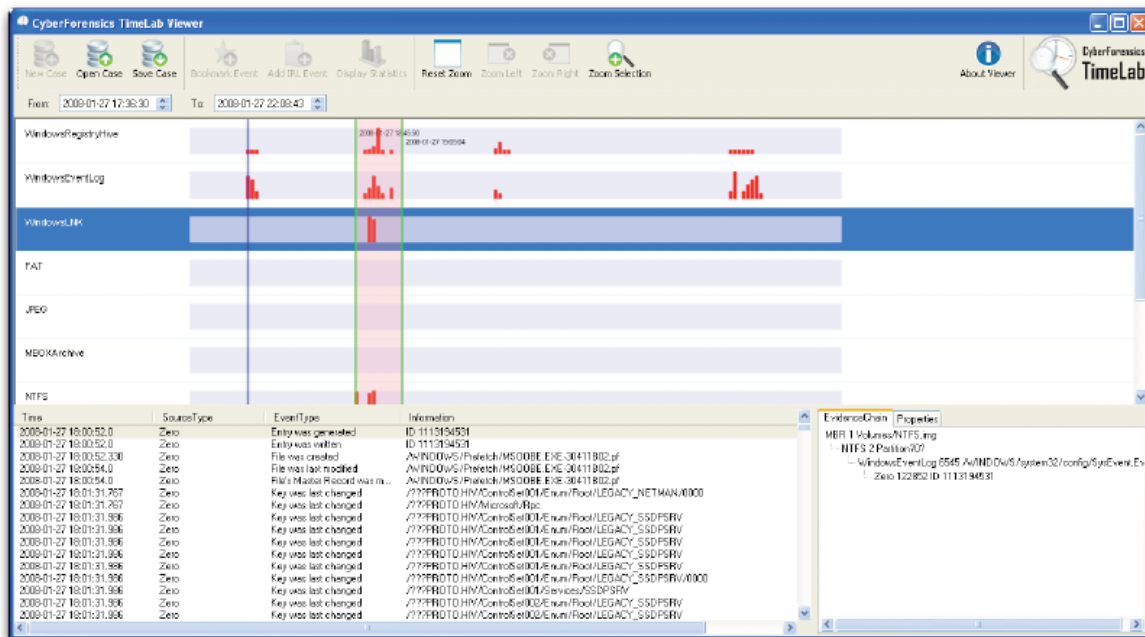
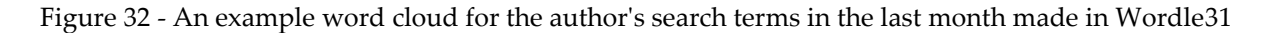


Figure 31 – CyberForensics TimeLab screenshot (Olsson & Boldt, 2009)

Olsson and Boldt conducted a user study, where 6 users used CFTL and 6 other users used Forensics Tool Kit (FTK) to investigate a fictional crime. The users were asked to answer 6 questions about the fictional crime as quickly as possible. Overall, those using CFTL got a few more questions right than with FTK but more strikingly the time to complete the task took on average 14mins with CFTL and 45mins with FTK. The authors also conducted a usability study with the same users, and found in general CFTL was easier and more intuitive to use. The authors realise that 12 people (none of whom were forensic investigators) is not enough to conclusively say whether a timeline approach is a better technique, but the results do seem to indicate it is a faster way of finding evidence (Olsson & Boldt, 2009). However, not using forensic investigators who may know how to utilise FTK better, and the use of time-centric crime scenarios may have skewed the results.

The basic log data produced by all web browsers are the accessed URL or resource and the time of this access. The data is temporal and a timeline may be the visualisation that fits best. Other information that can be extracted from all browsers are the search terms used in search engines and all the domain names visited (both can be extracted from URLs). Search terms, sub-domains and domains could be grouped together and displayed as a word-cloud, with the most frequently visited domain names / search terms appearing the largest (example in Figure 32).



31 <http://www.wordle.net/create>

The three investigators interviewed all agreed that a timeline would be the most suitable main visualisation. Their extra requirements were:

- Must be able to handle a lot of data (tens of thousands of entries).
- Be able to see an overall timeline and zoom into areas easily.
- Data must be filterable. The investigators from the financial company were very keen to have filtering that excluded all advert content. The investigator from the SCDEA wanted to filter URLs that were searches, images and files accessed on hard drives rather than URLs.
- Data must be 'taggable'. Users must be able to define categories such as "suspicious" and be able to put data in that category by filtering.
- Filtering and searching must allow for fuzzy searches or regular expressions to include spelling mistakes.

5.3.1 AVAILABLE VISUALISATION APIS

There are many JavaScript APIs available for drawing graphs, charts and timelines, each with benefits and disadvantages. Since there does not seem to be a JavaScript toolkit that covers all the visualisations, several different specialised ones will be used as well as using the JavaScript library jQuery to create some new ones.

For simple histograms and pie charts, Google Charts³² API will be used. Google Charts is very simple and provides well-presented, good-looking charts. The downside is that the API does not provide much room for extensions and customisation, so other libraries will be needed for charts that are more complex, such as jQuery Horizontal Bar Graph³³.

The timeline, will be created using Flot³⁴, a jQuery graph plotting library which can plot temporal data. Flot can handle very large amounts of data, and is completely customisable. The

³² <http://code.google.com/apis/chart/>

³³ <http://www.dumpsterdoggy.com/plugins/horiz-bar-graph>

³⁴ <http://code.google.com/p/flot/>

SIMILE Widget Timeline³⁵ was also looked into, but a few experiments with large datasets found it to be unwieldy and difficult to separate out the data points. An example can be seen in Figure 33. Timeline can only show about 20 points on a graph in close proximity before the graph begins to grow very large and difficult to follow.

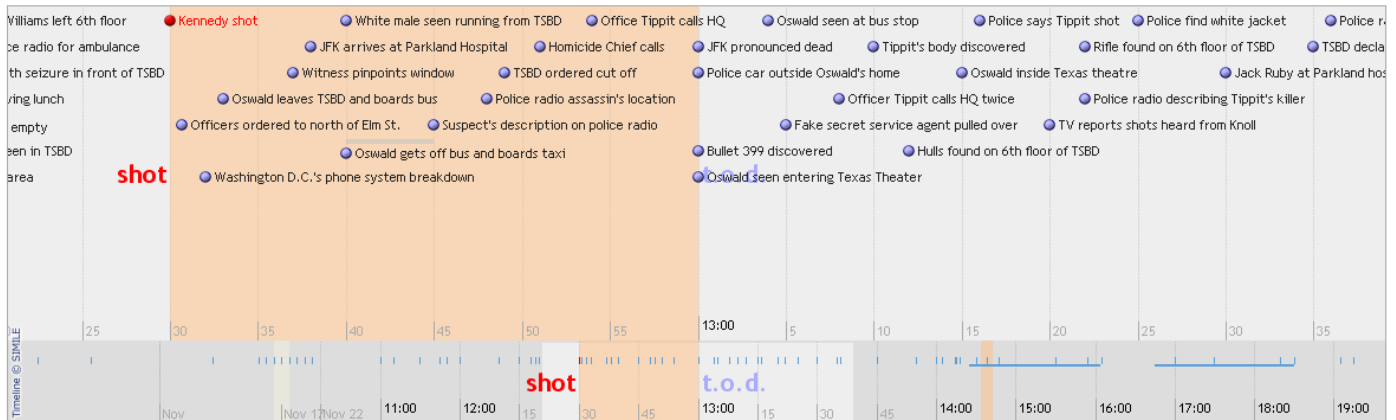


Figure 33 - Screenshot of Timeline showing the events leading up to and after the assassination of JFK

³⁵ <http://www.simile-widgets.org/timeline/>

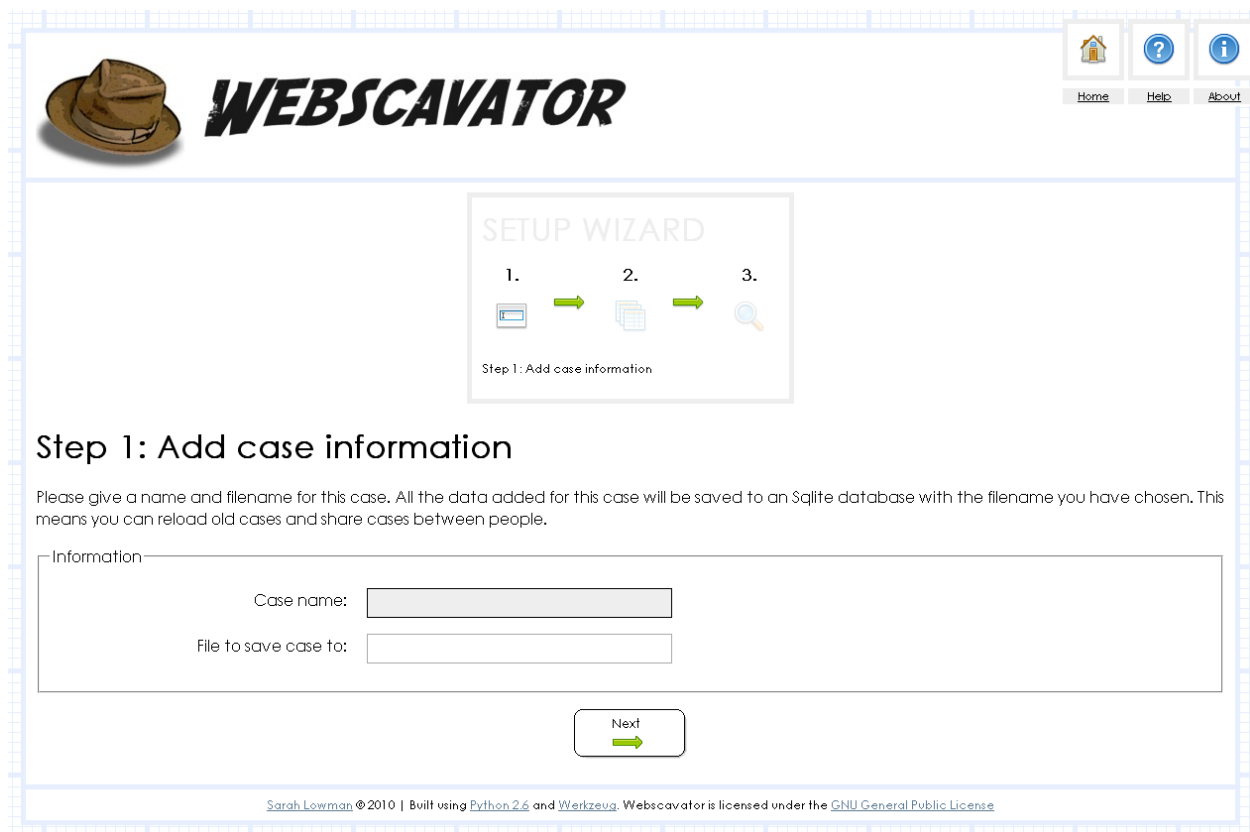
CHAPTER 6: WEBSCAVATOR

This chapter describes Webscavator, a tool for visualisation web history that was developed as part of this project to help answer the research questions proposed. For input, Webscavator accepts CSV and XML files from programs such as Net Analysis and Web Historian. This way, Webscavator does not need to handle the complexities of directly parsing web history files and, if necessary, forensic investigators can use these tools to pre-process the data to be visualised. The user and developer documentation can be found in the appendix.

Each unique visited website or file is stored in a database as an *entry*. The minimum data required for each entry is the resource location and the access date and time, as this is all Webscavator needs to be able to visualise the data. This database also stores information about the case and saved filters. The data for each case is stored in a separate SQLite database file. This ensures different cases are kept completely separate and makes it easy for cases to be shared and moved.

To start a new case, the user follows a wizard during which they choose a case name and database file (see Figure 34). They can then upload as many CSV or XML files as they wish, giving each a different ‘group’ name. Webscavator then parses the uploaded files, automatically

extracting domain names and search terms from any URLs. Once complete, the date and an MD5 hash of the database is created and stored in a text file with the same name as the database. Any further updates to the database will cause a new MD5 hash to be appended to this file alongside a message describing the update. This simple technique allows investigators to check the integrity of the database by comparing the latest recorded MD5 hash to the actual hash of the database.



The screenshot shows the Webscavator application interface. At the top left is a logo featuring a brown fedora hat next to the word "WEBSCAVATOR" in a bold, black, sans-serif font. To the right of the logo are three navigation buttons: "Home" (with a house icon), "Help" (with a question mark icon), and "About" (with an information icon). Below the navigation bar is a "SETUP WIZARD" section. It contains a progress indicator with three steps: 1. (represented by a document icon), 2. (represented by a document icon with a green arrow pointing to it), and 3. (represented by a magnifying glass icon). Below the progress indicator, it says "Step 1: Add case information". The main content area is titled "Step 1: Add case information" and contains a paragraph of instructions: "Please give a name and filename for this case. All the data added for this case will be saved to an Sqlite database with the filename you have chosen. This means you can reload old cases and share cases between people." Below the instructions is a form with two input fields: "Case name:" and "File to save case to:". At the bottom of the form is a "Next" button with a green arrow pointing right. At the very bottom of the interface is a footer line that reads: "Sarah Lowman © 2010 | Built using Python 2.6 and Werkzeug. Webscavator is licensed under the GNU General Public License".

Figure 34 - Screenshot of the wizard to add a new case

Once the case has been loaded, the visualisations are presented in different tabs (see Figure 35). Above the tabs is a filter box with a set of prebuilt filters – this will be described later.

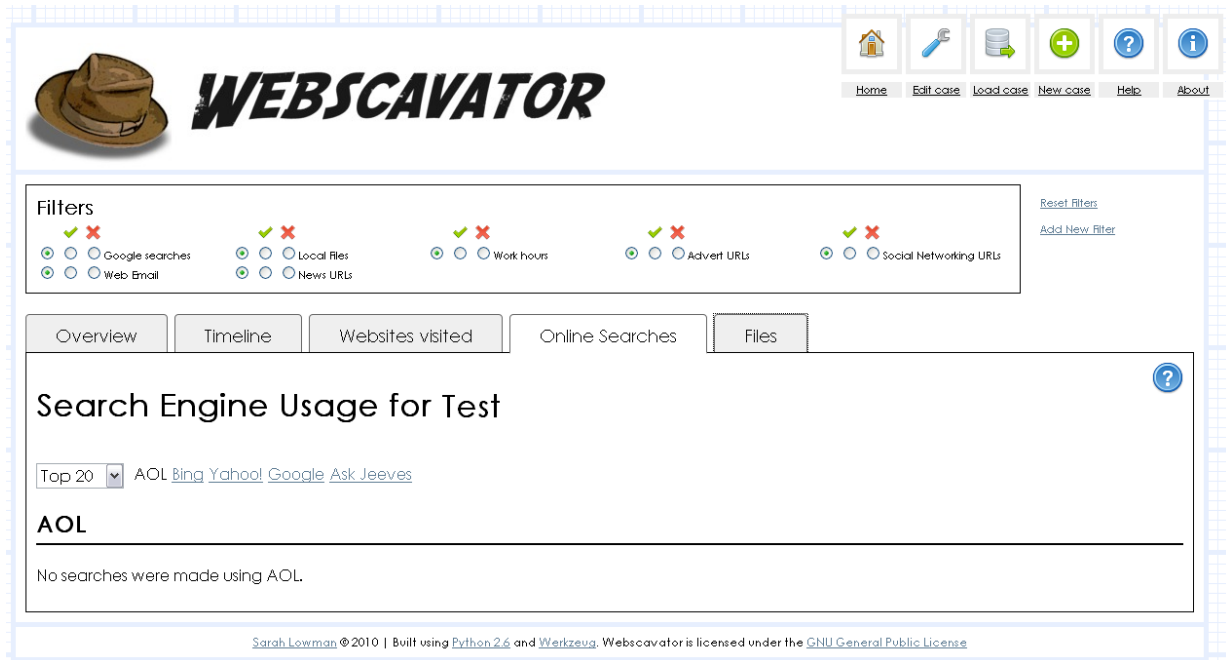


Figure 35 - Screenshot of the home page of Webscavator once a case has been loaded

6.1 FILTERS

Filters allow the user to highlight or remove entries from the visualisations by matching on various attributes. They are useful to narrow down results, highlight particular features of interest or reduce clutter due to irrelevant entries. There are several filters provided by default, including *Google searches*, *local files*, *work hours*, *advert URLs* and *social networking URLs*. The filter bar is always present at the top of the screen, and a filter is activated by clicking on a *highlight* or *remove* radio button. Highlighted filters have their filter name highlighted in green, and removal filters in red to show they are activated, as it is often easy to forget that filters have been turned on (see Figure 36).



Figure 36 - Screenshot of the filter bar with 'Social Networking URLs' highlighted and 'News URLs' removed

New filters can be created by clicking on 'Add filter', which brings up a pop-up form (see Figure 37). The user can choose what attributes to filter on by selecting the appropriate values in the provided select boxes. Some attributes such as URL protocol, URL username and URL password ask the user to pick from a predefined list of existing values rather than having them type something by hand (see Figure 37).

The user can choose to show entries which match exactly, don't match at all, contain the given value as a substring, are greater than or less than the given value (for date/time attributes), match a particular regular expression or are part of a given list. For list values, the user can create text files defining the list, with each line treated as an entry. These files are put in a 'file lists' folder, after which they can be chosen in the filter. The premade filters *Social Networking URLs* and *Advert URLs* are based on matching the domain name to those in a list of social networking websites and advert websites respectively.

It is easy to create a filter that matches based on multiple criteria by clicking the "AND" link, which will allow multiple matching rules to be specified. The entire filter matches if all of the given rules match.

The screenshot displays a web form for adding filters, divided into two main sections: "Filter Details" and "Create Filters".

Filter Details: This section contains a label "Filter quick name:" followed by a text input field containing the word "Test".

Create Filters: This section contains a series of dropdown menus for selecting filter criteria. The first three dropdowns are labeled "URL Part", "Protocol", and "Is". The fourth dropdown is open, showing a list of options: "Pick...", "Pick...", "https", and "http". To the right of the dropdowns is a blue link labeled "AND". Below the dropdowns is a "Submit" button.

Figure 37 - Screenshot of the add filter form

6.2 VISUALISATIONS

6.2.1 HEAT-MAP

On the first tab, labelled ‘Overview’, there is a heat-map (see Figure 38) showing the number of accesses that occur for each hour of each day of the week. A heat-map is a visualisation where different values are represented as different colours. It is the most widely used visualisation in biology (Wilkinson & Friendly, 2009). In this case, the higher the value, the lighter the colour - small values are dark blue and high values are light blue. Another commonly used variation uses blue to indicate small values and red for high values similar to a temperature scale.

Heat-maps have the advantage that they “*compact large amounts of information into a small space to bring out coherent patterns in the data*” (Weinstein, 2008). Heat-maps also conform to the Preattentive Processing Theory described in Chapter 5. Differences in colour are processed quickly so inferences of the data are made much faster than without colour.

Webscavator’s heat-map can highlight patterns in web browser usage times. Figure 38 shows the heat-map for some example data. From the colours, it is easy to see that this person used the internet between 9am and 6pm on weekdays and usage dropped significantly from 12pm to 1pm, possibly due to a lunch break. Any unusual access times, such as Sunday at 6am, would instantly stand out due to a colour change.

The heat-map is created using JavaScript to colour each cell in the HTML table according to a scale of blue intensity depending on the value. Any cells with the value 0 are coloured black.

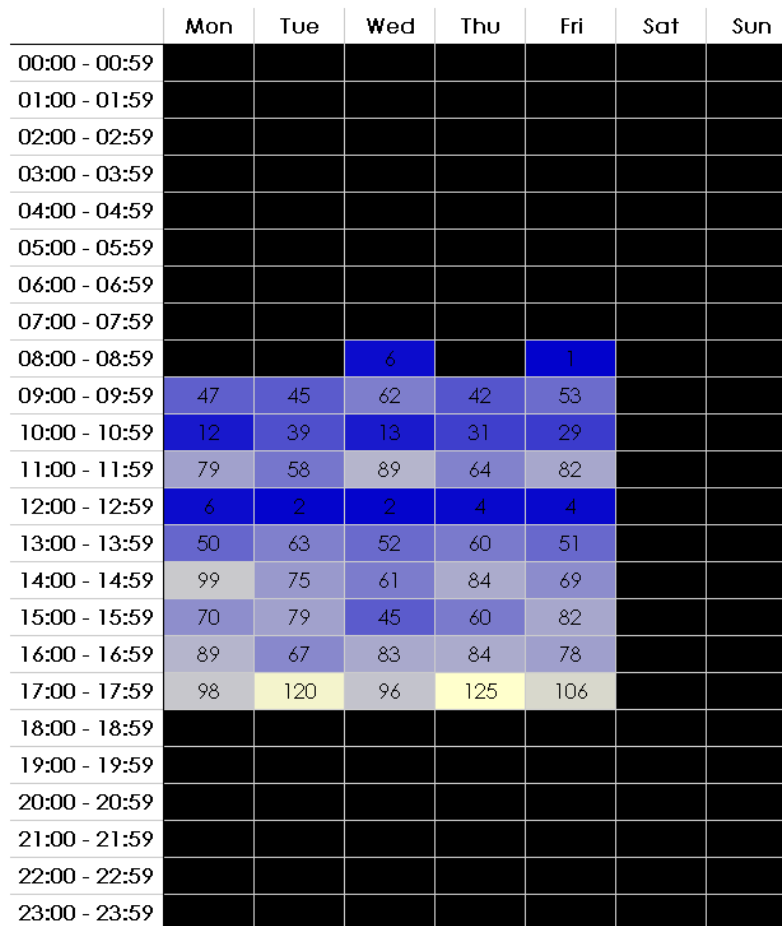


Figure 38 - Screenshot of a heat-map

6.2.2 TIMELINE

The second tab, 'Timeline', displays the main visualisation: a timeline. The timeline has the day on the x-axis and the time along the y-axis. Each point on the graph is an entry in the database (see Figure 39). Weekends have a background colour of light grey. The second, smaller timeline provides an overview of the current data, and is used to keep a sense of perspective when the main timeline is zoomed or filtered (as it always displays the complete data).

Hovering over any point will display a pop-up with the time, date and the resource accessed (see Figure 40). Clicking on any point will show more detailed information underneath the plots, such as page title and link type (if that information is available). Checking 'Show Labels'

shows the resource accessed beside the point on the graph, so is only suitable when zoomed in enough to not crowd the timeline. To move the timeline backwards or forwards, the user can click on 'Move back 31 days'/'Move forward 31 days'. The user can also move by clicking and dragging the mouse to pan.

[Reset Graph](#) | [Zoom Out](#) | Show Labels ☐

◀ [Move back 31 days](#)

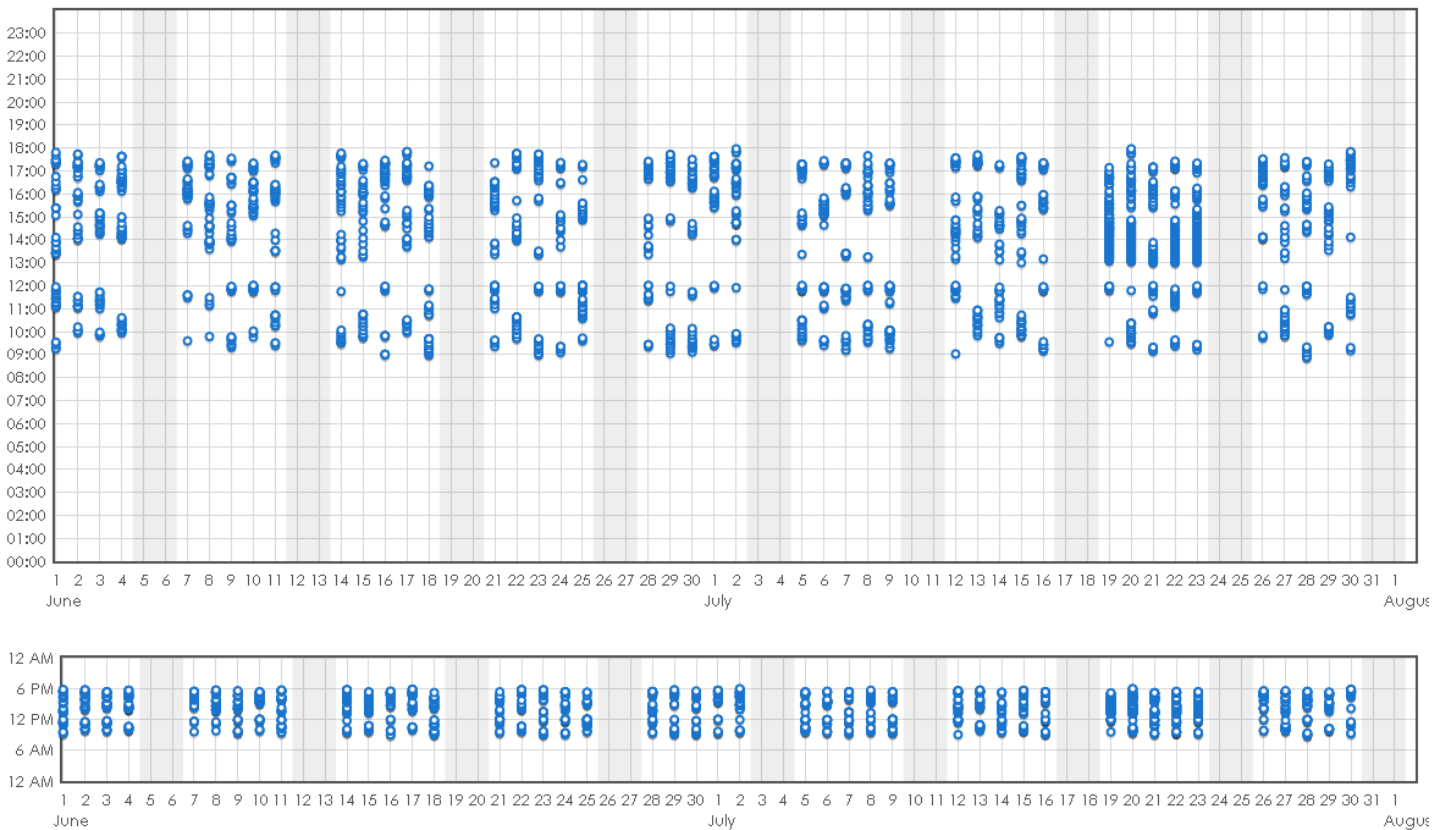


Figure 39 - Screenshot of the timeline

Double-clicking on a point in the larger graph causes it to zoom centred on that point. The units of the y-axis will change as the graph is zoomed to show hours, minute and seconds as appropriate depending on the zoom level. The smallest granularity the y-axis will show is one second, as this is the smallest unit of time supported all of the input formats. Clicking on the 'Zoom Out' link will zoom the graph back out. Clicking on 'Reset Graph' will reset the graph to its original zoom level. To zoom in more quickly, the mouse can be clicked and dragged to

select a rectangular area in the overview graph. This will zoom in on that area in the main graph. When the main graph is zoomed, the overview graph shows a rectangle indicating the area of the dataset currently visible in the main graph, as shown in Figure 41. This is intended to help the user keep the bigger picture in mind when examining details. When zoomed, the user can still pan in any direction to browse the graph.

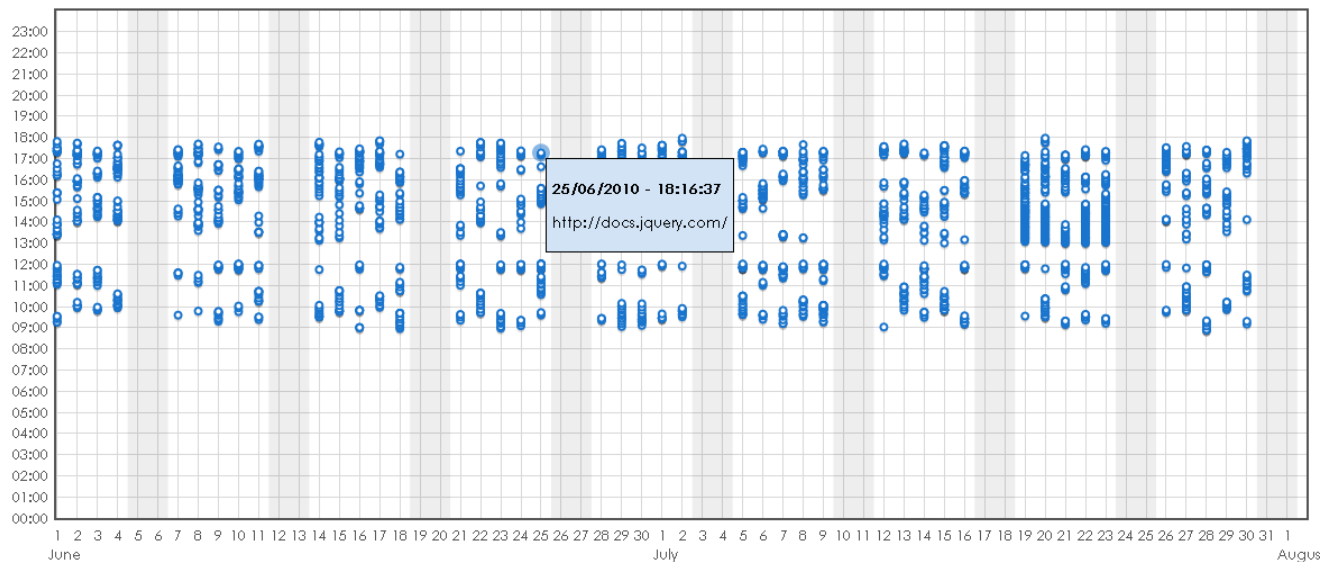


Figure 40 - Screenshot of the timegraph when the mouse hovers over a point

In the timeline, setting a filter to remove data will make all the entries that match that filter turn a very light grey. Entries are not removed entirely so that a sense of context is maintained. Setting a filter to highlight data will turn all the entries that match the filter orange, like some of the entries in Figure 41. Removal filters take precedence over highlight filters, so if an entry matches both a removal and a highlight filter, then it will be marked removed.

The timeline was made using the JavaScript library Flot. Each time the user zooms, pans or changes the filters an AJAX request is sent to the Python code, which determines which entries satisfy the filter and date constraints. The entries are sent back and the graph is redrawn.

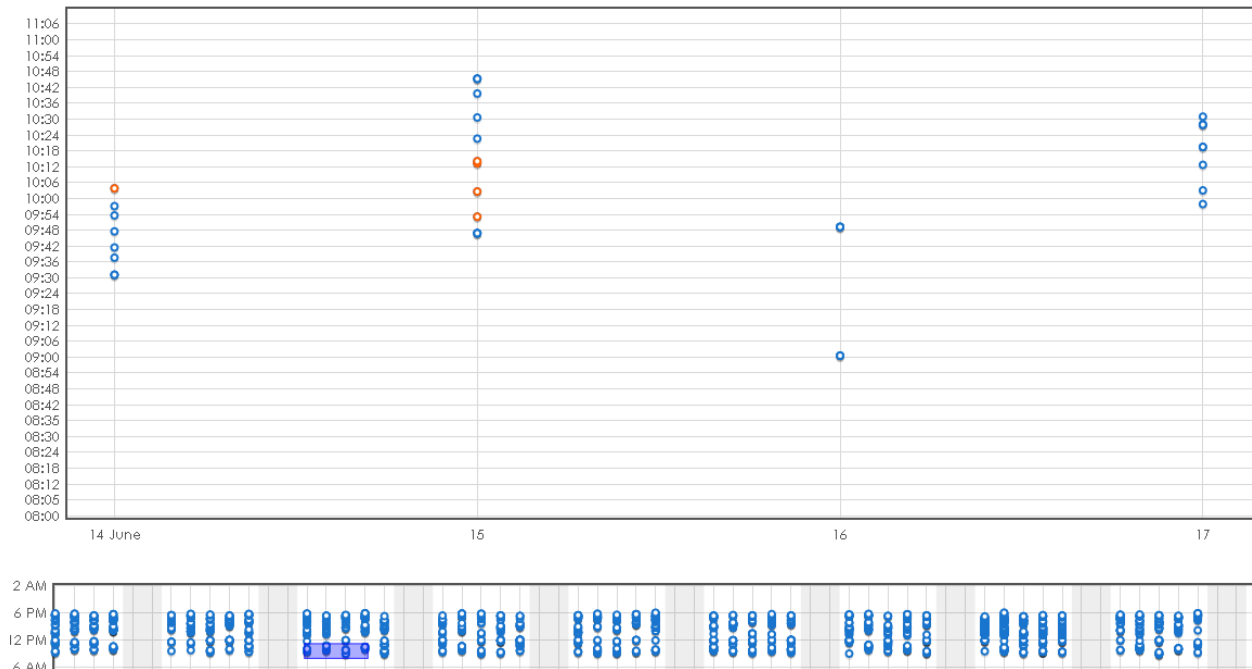


Figure 41 - Screenshot of the timegraph zoomed in with highlighting

The timeline conforms to six out seven tasks Shneiderman deems all visualisations should be able to do (Shneiderman, 1996) (discussed in Chapter 5).

1. Overview: the timeline shows 2 months worth of data to begin with. Although this is not all the data, the timeline would be too crowded and it would be difficult to show the x-axis labels if it was all displayed. It is easy to go back and forward to see different data.
2. Zoom: The timeline provides zooming and panning, with the overview plot giving context.
3. Filter: Webscavator has a filtering system to remove unnecessary data.
4. Details-on-demand: the user is in control when they want to see detailed information about any points.
5. Relate: The filters allow similar entries to be highlighted together.
6. History: Filters are saved and when tabs are switched the timelines stay unchanged so the user can go back to where they were. Although no “undo” button is provided, it is very easy to pan in the opposite direction, zoom back out and turn filters on or off.

7. Extract: The timeline does not conform to this task as information cannot be extracted. Although the user can take a screenshot, they cannot turn the timeline back into a spreadsheet of entries or summary PDF. If more time was available this feature would be implemented.

6.2.3 BAR CHART

The third tab, 'Websites Visited', shows a bar chart of the top 20 domain names visited (see Figure 42). This can be changed to show the top 50, top 100 or all domain names visited. The total number of visits for that domain is appended to the bar. Clicking on a bar shows a pop-up giving the sub-domain breakdown, like in Figure 43. The bar chart is also subject to the filters. Entries matching a removal filter will not be included in the chart, and if any highlight filters are turned on, then only entries that match these are included. This is useful if, for example, the user is only interested in a particular time period.

A bar chart was chosen because it quickly shows which domains are most visited and by how much. The length of the bars is quickly mentally processed, and like the heat-map fits the Preattentive Processing Theory. In Figure 42, it is clear that google.co.uk was visited nearly six times more than any other website. These visits can then be highlighted on the timeline to see when and why they were made. This visualisation is useful for determining the kind of websites that have been visited and can therefore help guide investigation and filter selection for the timeline.

The bar chart is made using a modified version of the JavaScript jQuery Horizontal Bar Graph library. Similar to the timeline, when the filters or the number of entries is changed, an AJAX request is sent to the Python web application, which then returns a list of domain names and counts to display. The bar chart is then redrawn with the new data.

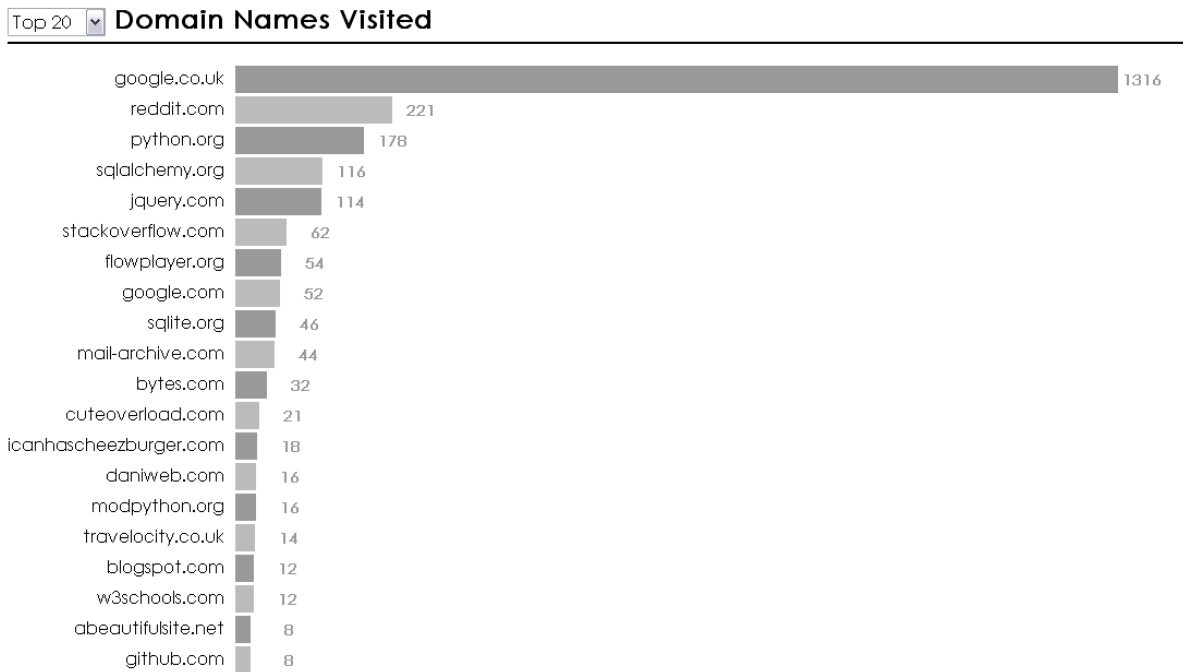


Figure 42 - Screenshot of the top 20 domain names

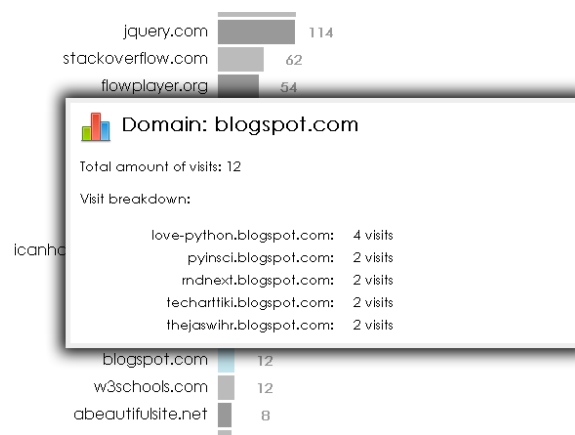


Figure 43 - Screenshot of the pop-up box when the 'blogspot.com' bar was clicked

6.2.4 WORD CLOUD

The forth tab, 'Online Searches', displays a word cloud of all the search terms used with particular search engines. A word cloud displays a set of words with the font size of each

varying depending on how important the word is. In this case, the more a particular search term occurs, the larger its font size.

Webscavator comes with a configuration file where search engines can be added and removed. The defaults are Google, Bing, Yahoo!, Ask Jeeves and AOL. When the case is first added, each URL is tested to see if it is a search and any search terms are collected and added to a separate table. Any search terms surrounded by quotes are kept as a phrase. Figure 44 shows a word cloud for the top 20 search terms used with Google. It is obvious that the person in question performs a lot of searches related to programming. The word cloud can be changed to show the top 50, top 100 or all search terms. Search terms are also subject to the filters, which act the same way as for the domain name bar chart. Each word in the word cloud can be clicked to show a pop-up with a list of all the full search queries in which it appeared, along with the date and time for each (see Figure 45).

A word cloud was chosen because it quickly highlights the most frequently used search terms, as large words attract more attention than smaller words (Lohmann, Ziegler, & Tetzlaff, 2009). People tend to scan rather than read word clouds, and so they are more useful for giving quick overall impressions rather than detailed information (Hassan-Montero & Herrero-Solana, 2006), which can be obtained either by clicking individual words or in conjunction with one of the other visualisations.

The word cloud was made using JavaScript to alter the font size of each word by multiplying its relative occurrence by a factor. The factor is calculated so that the smallest word will always be size 11pt, ensuring readability. Any words larger than 300pt are capped at this size so they do not appear so large they go off the screen. Like the bar chart and timeline, whenever any options change, an AJAX request is sent to the Python web application, which returns a list of words and their ratios, which are then displayed.

Google

The top 20 searched terms are below.

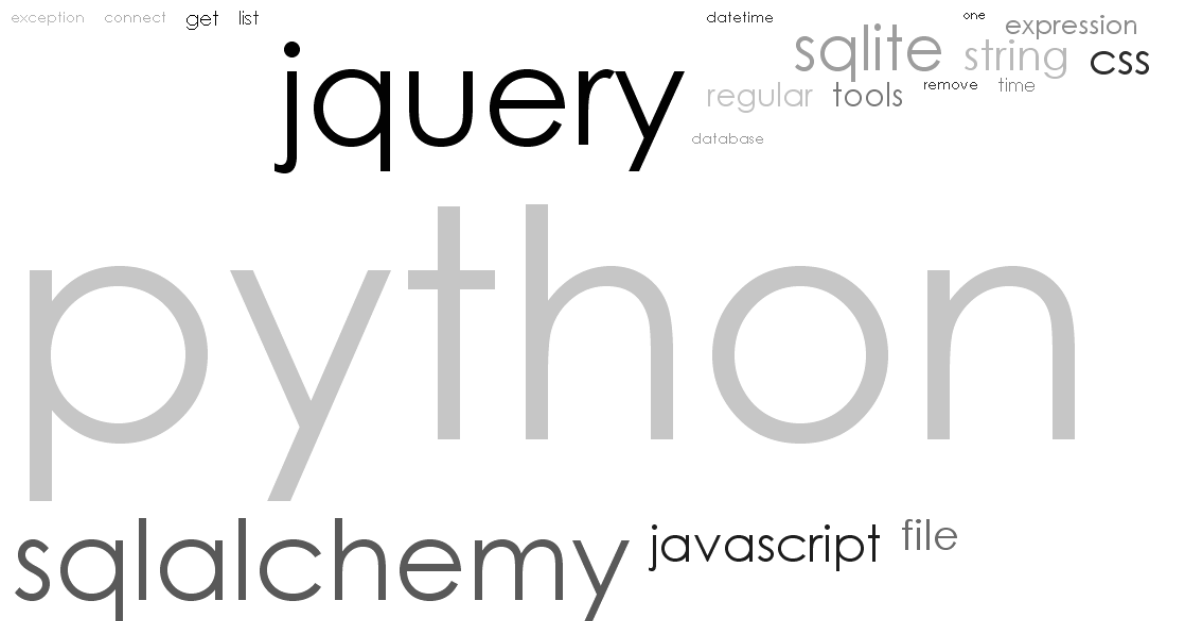


Figure 44 - Screenshot of the search terms word cloud

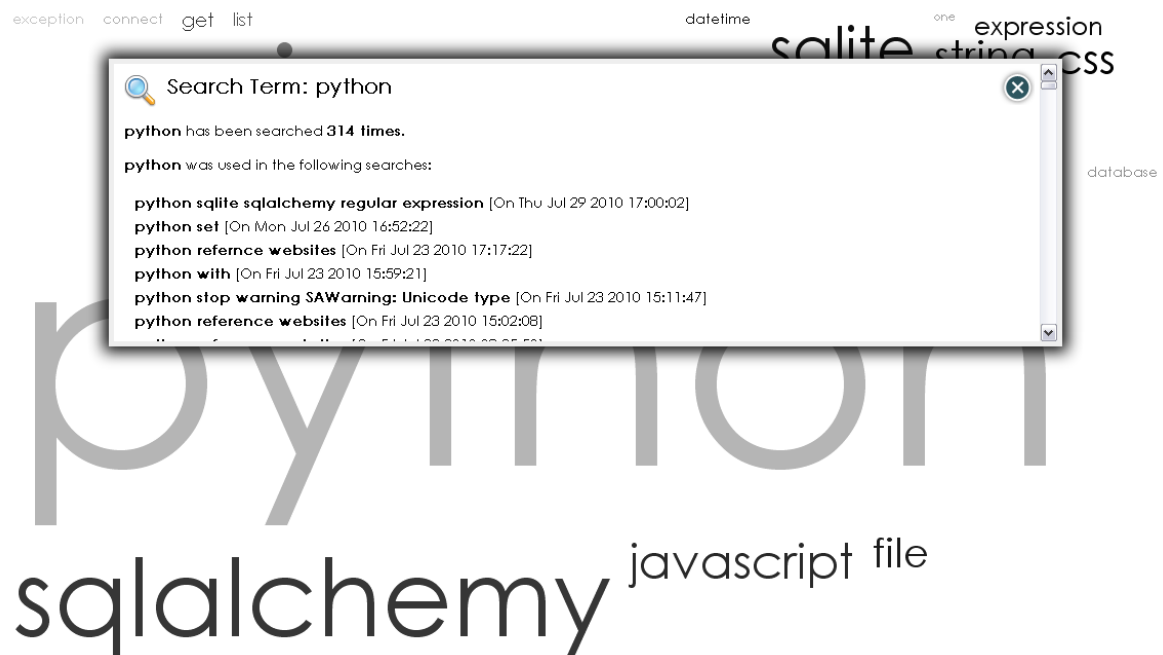





Figure 45 - Screenshot of pop-up when the term 'python' is clicked

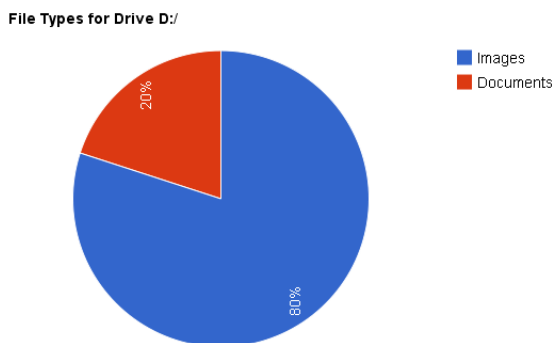
6.2.5 TREE STRUCTURE

The final tab, 'Files' shows all the files that have been accessed. These only appear in `index.dat` files, but are likely to be useful even if Internet Explorer is not used as a browser, because Windows records regular file accesses in `index.dat` files.

In total, 205 files were accessed 406 times on 3 drives.

 H:/ Drive [6 accesses]	show details
 C:/ Drive [385 accesses]	show details
 D:/ Drive [15 accesses]	hide details

Pie Chart of the different file types accessed



Images [12 accesses]

Documents [3 accesses]

- D:
 - images
 - stuff
 - names of security guards.docx [1 access]
 - how to make the fake id card.docx [1 access]
 - jewels in price order.docx [1 access]

Figure 46 - Screenshot of pie chart and document file structure for the D drive

The accessed files are partitioned first by drive letter (currently only Windows is supported) and then by file type (see Figure 46). Under each drive is a pie chart showing the breakdown of the different file types accessed. This is useful to determine what the drive is mainly used for, e.g. many documents and PDFs may suggest a work computer and more images and MP3s may

suggest a home computer. For each file type, a tree structure of where the files appeared on the drive is displayed. Each file can be clicked on to show a pop-up giving the dates and times of the accesses (see Figure 47).

Tree views are a common way of displaying files in file manager applications, so the presentation of the files here will be familiar. The indentation of folders gives context showing where files are in relation to each other in a way that simply listing the file paths does not. This also gives the cognitive benefits described in Chapter 5, as the brain does not have to do as much processing to understand the file path structure – the visualisation has done some of the processing already.

The files tab is not filterable, as there are usually significantly fewer file entries than websites visited and it was not deemed essential to filter these down further. The file entries are processed into a Python tree structure, which is recursed to draw the folders and files.

This visualisation could be particularly useful when trying to determine what sorts of files may have been accessed on no longer present USB key drives.



Figure 47 - Screenshot of the pop-up when the file 'DSC0281.JPG' is clicked

CHAPTER 7: ANALYSIS OF RESULTS

The three digital forensic investigators who took part in the initial research also took part in the user testing (described in Chapter 2). Each participant completed both scenarios described using Net Analysis and Webscavator, which was followed by an interview and usability questionnaire. For each scenario, the participants were timed when answering the questions. The question sheets used can be found in the appendix. Each answer was accompanied by a confidence rating ranging from 1 to 5. 1 indicates a complete guess and 5 indicates total certainty. The participants were also asked to fill in a usability questionnaire, which can also be found in the appendix.

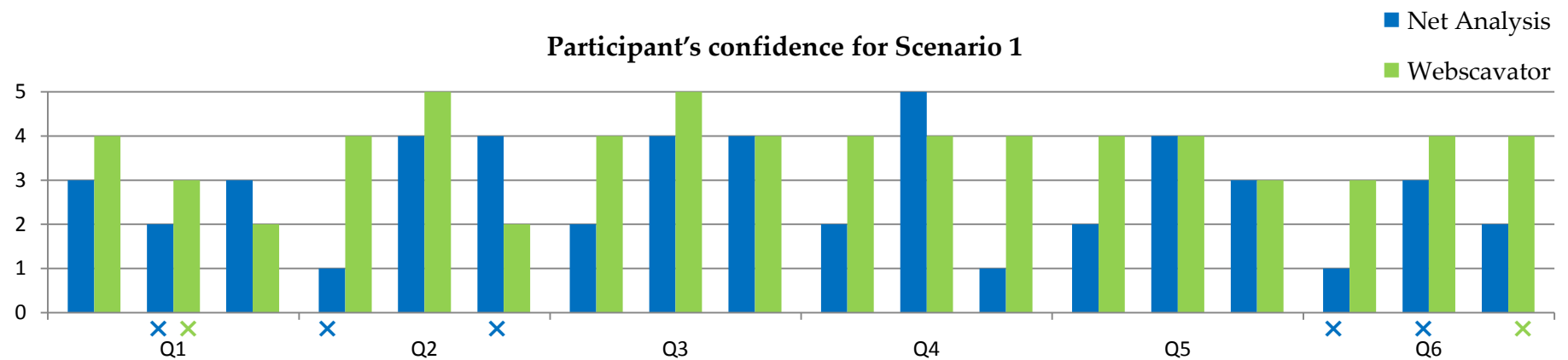
Participants 1 and 2 were digital forensic investigators from a large commercial bank. Participant 1 had some knowledge of Net Analysis as he had used it with a previous job several years earlier. Participant 2 had not used Net Analysis before. Both participants used EnCase as their main web browser history analysis tool. Participant 3 was a digital forensic investigator from the Scottish Crime and Drugs Enforcement Agency and uses Net Analysis extensively to perform web history analysis.

		1	2	3	4	5	6	Total	
Participant 1									
Net Analysis	Correct Answer	✓	✗	✓	✓	✓	✗	4	66.67%
	Confidence	3	1	2	2	2	1	11	37%
	Time	00:24	01:49	00:50	02:57	06:36	06:18	18:54	
Webscavator	Correct Answer	✓	✓	✓	✓	✓	✓	6	100.00%
	Confidence	4	4	4	4	4	3	23	77%
	Time	04:59	02:47	02:00	00:57	03:58	02:20	17:01	
Participant 2									
Net Analysis	Correct Answer	✗	✓	✓	✓	✓	✗	4	66.67%
	Confidence	2	4	4	5	4	3	22	73%
	Time	05:47	03:17	04:36	02:10	01:00	02:45	19:35	
Webscavator	Correct Answer	✗	✓	✓	✓	✓	✓	5	83.33%
	Confidence	3	5	5	4	4	4	25	83%
	Time	01:48	01:35	03:19	01:30	01:27	00:34	10:13	
Participant 3									
Net Analysis	Correct Answer	✓	✗	✓	✓	✓	✓	5	83.33%
	Confidence	3	4	4	1	3	2	17	56.66%
	Time	05:39	04:21	01:34	01:02	00:37	01:03	14:16	
Webscavator	Correct Answer	✓	✓	✓	✓	✓	✗	5	83.33%
	Confidence	2	2	4	4	3	4	19	63.33%
	Time	03:09	01:39	01:28	00:02	02:13	00:11	08:42	

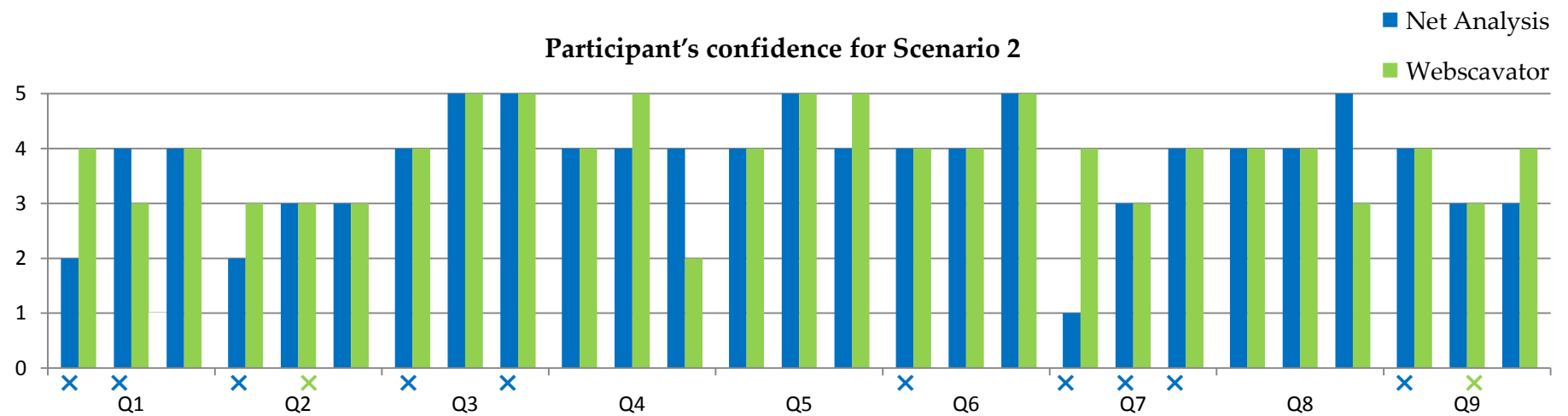
Table 9 - Table of results for Scenario 1

		1	2	3	4	5	6	7	8	9	Total	
Participant 1												
Net Analysis	Correct Answer	×	×	×	✓	✓	×	×	✓	×	3	33%
	Confidence	2	2	4	4	4	4	1	4	4	29	64%
	Time	02:21	01:04	01:15	01:56	00:49	00:25	06:38	02:21	01:08	17:57	
Webscavator	Correct Answer	✓	✓	✓	✓	✓	✓	✓	✓	✓	9	100%
	Confidence	4	3	4	4	4	4	4	4	4	35	78%
	Time	00:27	00:28	02:21	00:58	00:47	00:48	00:29	01:56	01:36	09:50	
Participant 2												
Net Analysis	Correct Answer	×	✓	✓	✓	✓	✓	×	✓	✓	7	78%
	Confidence	4	3	5	4	4	5	4	5	3	37	82%
	Time	04:31	02:44	00:15	00:27	00:34	04:35	01:34	01:51	00:41	17:12	
Webscavator	Correct Answer	✓	×	✓	✓	✓	✓	✓	✓	×	7	78%
	Confidence	3	3	5	5	5	4	3	4	3	35	78%
	Time	00:41	00:18	00:39	01:55	01:03	00:21	00:24	02:11	00:45	08:17	
Participant 3												
Net Analysis	Correct Answer	✓	✓	×	✓	✓	✓	×	✓	✓	7	78%
	Confidence	4	3	5	4	5	4	3	4	3	35	78%
	Time	01:51	00:42	00:29	00:18	02:27	00:19	03:02	02:15	04:02	15:25	
Webscavator	Correct Answer	✓	✓	✓	✓	✓	✓	✓	✓	✓	9	100%
	Confidence	4	3	5	2	5	5	4	3	4	35	78%
	Time	00:43	00:49	00:41	00:27	01:28	00:02	02:03	00:31	01:03	07:47	

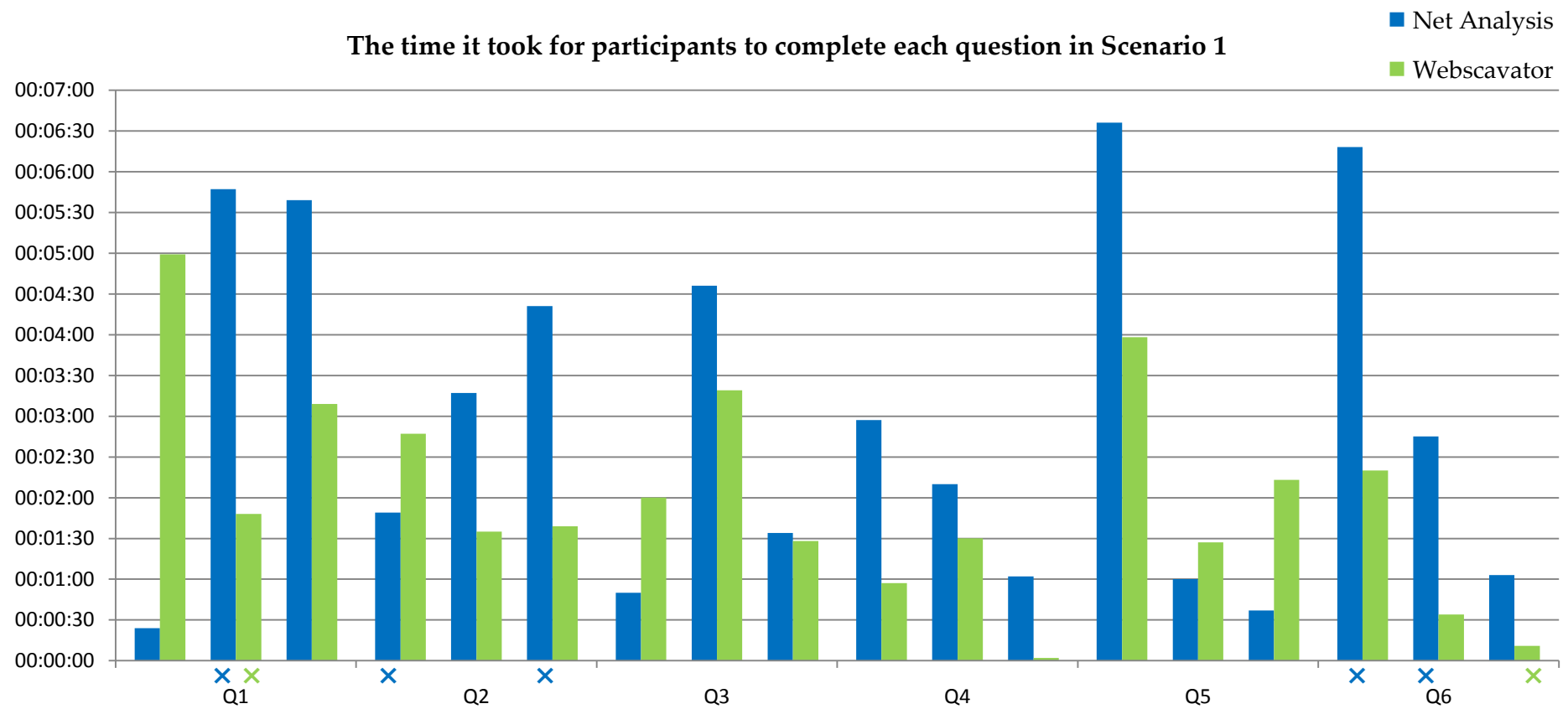
Table 10 - Table of results for Scenario 2



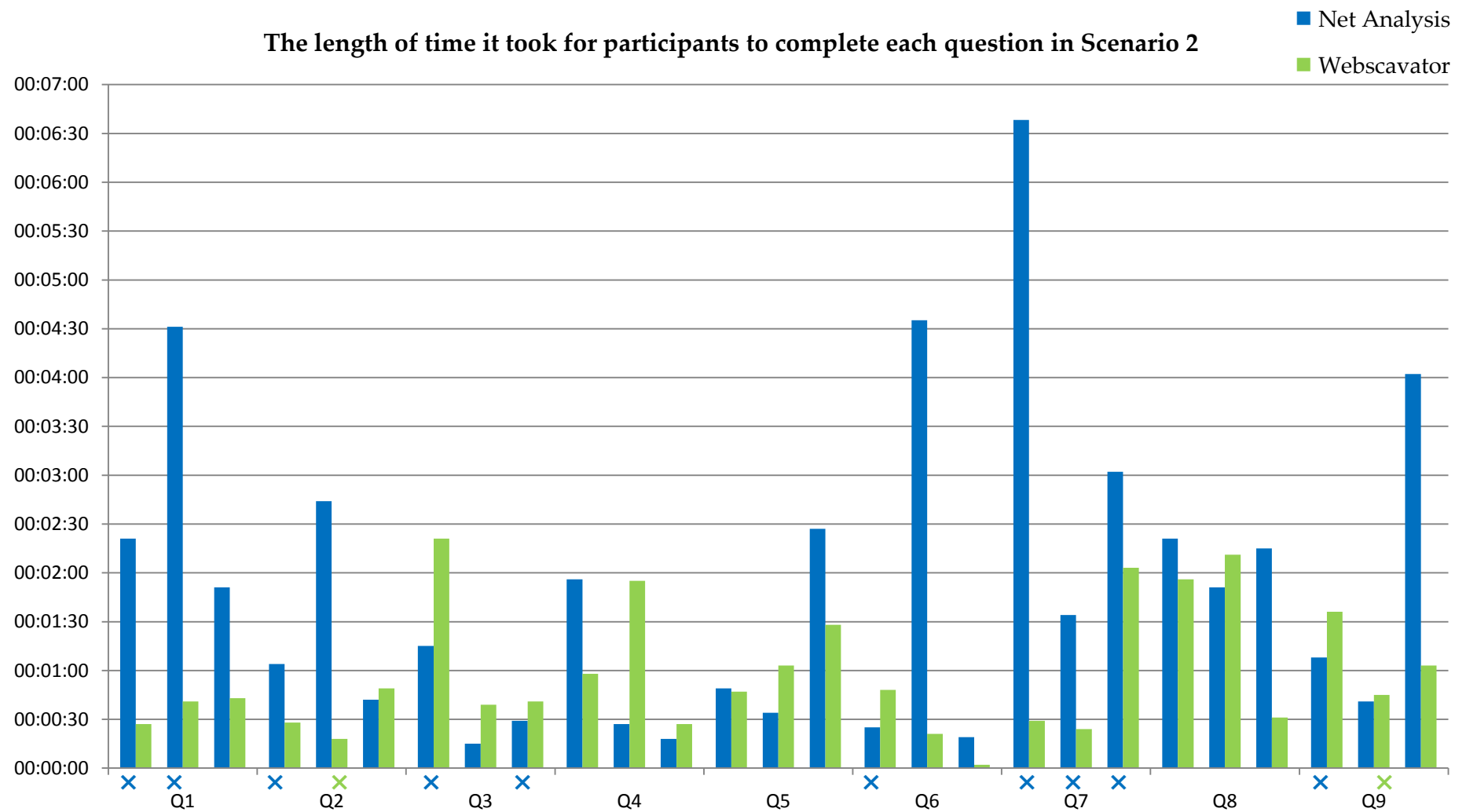
Graph 1 - Bar Chart showing the participant's confidence and errors in their answers for Scenario 1



Graph 2 - Bar Chart showing the participant's confidence and errors in their answers for Scenario 2



Graph 3 - Bar Chart showing the length of time it took to complete each question and the errors made in Scenario 1



Graph 4 - Bar Chart showing the length of time it took to complete each question and the errors made in Scenario 2

7.1 SCENARIO 1 RESULTS

Table 9 shows the amalgamated results for scenario 1. There were six questions in total, concentrating on finding out the rough amount of time spent on particular websites, the top websites visited and finding the periods of heaviest usage / gaps in usage. All three participants took less time and made fewer errors when using Webscavator, and two were more confident in their answers.

A large drawback of using Net Analysis to answer some of the questions is that it does not separate date and time, i.e. participants could not search for a specific time like 16:00. This meant a lot of the participants' time was spent browsing back and forth and counting numbers of entries – all three wrote down rough figures on the answer sheet before totting up a total. This was especially true of Question 6 (estimating how many hours in a week were spent on a particular website), where all three participants were at least three times faster when using Webscavator instead of Net Analysis. Two of the participants also got this answer wrong using Net Analysis, and participant 3, even though he got it right, scored that answer with a very low confidence, indicating this may have been a lucky guess.

Question 2 also caused difficulties in Net Analysis, where the participants were asked to list the top three websites visited over the whole dataset. Net Analysis can only sort domain names alphabetically and does not give any grouping options. This again meant a lot of searching through all the entries, and participant 1 became visibly frustrated with the Net Analysis. Participant 3 gave a 4 for his confidence even though they got the wrong answer. This concurs with the research in Chapter 5 that general impressions of large volumes of data can be wrong if the software does not help you mentally digest the information by presenting a useful summary.

Graph 1 shows the confidences of the participants' answers. In 12 cases, Webscavator scored the higher confidence; three cases they had equal confidence and in three cases Net Analysis scored a higher confidence. Two of the three cases where Net Analysis scored higher were with participant 3. This is not surprising due to this participant's pre-existing familiarity with Net

Analysis. However, despite the higher confidence, the participant got one of the answers wrong and it took them significantly longer to answer both questions when compared to Webscavator.

Graph 3 shows the time spent by each participant on each of the questions. Only 5 out of 18 times was Net Analysis faster than Webscavator. Three out of these five were from participant 1 in the first three questions. It was obvious from observation that participant 1 became quickly frustrated at the fact Net Analysis wouldn't allow him to filter and group in the way that he wanted, and he made quick estimates based on eyeballing the data rather than investigating thoroughly (hence the low confidence). He spent more time trying to get accurate answers on subsequent questions, consequently taking much longer to answer them. Participant 1 also took longer than the other participants to get used to Webscavator, which explains why it took them a comparatively long time to answer the first few questions using it.

Question 5 was answered more quickly by participants 2 and 3 in Net Analysis. The question asked to estimate how long a website was visited during a particular day. In Net Analysis this is very easy to answer, as once a filter is created, the user just needs to subtract the end time from the start time by scrolling down the page (see Figure 48 and Figure 49 for an example). In Webscavator, the user can either read off the times by hovering over the start and end points, or use the time scale on the y-axis to estimate a time period (see Figure 50). Even though both times are already on the screen and no scrolling is needed, this took longer. There are two reasons why this may have occurred. Firstly, to be able to distinguish the data points for the particular website, a filter should have been applied. Both participant 2 and 3 did not use any filters for this question. Secondly, even when the endpoints of the website are found on the graph, it is difficult to follow the grey lines back to the time on the left axis, since lines close together can be difficult to follow (Ambler, 2005). The second problem can be fixed by making horizontal lines thicker or darker when hovered over. In addition, allowing coloured horizontal marker bars to be set along particular points on the y-axis, such as in Figure 51, would make times easier to read.

NetAnalysis v1.51 - Forensic Internet History Analysis

File Filter Searching Tools Bookmarks Reports Audit View Column Help

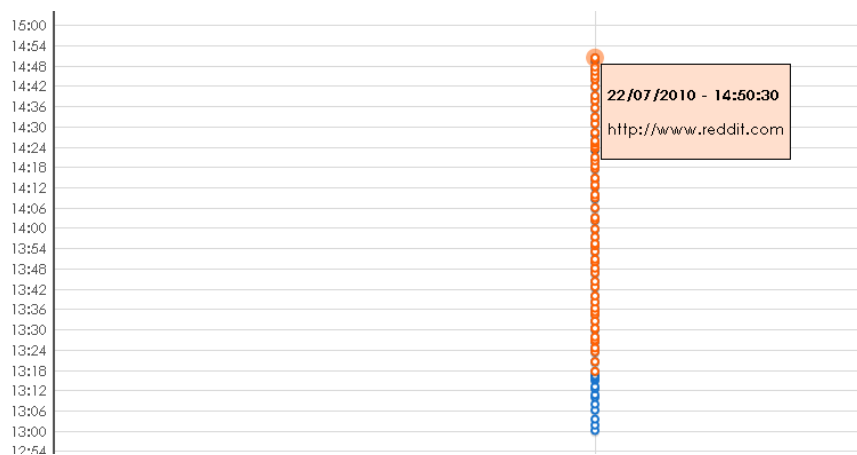
GMT Standard Time [UTC +0000]

Type	Last Visited [UTC]	Last Visited [Local]	Hits	User	URL	Host
http	22/07/2010 13:50:13 Thu	22/07/2010 14:50:30 Thu	12		http://www.reddit.com	www
http	22/07/2010 13:50:13 Thu	22/07/2010 14:50:30 Thu	1		http://www.reddit.com/r/drank/comments/cczj/problem_youre_drinking_and_all_of_the_cookies_you/	www
http	22/07/2010 13:47:44 Thu	22/07/2010 14:47:44 Thu	1		http://www.reddit.com/r/WTF/comments/ccv46/in_response_to_the_controversy_over_the/	www
http	22/07/2010 13:46:26 Thu	22/07/2010 14:46:26 Thu	1		http://www.reddit.com/r/AskReddit/comments/cd2pz/reddit_i_need_to_gain_weight_any_ideas/	www
http	22/07/2010 13:45:03 Thu	22/07/2010 14:45:03 Thu	1		http://www.reddit.com/r/pics/comments/cdebk/hey_reddit_8_and_12_months_ago_i_decided_to/	www
http	22/07/2010 13:44:07 Thu	22/07/2010 14:44:07 Thu	1		http://www.reddit.com/r/todayilearned/comments/cd8iy/tl_spain_was_once_ruled_by_an_inbred_retard/	www
http	22/07/2010 13:41:57 Thu	22/07/2010 14:41:57 Thu	1		http://www.reddit.com/r/pics/comments/cdur3/meanwhile_on_4chan/	www
http	22/07/2010 13:39:15 Thu	22/07/2010 14:39:15 Thu	1		http://www.reddit.com/r/AskReddit/comments/cdk4r/so_if_a_person_is_born_blind_what_do_they_dream/	www
http	22/07/2010 13:37:50 Thu	22/07/2010 14:37:50 Thu	1		http://www.reddit.com/r/AskReddit/comments/cdxs5/haggis_ever_tried_it/	www
http	22/07/2010 13:35:39 Thu	22/07/2010 14:35:39 Thu	1		http://www.reddit.com/r/AskReddit/comments/cdw3k/a_friend_told_me_he_found_usb_flash_drive_with/	www
http	22/07/2010 13:32:56 Thu	22/07/2010 14:32:56 Thu	1		http://www.reddit.com/r/WTF/comments/cdbw1/jesus_dude_im_so_sorry_pic/	www
http	22/07/2010 13:32:52 Thu	22/07/2010 14:32:52 Thu	1		http://www.reddit.com/r/WTF/comments/ce721/reddit_i_just_saw_a_woman_walking_a_goose_wearing/	www
http	22/07/2010 13:31:02 Thu	22/07/2010 14:31:02 Thu	1		http://www.reddit.com/r/AskReddit/comments/ce6aw/hey_reddit_im_a_childrens_book_illustrator/	www

Figure 48 - The end time for the website reddit.com on 22nd July 2010

http	22/07/2010 12:40:04 Thu	22/07/2010 13:40:04 Thu	1		http://www.reddit.com/r/AskReddit/comments/ck4rx/what_are_your_best_worst_resigning_from_work/	www
http	22/07/2010 12:38:15 Thu	22/07/2010 13:38:15 Thu	1		http://www.reddit.com/r/AskReddit/comments/ck3lo/men_of_reddit_what_are_your_menrelated_lifehacks/	www
http	22/07/2010 12:36:25 Thu	22/07/2010 13:36:25 Thu	1		http://www.reddit.com/r/funny/comments/ck1yb/i_have_23_england_jokes_for_you/	www
http	22/07/2010 12:35:20 Thu	22/07/2010 13:35:20 Thu	1		http://www.reddit.com/r/todayilearned/comments/ckeri/today_i_learned_that_there_are_illegal_prime/	www
http	22/07/2010 12:34:34 Thu	22/07/2010 13:34:34 Thu	1		http://www.reddit.com/r/pics/comments/ckc3o/this_is_exactly_what_happened_to_me_this_weekend/	www
http	22/07/2010 12:32:37 Thu	22/07/2010 13:32:37 Thu	1		http://www.reddit.com/r/AskReddit/comments/ck4ez/what_are_the_best_skills_to_learn_preferably_with/	www
http	22/07/2010 12:30:29 Thu	22/07/2010 13:30:29 Thu	1		http://www.reddit.com/r/AskReddit/comments/ckf5g/what_do_you_actually_think_but_are_too_afraid_to/	www
http	22/07/2010 12:30:22 Thu	22/07/2010 13:30:22 Thu	1		http://www.reddit.com/r/todayilearned/comments/ckazh/tl_you_can_make_potato_chips_in_the_microwave/	www
http	22/07/2010 12:27:56 Thu	22/07/2010 13:27:56 Thu	1		http://www.reddit.com/r/pics/comments/ckgc8/its_true_cats_really_dont_give_a_sht/	www
http	22/07/2010 12:27:03 Thu	22/07/2010 13:27:03 Thu	12		http://www.reddit.com	www
http	22/07/2010 12:26:38 Thu	22/07/2010 13:26:38 Thu	1		http://www.reddit.com/r/AskReddit/comments/ckw5l/am_so_upset_i_am_sick/	www
http	22/07/2010 12:24:41 Thu	22/07/2010 13:24:41 Thu	1		http://www.reddit.com/r/pics/comments/cky1t/why_couldnt_i_have_been_shown_this_in_maths_class/	www
http	22/07/2010 12:23:35 Thu	22/07/2010 13:23:35 Thu	1		http://www.reddit.com/r/AskReddit/comments/ckvxy/does_anyone_know_a_trick_to_make_your_brain_shut/	www
http	22/07/2010 12:20:38 Thu	22/07/2010 13:20:38 Thu	1		http://www.reddit.com/r/blog/comments/cky41/neatorama_and_reddit_present_an_unusual_talent/	www
http	22/07/2010 12:17:45 Thu	22/07/2010 13:17:45 Thu	1		http://www.reddit.com/r/AskReddit/comments/ckz61/what_was_office_life_like_before_computers_and/	www

www.digital-detective.co.uk Filter History C:\Documents and Settings\Sarah\...places.sqlite Index: 2488 URL Records: 60

Figure 49 - The start time for the website reddit.com on 22nd July 2010Figure 50 - The same example as above with a highlight filter for reddit.com on 22nd July 2010

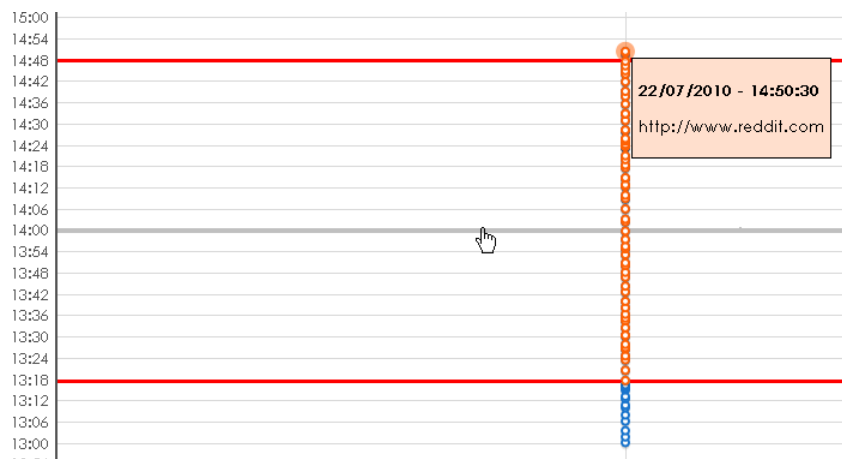


Figure 51 - Example improvement for reading off times

7.2 SCENARIO 2 RESULTS

Table 10 shows the results for scenario 2. In general, the results were similar to that of scenario 1. Using Webscavator was quicker for all three participants. In two out of three cases, the participants got less wrong answers using Webscavator, and in the third case an equal number. Two of the three participants also gave Webscavator better confidence ratings. Participant 3 (who has pre-existing familiarity with Net Analysis) gave both tools equal confidence, but got more answers correct when using Webscavator.

participants were more confident and made fewer mistakes. In the other case Net Analysis equalled Webscavator in confidence/correctness. The speed difference in scenario 2 was much more pronounced than in scenario 1, with each participant taking roughly 8 minutes less using Webscavator regardless of whether scenario 1 or 2 was done first. This meant it took participants roughly half the amount of time (48%-55%) to answer the questions using Webscavator.

Both participant 1 and 3 made no errors when using Webscavator. Participant 2 made 2 errors, the second being quite surprising. His second error occurred on the last question, which asked what sort of pattern of access the website from the previous question followed (i.e. regular, clustered or random). The screenshot in Figure 52 shows what he was given – a webcam

website which auto-refreshed exactly every 2 minutes. Instead of answering ‘very regular’, he put ‘clustered’ – which appears to be correct when zoomed out. Since he only took 45 seconds to answer this, it is likely he did not zoom into the data very much and hover over the points to spot the times were regular. Equally surprising, participant 1 answered this incorrectly in Net Analysis, which when filtered looks like Figure 53 – clearly showing a regular time pattern. The fact that this was not noticed helps confirm that large datasets give data overload – important information is lost amongst all the unnecessary clutter.

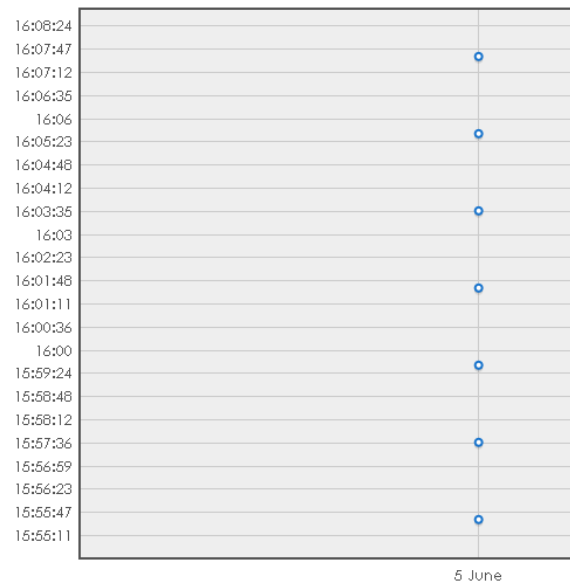


Figure 52 - regularly occurring websites with evenly spaced entries

Type	Last Visited [UTC]	Last Visited [Local]	Hits	User	URL
http	05/06/2010 15:07:37 Sat	05/06/2010 16:07:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 15:05:37 Sat	05/06/2010 16:05:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 15:03:37 Sat	05/06/2010 16:03:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 15:01:37 Sat	05/06/2010 16:01:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 14:59:37 Sat	05/06/2010 15:59:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 14:57:37 Sat	05/06/2010 15:57:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 14:55:37 Sat	05/06/2010 15:55:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 14:53:37 Sat	05/06/2010 15:53:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 14:51:37 Sat	05/06/2010 15:51:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 14:49:37 Sat	05/06/2010 15:49:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 14:47:37 Sat	05/06/2010 15:47:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php
http	05/06/2010 14:45:37 Sat	05/06/2010 15:45:37 Sat	12		http://www.scottishwebcamslive.com/scotland/edinburgh/diamonte_plaza.php

Figure 53 - Same example as above in Net Analysis. The times of the website are circled in red

A question that no one got right in Net Analysis was question 7. This also scored the lowest confidence for each participant when using Net Analysis. Question 7 asked for the three most common search words queried using the search engine in the previous question. Net Analysis has no way of extracting the search terms or phrases searched for with search engines, leaving the user to manually look at URL query strings, which can be very complicated (see Figure 54 for the Net Analysis view). For example:

```
http://www.google.co.uk/search?hl=&q=compare+google+search+with+bing&sourceid=navclient-ff&rlz=1B3GGGL_en-GBGB334GB334&ie=UTF-8
```

The search phrase for this URL is: *compare google search with bing*. To the trained eye this may not be difficult, but there may be hundreds or even thousands of searches to go through – the author's web history contains 23,014 Google searches over the course of 2 years. To analyse these search terms manually is a daunting, straining and error-prone task! The scenario had only 12 searches and yet none of the participants managed to find the top three searched words. The presentation of the data in Net Analysis did not help the investigators: the cluttered screen and the fact they had to process the URLs manually to get the search terms whilst also keeping count proved too difficult. The participants had no problems answering this question correctly

using Webscavator's word cloud, and all gave a high confidence of 4 and took less time than with Net Analysis.

The screenshot shows the NetAnalysis v1.51 interface with a table of filtered internet history data. The table has columns for Type, Last Visited [UTC], Last Visited [Local], Hits, User, URL, and Host. All entries are of type 'http' and have a hit count of 1. The URLs are filtered to include 'bing.com'. The 'Last Visited' times range from 05/06/2010 20:06:59 Sat to 01/06/2010 19:23:43 Tue. The 'Host' column shows 'www.bing.c' for all entries.

Type	Last Visited [UTC]	Last Visited [Local]	Hits	User	URL	Host
http	05/06/2010 20:08:23 Sat	05/06/2010 21:08:23 Sat	1		http://www.bing.com/search?q=diamonte+plaza+100+carat+diamond+on+display&form=QBR&filt=all&qs=n&sk=8&sc=8-10	www.bing.c
http	05/06/2010 20:06:59 Sat	05/06/2010 21:06:59 Sat	1		http://www.bing.com/search?q=diamonte+plaza+diamonds&go=&form=QBR&filt=all&qs=n&sk=8&sc=8-10	www.bing.c
http	05/06/2010 20:06:51 Sat	05/06/2010 21:06:51 Sat	1		http://www.bing.com/search?q=diamonte+plaza+diamond+collection&go=&form=QBR&filt=all&qs=n&sk=8&sc=8-10	www.bing.c
http	05/06/2010 14:45:23 Sat	05/06/2010 15:45:23 Sat	1		http://www.bing.com/search?q=edinburgh+webcam&go=&form=QBR&filt=all&qs=n&sk=8&sc=8-10	www.bing.c
http	01/06/2010 20:50:46 Tue	01/06/2010 21:50:46 Tue	1		http://www.bing.com/maps/default.aspx?q=diamonte+plaza+edinburgh&mk=en-GB&FORM=BYFD%233mNwPTU1Ljk1MDYwOTU4Njg5MDd+...	www.bing.c
http	01/06/2010 20:49:39 Tue	01/06/2010 21:49:39 Tue	1		http://www.bing.com/maps/default.aspx?q=diamonte+plaza+edinburgh&mk=en-GB&FORM=BYFD%233nE9LnF1YXkrGxhemErZWRpbm31Z...	www.bing.c
http	01/06/2010 20:47:39 Tue	01/06/2010 21:47:39 Tue	1		http://www.bing.com/maps/default.aspx?q=diamonte+plaza&mk=en-GB&FORM=BYFD	www.bing.c
http	01/06/2010 20:06:23 Tue	01/06/2010 21:06:23 Tue	1		http://www.bing.com/search?q=diamonte+plaza+opening+times&go=&form=QBR&filt=all&qs=n&sk=8&sc=8-10	www.bing.c
http	01/06/2010 20:02:33 Tue	01/06/2010 21:02:33 Tue	1		http://www.bing.com/images/search?q=diamonte+plaza&go=&form=QBR&qs=n&sk=8&sc=1-16%23focal=961b6f55562c236ae3b0f4a6be7f...	www.bing.c
http	01/06/2010 19:49:02 Tue	01/06/2010 20:49:02 Tue	1		http://www.bing.com/images/search?q=diamonte+plaza&go=&form=QBR&qs=n&sk=8&sc=1-16%23focal=4385805494765f1db1c667e357a...	www.bing.c
http	01/06/2010 19:45:43 Tue	01/06/2010 20:45:43 Tue	1		http://www.bing.com/images/search?q=diamonte+plaza&go=&form=QBR&qs=n&sk=8&sc=1-16%23focal=bfcbb4d4098903794f10b3858b18...	www.bing.c
http	01/06/2010 19:23:43 Tue	01/06/2010 20:23:43 Tue	1		http://www.bing.com/images/search?q=diamonte+plaza&go=&form=QBR&qs=n&sk=8&sc=1-16	www.bing.c

Figure 54 - Net Analysis results are filtering for 'bing' in the URL

Question 3 proved to be a problem for participants 1 and 3 using Net Analysis. This question asked how many files the suspect accessed on the D drive. Both participants had knowledge of the Net Analysis advanced filter *Files Accessed on Drive other than 'C'*, which showed them files accessed on the H and D drive. Both participants miscounted the number of D drive files and put the wrong answer down despite giving very high confidences. Because Net Analysis does not group information or give summaries, it is easy for mistakes like this to slip into real cases. Data in actual cases will of course be checked and verified, but simple slipups like this may delay investigations and cause incorrect assumptions right from the start, especially if a quick web history triage is done where more mistakes are likely. The participants all took longer with this question using Webscavator. A reason for this may be that the 'files' tab had not been shown to the participants before the testing. A quick demo had been prepared to showcase Webscavator and Net Analysis, however this was a Firefox file that did not contain any local file accesses, so the files tab was not used. Even so, they all got this question correct with very high confidences.

Graph 4 shows the time it took to answer the questions. Most of the questions in Webscavator took less than a minute to answer, with eight being under 30 seconds. Six Net Analysis questions were answered in under 30 seconds, but ten took more than two minutes compared

with only two with Webscavator. Comparatively, question 1, followed by question 7, took far longer in Net Analysis than Webscavator. Question 1 involved finding the most common time the suspect went on the internet on Monday-Thursdays. As explained earlier, Net Analysis cannot show websites ordered by time, and a lot of the participants time involved scrolling and looking for patterns. Participants 1 & 2 got this answer wrong. Conversely, pattern recognition was easy in Webscavator, with all three participants taking less than 45 seconds and giving the correct answer with a confidence of 4.

Figure 55 shows the heat-map with the data from one of scenario 2's datasets – it is clear to see that on Monday–Thursday, 11pm to midnight was the most common time the suspect was on the internet.

	Mon	Tue	Wed	Thu	Fri	Sat	Sun
00:00 - 00:59					8		
01:00 - 01:59							
02:00 - 02:59							
03:00 - 03:59							
04:00 - 04:59							
05:00 - 05:59							
06:00 - 06:59							
07:00 - 07:59							
08:00 - 08:59							
09:00 - 09:59							
10:00 - 10:59							
11:00 - 11:59							
12:00 - 12:59							
13:00 - 13:59							28
14:00 - 14:59						44	23
15:00 - 15:59						89	28
16:00 - 16:59						21	32
17:00 - 17:59						68	23
18:00 - 18:59					138	39	33
19:00 - 19:59	8			10	117	22	
20:00 - 20:59	29	20	11	5			7
21:00 - 21:59	44	25	33			3	31
22:00 - 22:59	56	44	31				48
23:00 - 23:59	122	218	139	221			

Figure 55 – Heat-map showing the most common times the suspect was on the internet

7.3 QUANTATIVE DATA SUMMARY AND CONCLUSIONS

Table 11 shows the results of the two scenarios together. In all accounts, using Webscavator gave more correct answers, more confident answers and in a faster time.

The lowest scorer with Net Analysis was participant 1, who had used Net Analysis for several years in a previous job. They took roughly the same amount of time as participant 2, who had never used Net Analysis before. This suggests Net Analysis may need to be re-learned if it is not used often. However, as this suggestion is based only on the experience of one user, more investigation would be needed to confirm or refute it. Interestingly, participant 3 (who uses Net Analysis frequently) gave the least amount of '5' confidences for Net Analysis, none in scenario 1 and only 2 in scenario 2.

		Net Analysis	Webscavator
Participant 1	Correct Answer	47% (8 errors)	100% (0 errors)
	Confidence	53%	77%
	Time	36:51	26:51
Participant 2	Correct Answer	73% (4 errors)	80% (3 errors)
	Confidence	79%	80%
	Time	36:47	18:30
Participant 3	Correct Answer	80% (3 errors)	93% (1 error)
	Confidence	69%	72%
	Time	29:41	16:29

Table 11 - Summary of Scenario Results

The main difficulties in Net Analysis compared to Webscavator were:

- **Pattern recognition** - especially spotting gaps or dense packed time periods. Webscavator provided many ways to spot patterns, particularly the heat-map and the timeline. Webscavator performed better and faster than Net Analysis for all pattern recognition questions.
- **Summarising data** – counting even small sets of data gave wrong answers. Webscavator gave more accurate and confident results for summarising local file accesses and giving the top three websites visited. The file accesses took slightly longer, but this may be due to the participants not having seen the files tab previously.

- **Interpreting search engine queries** - Net Analysis provides no extraction of search terms, meaning the user has to mentally separate out the terms. Webscavator provided a word cloud of search terms, which gave more accurate and faster results.
- **Information overload** – there were a few mistakes in Net Analysis due to just not spotting the relevant information. Webscavator’s filters are more fine grain and allow for exclusion or highlighting of information. Zooming in also gets rid of irrelevant noise. Information is only visible when the user requires it, i.e. by hovering, clicking or checking “show labels”, whereas this information is generally always shown in Net Analysis. Webscavator performed slightly better and with more confidence on questions requiring timeline usage.

The main difficulty in Webscavator was accurately measuring time periods. This is mainly due to the participants not using filters to highlight the relevant data. Filters was not explained during the Webscavator demo, and so this may explain why two out of the three participants did not try and use them.

In conclusion, the visualisations seemed to provide cognitive and perceptive benefits listed in Chapter 5 compared with the tabular format of Net Analysis.

7.4 USABILITY RESULTS

After the user testing tasks, the participants were given a questionnaire to fill in and an interview. The interview consisted of discussing further work for Webscavator (available in the next chapter) and their general thoughts on the tasks. The consensus was that Webscavator was much easier to use and get to grips with than Net Analysis, and presented the data in a much more pleasing and useful format.

7.4.1 QUESTIONNAIRE RESULTS

The questionnaire consisted of three sets of 10 questions with 5 ratings ranging from strongly disagree to strongly agree. The first two sets of questions used Brooke’s *System Usability Scale* (SUS) on Net Analysis followed by Webscavator. The SUS scale can be converted into a number

between 0 to 100 giving an overall sense of the usability of the program (Brooke, 1996). Kortum and Bangor developed an adjective scale based on the SUS to give a better understanding on what the SUS scale means (Kortum & Bangor, 2009). The results are in Table 12 and Figure 56.

As predicted, Net Analysis scored lower in usability. Participant 2 had no previous knowledge of Net Analysis, and was strongly put off by its performance in the tasks previously, and very impressed by Webscavator's usability and functionality, giving the lowest and highest scores for both respectively.

	Net Analysis		Webscavator	
Participant 1	57.5	Good (Low)	72.5	Good/Excellent
Participant 2	40	OK (Not Acceptable)	85	Excellent/Best Imaginable
Participant 3	55	Good (Low)	67.5	Good (High)

Table 12 - SUS and adjective scale for Net Analysis and Webscavator

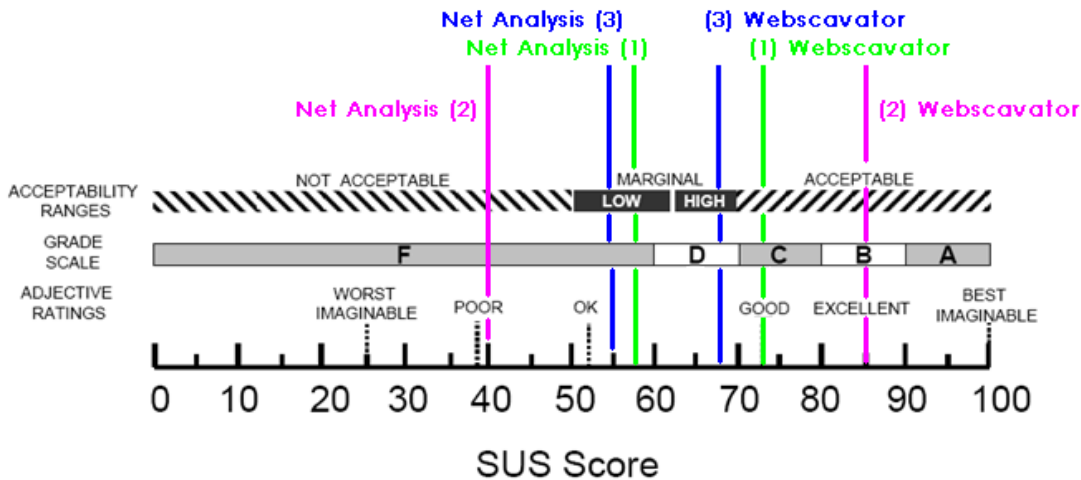


Figure 56 - Where the results appear on the SUS & adjective scale. Image modified from (Kortum & Bangor, 2009)

Participant 1 was a little hesitant when it came to Webscavator, commenting on the question sheet *“very usable but would need to verify results”* on ‘Q9 – I felt very confident using the system’, marking his answer as ‘neither agree or disagree’. Unfortunately, this is a functional problem

relating to confidence in the program's correctness rather than its ease of use and is not a usability problem. If the result was changed to *strongly agree* (to match his comment), Webscavator would have gotten an even better score of 77.5/Excellent.

Participant 3 did not score Webscavator as highly as participants 1 or 2 did. Unsurprisingly he also gave Net Analysis the highest score out of the three, possibly because he knew the software best. The question that he gave the lowest score on Webscavator was Q5, "*I found the various functions in this system were well integrated*", marking this as *disagree*. This is understandable, as each visualisation performs a very separate function. However, all visualisations are linked by the filters and common data and both other participants answered this question with *agree*. There is certainly much scope for enhancement by linking the visualisations more closely.

The final 10 questions were a functionality evaluation for Webscavator to determine if each visualisation was considered useful and would help with investigations. The results are in Table 13.

		Strongly Disagree	Disagree	Neither disagree nor agree	Agree	Strongly Agree
1	I think the system overall visualises web browser history well				3	
2	I think the heat-map on the overview will be useful in web history analysis			1	1	1
3	I think the timegraph will be useful in web history analysis				3	
4	I think the bar chart of top visited domain names will be useful in web history analysis				3	
5	I think the search terms word cloud will be useful in web history analysis				3	
6	I think the file directory tree will be useful in web history analysis				2	1
7	Using the system will cut down time spent looking at web history		1		2	

8	I think I will make less errors if I used the system for web history analysis	1			2	
9	I will be able to explain my web history findings more easily to non-technical people with the output from this system				3	
10	I think the system will help me focus my attention more quickly on relevant parts of the web history			1	2	

Table 13 - Results of Webscavator Functionality Evaluation

The first six questions asked whether each visualisation would be useful for web history analysis. Although opinions were divided over the heat-map, every other visualisation scored an *agree* or *strongly agree*. The final four questions enquired about efficiency, errors and if the visualisations would help to explain results to non-technical people. Most of the answers were *agree*.

One of the important aims of Webscavator was to make sure its visualisations were easy to understand by non-technical or non-forensically trained people. Question 9, which asked this, scored a unanimous *agree*. All participants agreed that with some reporting tools built in (e.g. the ability to turn the visualisations into JPEGs and produce PDFs summarizing the data) Webscavator would be incredibly invaluable for explaining results to police officers and lawyers, and would even make a good learning tool for computer forensics students.

Participant 3 gave all four of the lowest scores, including a *strongly disagree* for question 8. This ties in with their relatively low usability scoring mentioned on the previous pages. This is interesting given that participant 3 answered questions more accurately when using Webscavator instead of Net Analysis. In the interview following the scenarios, participant 3 commented on the fact that some of the questions were infrequently asked during a real investigation (although none of the other participants mentioned this). However, he understood that with a lack of experience investigating real cases it is difficult to write realistic scenarios that cover all the different attributes of web history and also test the different parts of Webscavator. Whilst participant 3 may not have been fully confident and correct with his answers for the scenarios, he has many years experience working on real cases. Therefore, given

that he already perceives his error rate to be low, it may be difficult for him to see how using an extra tool would improve things, which could explain the *strongly disagree*.

CHAPTER 8: CONCLUSIONS

8.1 ACHIEVEMENT OF AIMS

1. *How do current forensic investigators use web history information and what are the current problems with the tools available?*

This was achieved in Chapter 3 and 4. Three digital forensic investigators were interviewed about how they use web history and the problems they have with the current tools they use. These tools were investigated to form an idea of the current state of web history analysis tools.

2. *What visualisations would be appropriate?*

Chapter 5 is a literature review of why visualisations in general are useful, followed by an analysis of several different attempts at visualisation tools for digital forensics. Finally, an investigation was carried out into what visualisations would be appropriate for displaying web history. The opinions of the interviewed investigators, along with the knowledge gained during the literature review, fed into the design of Webscavator, the visualisation tool developed.

Chapter 6 describes the included visualisations, why they were chosen and how they were created.

3. *Are visualisations for web history useful and an improvement on what is currently available?*

User testing was carried out to test the hypothesis that visualisations would indeed help forensic investigators. Webscavator was tested against Net Analysis, the most commonly used forensic web history analysis program. Chapter 7 analyses the results of usage and usability experiments undertaken with three forensic investigators.

Overall, the participants made fewer mistakes, were more confident and took less time when using Webscavator. Several of the problems discussed as part of the the problem statement in Chapter 2 and in Chapter 4 manifested when the participants used Net Analysis, but disappeared when using Webscavator. These include difficulty in finding correlations and patterns in web history, information overload and difficulty in obtaining a summary overview of data. The usability and functionality questionnaires clearly show Webscavator is easier than Net Analysis to use, and that it would be a useful tool in digital forensics in terms of helping to cut down time spent looking at web history, making less errors, focusing attention and especially explaining results more easily to others. All three investigators expressed a wish to use Webscavator to help with their future professional investigations.

4. *Can this be generalised?*

The domain of web history analysis was chosen because investigation showed it to be an area much in need of visual tools. Other areas of digital forensics share many of the same problems and attributes:

1. **Similar tools:** The output from most forensic tools is either a structured file or a spreadsheet embedded in the tool's user interface. Most tools do not intelligently process data, but instead just output it for the user to analyse manually.
2. **Similar questions:** A lot of questions involve finding patterns and anomalies, summarising data or obtaining precise details of just a few parts of a large data set.

As it has been shown that visualisations can help with these issues when investigating web history, it is likely that similar visualisations will also prove useful when applied to other areas of digital forensics. Experiments carried out by others using pre-existing forensic visualisation tools (described in Chapter 5) provide evidence for this claim.

Having only three investigators makes it difficult to generalise the results of the experiments presented here. However, the consistent results and general agreement amongst the participants provide strong evidence that well-chosen visualisations can play an important and useful role in digital forensics.

8.2 RECOMMENDATIONS

The results of this research suggest that visualisations should be taken seriously by the producers of forensic software. Web Historian has taken the first step by adding a crude timeline and pie chart (see page 38, Figure 14), but more effort is needed to make this usable. Visualisations free up time analysing the data and help investigators make fewer errors. There are hundreds of tools available that perform many different forensics functions, but so far not many tools have utilised visualisation. Given that well-chosen visualisations can help significantly in interpreting the large volumes of data produced during digital forensics, it is likely the most effective future tools will include visualisation.

8.3 FURTHER WORK

8.3.1 WEBSCAVATOR IMPROVEMENTS

Several improvements were discussed during the interviews following on from user testing. In particular, these included:

- **Improve the filters** to be able to view, edit and delete them. Currently filters can only be added and it is not possible to see the query behind the filter.

- **Add reporting.** For this dissertation Webscavator was just a visualisation tool, however to be used in practice it needs to be able to generate reports and images. Summaries in PDF format and screenshots as JPEGs would be helpful to investigators.
- **Ability to draw coloured lines** across the timeline to help eyes follow lines more clearly. This was discussed on page 90 in Figure 51. Another improvement for the timeline is to add days of the week along the top.
- **Make the heat-map filterable.** Currently it is just a summary of all the data, but the participants thought this would be useful filtered too.
- **Integrate the visualisations more.** Following on from usability questionnaire results, more could be done to integrate the different visualisations. For example clicking on a search term in the word cloud or a domain name in the bar chart could automatically highlight them in the timeline.

A couple of filter options were not completed due to lack of time such as the ability to perform a fuzzy search (matches values that are homophones and spelt incorrectly) and a periodical search (matches values that are at regular intervals apart). These did not hinder the user testing but would be valuable filters. Many people misspell search terms (especially since the likes of Google auto correct spellings), and by filtering on an exact word some searches with spelling errors may be missed. Many employees in a company subscribe to regular, automatic news tickers or RSS feeds, and it would be useful to locate these and remove them from the results.

8.3.2 FURTHER TESTING

Further scenario testing with more forensic investigators would enhance the reliability of the results. Doing testing with non-forensic investigators would see if Webscavator was easy to use for those who are not familiar with web history analysis – making it a good tool to introduce in digital forensics courses.

Where possible, scenarios involving real (or as realistic as possible) data and realistic scenarios would be preferable. Best would be scenarios made up by forensic investigators, drawing on their experience to the types of questions asked.

APPENDIX

BIBLIOGRAPHY

AFP. (2008, June 18). *Briton Googled 'how to kill' days before murders: court*. Retrieved February 14, 2010, from http://afp.google.com/article/ALeqM5hNX7099fMvF7_qHCpy1Bz4VhtR6A

Ambler, S. W. (2005). *The Elements of UML 2.0 Style*. Cambridge University Press.

Brooke, J. (1996). SUS: a "quick and dirty" usability scale. *Usability Evaluation in Industry*. .

Buchholz, F., & Falk, C. (2005). Design and Implementation of Zeitline: a Forensic Timeline Editor. *Digital Forensic Research Workshop (DFRWS)* , pp. 1-7.

Bunting, S. M. (2010). *Understanding index.dat Files*. Retrieved June 11th, 2010, from Computer Forensics Resources: http://www.stevebunting.org/udpd4n6/forensics/index_dat1.htm

Chromium. (2010a). *Chromium Developement Website*. Retrieved June 10th, 2010, from <http://dev.chromium.org/chromium-projects>

Chromium. (2009b, July 14th). *Issue 16705: User Data Grows Too Large (~2GB)*. Retrieved June 10th, 2010, from <http://code.google.com/p/chromium/issues/detail?id=16705>

Chromium. (2010b). *Source code for page_transition_types.h*. Retrieved June 10th, 2010, from http://src.chromium.org/viewvc/chrome/trunk/src/chrome/common/page_transition_types.h

Chromium. (2009a, October 13th). *Source Code for Time.cc*. Retrieved June 10th, 2010, from <http://src.chromium.org/viewvc/chrome/trunk/src/base/time.cc?view=markup>

Cooley, R., Mobasher, B., & Srivastava, J. (1999). Data preparation for mining world wide web browsing patterns. *Knowledge and Information systems*. Vol 1, Issue 1. , pp. 5-32.

Craigier, P., & Burke, P. (?). *Mac Forensics: Mac OS X and the HFS+ File System*. National Center for Forensic Science.

Cunningham, J. (2007). Analyzing Safari 2.x Web Browser Artifacts using SFT.

DigitalDetective. (2010). *Net Analysis*. Retrieved June 15th, 2010, from <http://www.digital-detective.co.uk/netanalysis.asp>

Dix, A., Finlay, J., Abowd, G. D., & Beale, R. (2003). *Human Computer Interaction*. Prentice Hall.

Fei, B. K. (2007). *Data Visualisation in Digital Forensics Masters Thesis*. Computer Science Dept, University of Pretoria.

Ferox, W. (2009, June 8th). *Safari 4.0 How to turn off Top Sites and reclaim CPU cycles*. Retrieved June 11th, 2010, from <http://imnotbruce.blogspot.com/2009/06/safari-40-stealing-cpu-cycles.html>

Friendly, M. (2009). Milestones in the history of thematic cartography, statistical graphics, and data visualization. *National Sciences and Engineering Research Council of Canada* .

Gudjonsson, K. (2010, January 27th). *Updates to log2timeline*. Retrieved June 11th, 2010, from IR and Forensics Talk: <http://blog.kiddaland.net/2010/01/updates-to-log2timeline/>

Hassan-Montero, Y., & Herrero-Solana, V. (2006). Improving Tag-Clouds as Visual Information Retrieval Interfaces. *International Conference on Multidisciplinary Information Sciences and Technologies* .

Healey, C. G. (2009). *Perception in Visualization*. Department of Computer Science, North Carolina State University.

Hendee, W. R., & Wells, P. N. (1997). *The perception of visual information*. Springer.

Herder, E., & Weinreich, H. (2005). Interactive web usage mining with the navigation visualizer. *CHI '05 extended abstracts on Human factors in computing systems* .

Hitz, O., Robadey, L., & Ingold, R. (2000). Visualization of document recognition results using XML technology. *Colloque International sur le Document Electronique* , 207-215.

Jones, K. (2003). *Forensic analysis of internet explorer activity files*.

Kortum, P., & Bangor, A. (2009). Determining What Individual SUS Scores Mean: Adding an Adjective Rating Scale. *Journal of Usability Studies Vol. 4, Issue 3* , pp. 114-123.

Krasser, S., Conti, G., Grizzard, J., Gribschaw, J., & Owen, H. (2005). Real-time and forensic network data analysis using animated and coordinated visualization. *Proceedings from the Sixth Annual IEEE Systems, Man and Cybernetics (SMC) Information Assurance Workshop* , pp. 42-49.

Kruse, W. G., & Heiser, J. G. (2001). *Computer Forensics Incident Response Essentials*. Addison-Wesley Professional.

Labroche, N., Lesot, M.-J., & Yaffi, L. (2007). A New Web Usage Mining and Visualization Tool. *19th IEEE International Conference on Tools with Artificial Intelligence* , pp. 321-328.

Larkin, J., & Simon, H. (1987). Why a diagram is (sometimes) worth 10,000 words. *Cognitive Science, Volume 11* , pp. 65-99.

Lohmann, S., Ziegler, J., & Tetzlaff, L. (2009). Comparison of Tag Cloud Layouts: Task-Related Performance and Visual Exploration. *Lecture Notes in Computer Science, Volume 5726* .

MacScripter. (2006). *MacScripter - Sort Safari history by date*. Retrieved June 10th, 2010, from <http://macscripter.net/viewtopic.php?id=17989>

Mandiant. (2010). *Company Overview*. Retrieved June 15th, 2010, from http://www.mandiant.com/about/company_overview/

Marcella, A. J., & Menendez, D. (2007). *Cyber forensics: a field manual for collecting, examining and preserving evidence of computer crimes*. CRC Press.

Marchionini, G. (1997). *Information Seeking In Electronic Environments*. Cambridge University Press: Issue 9 of Cambridge series on human-computer interaction.

Mayer, M., & Bederson, B. (2001). Browsing Icons: A Task-Based Approach for a Visual Web History. *HCIL-200119, CS-TR-4308, UMIACS-TR-2001, Volume 85*.

Mil Incorporated. (2009). *Delete index.dat files*. Retrieved June 11th, 2010, from http://www.milincorporated.com/a_indexdat.html

Mozilla Developer Center. (2005). *nsNavHistory.cpp*. Retrieved June 9th, 2010, from <http://mxr.mozilla.org/mozilla-central/source/toolkit/components/places/src/nsNavHistory.cpp>

Mozilla Developer Center. (2008b, June 8th). *PRTime*. Retrieved June 9th, 2010, from <https://developer.mozilla.org/en/PRTime>

Mozilla Developer Center. (2008a, June 10th). *The Places frecency algorithm*. Retrieved June 9th, 2010, from https://developer.mozilla.org/en/The_Places_frecency_algorithm

Murr, M. (2009, September 10th). *The Meaning of LEAK Records*. Retrieved June 11th, 2010, from Forensic Computing: <http://www.forensicblog.org/2009/09/10/the-meaning-of-leak-records/>

Nielsen, J. (2006, June 26th). *Quantitative Studies: How Many Users to Test?* Retrieved June 30th, 2010, from http://www.useit.com/alertbox/quantitative_testing.html

Nielsen, J. (1994). *Usability Engineering*. San Diego: Academic Press.

- Nishimura, K., & Hirose, M. (2007). The Study of Past Working History Visualization for Supporting Trial and Error Approach in Data Mining. *Human Interface and the Management of Information. Methods, Techniques and Tools in Information Design* , pp. 327-334.
- North, C., Stasko, J. T., Fekete, J.-D., & van Wijk, J. J. (2008). The Value of Information Visualisation. *Information Visualization: Human-Centered Issues and Perspectives* .
- Olsson, J., & Boldt, M. (2009). Computer forensic timeline visualization tool. *Digital Investigation, Volume 6* , pp. S78-S87.
- Pereira, M. T. (2009). Forensic analysis of the Firefox 3 Internet history and recovery of deleted SQLite records. *Digital Investigation. Vol 5. Issue 3-4* , pp. 93-103.
- Phan, D. (2007). *Supporting the visualization and forensic analysis of network events*. PhD in Computer Science Thesis, University of Stanford.
- Plaisant, C. (2004). The Challenge of Information Visualization Evaluation. *Proceedings of the working conference on Advanced visual interfaces* , pp. 109-116.
- Plaisant, C., Milash, B., Rose, A., Widoff, S., & Shneiderman, B. (1996). LifeLines: visualizing personal histories. *Proceedings of the SIGCHI conference on Human factors in computing systems: common ground* , pp. 221-227.
- SANS Computer Forensics. (2010, January 21st). *Google Chrome Forensics*. Retrieved June 10th, 2010, from <http://blogs.sans.org/computer-forensics/2010/01/21/google-chrome-forensics/>
- Schwartz, M., & Liebrock, L. (2008). A Term Distribution Visualization Approach to Digital Forensic String Search. *Proceedings of the 5th international workshop on Visualization for Computer Security* , pp. 36-43.
- Shneiderman, B. (1996). The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. *IEEE Symposium on Visual Languages* , pp. 336-343.

SQLite. (2010). *SQLite.org*. Retrieved June 9th, 2010, from <http://www.sqlite.org/>

Srivastava, J., Cooley, R., Deshpande, M., & Tan, P. (2000). Web usage mining: Discovery and applications of usage patterns from web data. *ACM SIGKDD Explorations Newsletter, Vol 1, Issue 2* .

Stamps, A. S., Franck, J., Carver, J., Jankun-Kelly, T., Wilson, D., & Swan, J. E. (2009). A Visual Analytic Framework for Exploring Relationships in Textual Contents of Digital Forensics Evidence. *Proceedings of Workshop on Visualization for Cyber Security* , pp. 39-44.

Sternberg, R. J., & Mio, J. (2005). *Cognitive psychology*. Cengage Learning.

Takada, T., & Koike, H. (2002b). MieLog: A Highly Interactive Visual Log Browser Using Information Visualization and Statistical Analysis. *Sixteenth USENIX Large Installation System Administration Conference* , pp. 133-144.

Takada, T., & Koike, H. (2002a). Tudumi: information visualization system for monitoring and auditing computer logs. *Proceedings Sixth International Conference on Information Visualisation* , pp. 570-576.

Teerlink, S. (2004). A Graphical Representation of File Statistics for Computer Science, Masters Thesis. *Computer Science Dept., University of Utah* .

Teerlink, S., & Erbacher, R. (2006b). Foundations for Visual Forensic Analysis. *2006 IEEE Information Assurance Workshop* , pp. 192-199.

Teerlink, S., & Erbacher, R. (2006a). Improving the Computer Forensic Analysis Process through Visualization. *Communications of the ACM, Volume 49. Issue 2* , pp. 71-75.

Tufte, E. R. (2001). *The Visual Display of Quantitative Information*. Graphics Press USA.

Weinstein, J. N. (2008). A Postgenomic Visual Icon. *Science Vol. 319. no. 587* , pp. 1772 - 1773.

Wikipedia. (2010, June 7th). *Usage share of web browsers*. Retrieved June 11th, 2010, from http://en.wikipedia.org/wiki/Usage_share_of_web_browsers

Wilkinson, L., & Friendly, M. (2009). The history of the cluster heat map. *The American Statistician* .

Youssefi, A., Duke, D., & Zaki, M. (2004). Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters. *ACM New York*, (pp. pp. 394–395). NY, USA.