

CRANFIELD UNIVERSITY

ANNE DAVID

SMURF: A RELIABLE METHOD FOR
CONTEXTUALISING SOCIAL MEDIA ARTEFACTS

DEFENCE AND SECURITY

PhD

Academic Year: 2020

Supervisors: Dr. Sarah Morris, Dr. Gareth Appleby-Thomas
February 2020

CRANFIELD UNIVERSITY

DEFENCE AND SECURITY

CENTRE FOR ELECTRONIC WARFARE,
INFORMATION, AND CYBER

PhD

Academic Year: 2020

ANNE DAVID

SMURF: A RELIABLE METHOD FOR
CONTEXTUALISING SOCIAL MEDIA ARTEFACTS

Supervisors: Dr. Sarah Morris, Dr. Gareth Appleby-Thomas
February 2020

© Cranfield University 2020. All rights reserved. No part of
this publication may be reproduced without the written
permission of the copyright owner.

Abstract

This research aims to evaluate whether artefacts other than the content of user communication on social media can be used to attribute actions or relationships to a user. Social Media has enhanced the way users communicate on the Internet; providing the means for users to share content in real-time, and to establish connections and social relationships with like-minded individuals. However, as with all technology, social media can be leveraged for disagreeable and/or unlawful activities such as cyber bullying, trolling, grooming, or luring.

There are reported cases where evidence from social media was used to secure convictions; for example, the tragic cases of Ashleigh Hall in 2009 and Kayleigh Haywood in 2015. The social media evidence e.g. the messages sent to the victim to arrange a meet up was used to link the suspect to the victim, and attribute actions to the suspect; in addition to other physical evidence presented as part of the case. Investigations with elements of social media is growing within digital forensics. This reinforces the need for a technique that can be used to make inferences about user actions and relationships, especially during a live triage investigation where the information needs to be obtained as quickly as possible.

This research evaluates the use of live triage in the investigation of social media interactions, in order to determine the reliability of such a technique as a means of contextualising user activity, and attributing relationships or actions to a user. This research also evaluates the reliability of artefacts other than the actual content exchanged on social media; in the event that the content of communication is not immediately accessible/available to the investigator. To achieve this, it was important to break down the events that occur before, during and after user activity on social media; followed by the determination of what constitutes communication content in the context of this research.

This research makes the following contributions: establishes a method for the categorisation of social media artefacts based on perceived user activity; communication content was characterised, thus highlighting evidential data of interest from user social media activity; the criteria for assessing the reliability of social media artefacts in a live triage investigation was proposed; a novel framework for social media investigation was developed with a Proof of Concept (PoC) to test its viability. The PoC demonstrates that it is possible to attribute actions or relationships to a user, using artefacts other than the actual content exchanged on social media.

Keywords: live triage, social media, digital forensics, user relationship attribution.

Acknowledgements

Thanks be to God for His preservation, and for making it possible for me to get to where I am.

I would like to thank my supervision team Dr Sarah Morris and Dr Gareth Appleby-Thomas for their guidance and support.

I would also like to thank the members of my thesis committee, Professor Mark Richardson and Dr Duncan Hodges.

Thank you to Melissa Hadgkiss for sorting access to the labs.

Thanks to my family for their support and patience. Special thanks to “diribe” for looking after me.

Contents

Abstract	i
List of Figures	viii
List of Tables	x
Acronyms	xi
1 Introduction	1
1.1 Introduction	1
1.2 Justification	2
1.3 Research Aim	3
1.4 Thesis Outline	3
1.5 Contributions	4
2 Literature Review	6
2.1 Introduction	6
2.2 Digital Forensics	6
2.2.1 Defining digital forensics	7
2.2.1.1 Identification	8
2.2.1.2 Preservation	8
2.2.1.3 Analysis	8
2.2.1.4 Presentation	9
2.2.2 Areas of research within Digital Forensics	10
2.3 Web browser Forensics	10
2.4 Social Media Communication	11
2.4.1 Communication	11
2.4.2 Social media platforms as web applications	13
2.4.2.1 Case examples	13
2.4.3 Challenges posed by social media communication	14
2.5 Digital Investigations	15
2.5.1 Defining Digital Investigations	15
2.5.2 Digital Investigation models	17
2.5.2.1 Linear models	17
2.5.2.2 Non-linear models	19
2.5.3 Principles of digital investigations	21
2.6 Digital Forensics Triage	23
2.6.1 Existing triage methods	23
2.6.2 Post-mortem triage	24
2.6.3 Live triage	25
2.6.4 Extracting features from social media artefacts	27
2.6.5 Summary	28

CONTENTS

2.7	Digital Evidence	29
2.7.1	Rules of evidence	31
2.7.2	Digital evidence: Challenges	32
2.7.3	Reliability of digital evidence	33
2.8	Working definitions	33
2.8.1	Key terms	34
2.9	Summary	35
3	Research Methodology	36
3.1	Introduction	36
3.2	Problem Definition	36
3.2.1	Challenges	37
3.2.2	Limitations	38
3.3	Research Question	39
3.4	Objectives	40
3.5	Considerations	42
3.5.1	Experimental Constraints	42
3.5.2	Evidential Constraints	43
3.5.3	Legal Constraints	44
3.5.4	Procedural Constraints	45
3.5.5	Reliability Constraints	45
3.5.6	Ethical Constraints	46
3.6	Tools	47
3.6.1	Experimental Tools	48
3.6.2	Analysis Tools	49
3.7	Social Media Selection	53
3.7.1	Facebook	54
3.7.2	Google Drive	54
3.7.3	Twitter	54
3.8	Social Media: Terms of Acceptable Use	55
3.8.1	Facebook Community Standards	55
3.8.2	Google “The Terms”	57
3.8.3	The Twitter Rules	59
3.9	Environment Setup	61
3.9.1	Virtualisation Vs Physical Devices	61
3.9.2	Baseline Setup	63
3.10	Experimental Methodology	64
3.10.1	Traditional digital forensics solution	64
3.10.2	Framework conceptualisation	70
3.11	Summary	71
4	Firefox: Social Media Artefacts	72
4.1	Introduction	72
4.2	Twitter	72
4.2.1	Experiments	72
4.2.1.1	Availability and Recoverability of Artefacts	75
4.2.2	Results	75

4.2.3	Summary	89
4.3	Facebook	89
4.3.1	Experiments	89
4.3.1.1	Availability and Recoverability of Artefacts	91
4.3.2	Results	91
4.3.3	Summary	98
4.4	Google Drive	98
4.4.1	Experiments	98
4.4.1.1	Availability and Recoverability of Artefacts	100
4.4.2	Results	100
4.4.3	Summary	107
4.5	Relationship to Objectives	107
4.6	Results Overview	108
4.7	Summary	109
5	Chrome: Social Media Artefacts	110
5.1	Introduction	110
5.2	Twitter	110
5.2.1	Experiments	110
5.2.1.1	Availability and Recoverability of Artefacts	111
5.2.2	Results	112
5.2.3	Summary	124
5.3	Facebook	124
5.3.1	Experiments	125
5.3.1.1	Availability and Recoverability of Artefacts	126
5.3.2	Results	126
5.3.3	Summary	132
5.4	Google Drive	132
5.4.1	Experiments	132
5.4.1.1	Availability and Recoverability of Artefacts	133
5.4.2	Results	134
5.4.3	Summary	141
5.5	Relationship to Objectives	141
5.6	Results Overview	142
5.7	Summary	143
6	SMURF: Conceptualisation and Testing	144
6.1	Introduction	144
6.2	Framework Problem Definition	144
6.3	Developing SMURF	146
6.3.1	Characteristics of the Framework	146
6.3.2	Purpose of Framework	148
6.3.2.1	Individual Concerns	148
6.3.2.2	Evidential Concerns	148
6.3.2.3	Non-dependency of the Framework	149
6.3.2.4	Defining Relationships	149
6.3.3	Framework Development	150

CONTENTS

6.3.3.1	Framework Components	150
6.3.3.2	Development Environment: Set up	152
6.3.4	Building the Framework PoC	152
6.3.4.1	device_identifier	153
6.3.4.2	predefined_paths	155
6.3.4.3	app_config	155
6.3.4.4	system_info	157
6.3.4.5	session_info	161
6.3.4.6	browser_history	163
6.3.4.7	browser_cache	164
6.3.5	Testing the core function of the framework	165
6.3.6	Reporting the results	166
6.4	SMURF: How it works	168
6.5	SMURF PoC Implementation	170
6.5.1	Justification for Implementation	170
6.5.2	Method of deployment	171
6.5.3	Framework Testing	171
6.5.3.1	Registration/Authentication	173
6.5.3.2	Search and View Content	174
6.5.3.3	Share Content	176
6.5.3.4	Association/Relationship	177
6.5.3.5	Document Creation	178
6.5.3.6	Document Update	179
6.5.3.7	Document Collaboration	180
6.5.3.8	Generalisability of SMURF	182
6.6	Summary	182
7	Discussion	184
7.1	Introduction	184
7.2	Addressing key research problems	184
7.3	Research objectives	185
7.3.1	Characterise communications content	185
7.3.2	Categorise artefacts generated from the use of social media	187
7.3.3	Develop a framework for the recovery of social media artefacts	188
7.3.4	Assess the reliability of social media artefacts	190
7.3.5	Determine the relationships between communicating entities	192
7.4	Research Question	192
7.4.1	absence of the contents of user communication	193
7.4.2	social media artefacts	193
7.4.3	reliably attribute actions	194
7.4.4	link other entities involved	194
7.5	Conclusion	195
8	Conclusions and future research	196
8.1	Research Conclusions	196
8.2	Contributions	196

8.2.1	Established a method for the categorisation of social media artefacts	197
8.2.2	Criteria for assessing the reliability of social media artefacts	197
8.2.3	Developed a framework for social media investigations	198
8.3	Future Work	198
8.4	Summary	199
References		201
Appendices		217
Appendix A SMURF Example Output		218
Appendix B Context Artefacts		238
Appendix C Terms of Acceptable Use		245
Appendix D Peer Reviewed Publications		266
Appendix E Larger Representation of Modules		369
Appendix F Ethics Approval		374

List of Figures

2.1	U.S. Department of Justice model	17
2.2	DFRWS's model	18
2.3	Reith et al.'s model	18
2.4	Carrier's model	19
2.5	Casey's model	20
3.1	Social Media Stats	53
3.2	Virtual Machine Layout	62
3.3	Example illustration of Registry operations create, open, query, enumerate key activity (based on data from Procmon).	68
4.1	Contents of the Firefox History Profile directory	76
4.2	Extracts from <i>moz_pages_w_icons</i> indicating user activity on Twitter	78
4.3	Extracts from <i>moz_formhistory</i> user details (credentials) for Twitter	79
4.4	Tables in the <i>places.sqlite</i> database	79
4.5	Extract from <i>webappsstore.sqlite</i> shows account information for user 'Obed'	81
4.6	Extract from <i>sessionstore</i> infers an action was taken to view the users following Josh	82
4.7	Tweet content recovered from <i>sessionstore</i>	83
4.8	Twitter profile image and banner for test user Josh as seen on Twitter	85
4.9	Twitter profile image for test user Josh as captured by Fiddler	86
4.10	Twitter profile banner for test user Josh as captured by Fiddler . . .	86
4.11	Twitter profile image and banner for test user Obed as seen on Twitter	86
4.12	Twitter profile image for test user Obed as captured by Fiddler . . .	87
4.13	Twitter profile banner for test user Obed as captured by Fiddler . . .	87
4.14	Shared network folder (mapped as a network drive) added for uploads and downloads	92
4.15	Facebook cookie data set for user in Firefox	93
4.16	<i>Favicon</i> artefacts showing Facebook account activities	93
4.17	Extracts from <i>moz_formhistory</i> showing user name and associated email addresses	94
4.18	Extracts from <i>moz_formhistory</i> showing the mobile phone number used for logging in to Facebook	94
4.19	Extracts showing Facebook history activity, sorted by visit date . . .	95
4.20	Extracts showing Facebook searches and page visit	96
5.1	Chrome User Profile (History) contents	113
5.2	Tables in the Chrome History database	115
5.3	Network Action Predictor database	117
5.4	Example of Chrome's omnibox functionality	118
5.5	Web Data database	119

LIST OF FIGURES

5.6	Google account verification with a mobile number	137
6.1	An illustration of the weighting scale	146
6.2	A two-part illustration of SMURF components and data flow	160
6.3	An illustration of how SMURF works	168
7.1	Graphical representation of the framework method	189
A.1	SMURF - Example HTML report (main view)	231
A.2	Deployment device, system information	231
A.3	Firefox - Example output	233
A.4	Google Chrome - Example output	234
A.5	Windows Registry - Example Output	236
A.6	Event logs - Example output	237
A.7	Prefetch - Example output	237
B.1	Device driver installation when a USB is plugged in	240
B.2	Verification of Firefox Cache entry filename	244
E.1	app_config	369
E.2	browser ₁ _cache	369
E.3	browser _n _cache	370
E.4	browser ₁ _hist	370
E.5	device_id	371
E.6	predefined_paths	371
E.7	session_info	372
E.8	smurf_main	372
E.9	system_info ¹	373
E.10	system_info ⁿ	373

List of Tables

3.1	Ethical considerations and compliance	47
3.2	Tools Summary - Data examination and acquisition tools (in alphabetical order)	52
4.1	Font conventions	72
4.2	Results overview - user activities and artefacts (Firefox)	108
5.1	Font conventions	110
5.2	Extract from the <i>omni_box_shortcuts</i> table	119
5.3	Extract from the <i>autofill</i> table	120
5.4	Extract from the Cookies database file	127
5.5	Extract from the <i>logins</i> table	129
5.6	Results overview - user activities and artefacts (Chrome)	142
6.1	Font conventions	144
6.2	Group of artefacts related to user registration or authentication . . .	173
6.3	Group of artefacts related to user search and view activity	175
6.4	Group of artefacts related to user content sharing	176
6.5	Group of artefacts inferring user associations/relationships	178
6.6	Group of artefacts related to user document creation	179
6.7	Group of artefacts related to user document update	180
6.8	Group of artefacts related to user document collaboration	181
7.1	Relationship between the research objectives, research question, and key research problems addressed	185
A.1	Examples of artefacts recovered by the PoC	219
B.1	Example: Event IDs of interest	241

Acronyms

ACPO	Association of Chief Police Officers
CFFTPM	Computer Forensic Field Triage Process Model
CLI	Command Line Interface
CMC	Computer-Mediated Communication
DFT	Digital Field Triage Model
DTDFPM	Dual-Triage Digital Forensic Process Model
FAQs	Frequently Asked Questions
FIFO	First In First Out
GIFCT	Global Internet Forum to Counter Terrorism
GUI	Graphical User Interface
HTTP	HyperText Transfer Protocol
IDE	Integrated Development Environment
IoT	Internet of Things
IP	Intellectual Property
NPCC	National Police Chiefs' Council
OS	Operating System
PoC	Proof of Concept
RegEx	Regular Expression
SMURF	Social Media User Relationship Framework
SOPs	Standard Operating Procedures
UI	User Interface
VSC	Volume Shadow Copies

1 Introduction

1.1 Introduction

Social media has permeated many aspects of society; it presents a means for Internet enabled, interactive real-time communication amongst users, providing the ability to share content such as photos, videos, data files and messages [Manning, 2014], thus extending the way in which users interact and form social relationships with like minded individuals. Communication through social media platforms can be achieved through a web browser or a dedicated application. These dedicated applications can also be described as web applications.

A web application (web app) is a client-server application that runs through a web browser [Shklar and Rosen, 2009]; for example, social media platforms. Although social media presents benefits to individuals and businesses, there is the potential for it to be used in a range of unlawful activities such as trolling [Agency, 2015]; this has resulted in the service providers (e.g. Google, Facebook, Twitter) implementing measures [Humphries, 2010] to police the abuse of their platforms, by allowing users to report the offending content.

The format for artefacts from social media activity has no standard implementation however, it has been observed that standardisation can be found at the layer which the communication occurs e.g. through the URL syntax specification. Oftentimes, during time sensitive investigations, there may be a requirement to recover data from a live environment [Wiles and Reyes, 2007] otherwise known as a live triage.

This may include data from social media activity which may be key to potentially generating investigative leads. Triage as a practice originated in the medical field and was used to assess hospital patients in order to determine how and when to prioritise treatment [Robertson-Steel, 2006]; in digital forensics, triage is a growing practice with two streams; live and post-mortem triage. Post-mortem triage deals with device or digital evidence prioritisation post-seizure while live triage deals with evidence acquisition from a live device at a crime scene.

When a user initiates communication on social media through a web browser, it creates traces [Oh et al., 2011] and the data from such activity when analysed can provide valuable information to forensic investigators. It is important to note that

1.2. JUSTIFICATION

this data may not always be immediately or easily accessible on disk thus, it is important to evaluate the feasibility of using alternative (related) data to infer user social media activity.

1.2 Justification

Data from social media has been used successfully during the prosecution and conviction of offenders; for example, where a fake profile was created and used to lure and murder an unsuspecting victim [Haroon and Carter, 2010]. It is important when social media artefacts are being used during an investigation, to interpret the evidence in context and to corroborate with additional pieces of evidence. For example, in *COMMONWEALTH v. MANGEL* [Pennsylvania Superior Court, 2017], the court determined that circumstantial evidence such as metadata (e.g. user account/profile ID) was required for user attribution.

In some cases, one or more of the user computers involved can be analysed to get information about an event or incident as seen for example in the Birmingham Riots investigation in 2011 [McCarthy, 2011], where information gathered was used to identify other persons involved. Social media data may also be collected through third-parties who volunteer the information; in *U.S. V. MEREGILDO* [Southern District of New York (S.D.N.Y), 2012], a friend of the defendant who was described as a “cooperating witness” for the prosecution provided access to the defendant’s Facebook profile. Thus, the defendant’s profile could be viewed from the friend’s profile. This produced additional information about the defendant’s activities and the court decided that the defendant’s “reasonable expectation of privacy ended when social media posts and information were disseminated to the public”.

In a live triage, when recovering digital evidence that can be used in relationship attribution, a framework may be required as there are lots of variables such as cached background data to consider. It is important to note that conducting a live triage does not remove the requirement for a detailed forensic analysis [Cantrell and Dampier, 2012], however it may be used for intelligence gathering and in corroborating other evidence collected; as the data recovered through the live triage, on its own may not be admissible in court.

The same principles defined by ACPO [2012] is applicable to live triage to ensure that evidential integrity and reliability are maintained. The potential importance of social media artefacts as a source of intelligence, and their prompt recovery during a live triage highlights the need for more research in this domain.

1.3 Research Aim

The recovery of evidence pertaining to user activity on social media is valuable to digital investigations. In order to make good use of such evidence, it is important to understand the artefacts of interest. It is equally important to determine what constitutes communication content, and how related artefacts may be used to infer user activity. This is based on the premise that the content of user communication is unavailable/inaccessible and there is a need for quick actionable intelligence at the crime scene.

This research aims to substantiate whether artefacts other than the content of user communication on social media can be used to attribute actions or relationships to a user during a live triage investigation.

The approach taken to meet the research aim is to describe what constitutes social media activity and communications content [**Chapter 3**]; provide an understanding of what the artefacts of interest are, where to find them and how they can be recovered [**Chapters 4 and 5**]; and develop a framework for the recovery and contextualisation of these artefacts in a live triage scenario [**Chapter 6**].

1.4 Thesis Outline

The structure of the remainder of this thesis is as follows:

Chapter 2 critically reviews related work in digital forensics, digital investigations and digital triage; and establishes working definitions for these key domains. It reviews the rules of evidence in digital forensics and the importance of the reliability of digital evidence.

Chapter 3 identifies key problems to be considered as part of this research and defines the research problem. This is followed by the proposal of a research question, and the definition of the research objectives. The research constraints are discussed, followed by a discussion of the methodology and tools used for the research experiments and analysis.

Chapter 4 discusses the experimental methodology and the results of the manual analysis of social media activity using Firefox web browser. This chapter highlights artefacts of interest, their location, and evidential value. This is followed by the categorisation of potential user activity to be used as a baseline in the Proof of Concept (PoC) framework.

1.5. CONTRIBUTIONS

Chapter 5 discusses the experimental methodology and the results of the manual analysis of social media activity using Google Chrome web browser. This chapter highlights artefacts of interest, their location, and evidential value. This is followed by the categorisation of potential user activity to be used as a baseline in the PoC framework.

Chapter 6 discusses the framework conceptualisation and reviews the risks and benefits of conducting a live triage investigation. This chapter discusses the theoretical build process for a PoC framework; followed by a justification of the framework testing, and implementation through a PoC written in Python, using the data recovered in **Chapters 4 and 5**; finally, the generalisability of the framework is evaluated.

Chapter 7 evaluates the overall and individual methodologies employed in this research; followed by a critical analysis of how the research objectives were achieved, highlighting the strengths, weaknesses and limitations. Finally, the research question is broken down into key parts and analysed.

Chapter 8 presents a summary of the research; the contributions that this research makes to the digital forensics community, law enforcement and researchers. It also discusses potential future work stemming from this research.

1.5 Contributions

This research establishes a method for the categorisation of social media artefacts based on perceived user activity. It highlights the importance of social media artefacts to live triage investigations and digital forensics in general. The categorised data can be used to prioritise the analysis of the live triage data and potentially point to areas that may need further attention during a detailed forensic analysis.

This research characterises communication content and highlights evidential data of interest from user social media activity. This involves prior recovery of such artefacts using conventional forensic tools to understand where the evidential material of interest is stored and a suitable technique for its recovery. The output is used to define a baseline for the set of artefacts that are expected to be recovered by the PoC framework.

This research also proposes the criteria for assessing the reliability of social media artefacts in a live triage investigation by contextualising user activity to ensure that each artefact can be corroborated and linked to a specific event. Live triage

1.5. CONTRIBUTIONS

investigations involve interacting with a device that is powered on thus, providing concise documentation on the process can enable verification by independent third parties.

During the course of this research, existing triage techniques were reviewed as well as the risks and benefits of a live triage investigation. This facilitated the development of a novel framework for the recovery of social media artefacts in a live triage. Using the output from the categorisation of evidential data, a PoC was developed to test the applicability and generalisability of the framework.

2 Literature Review

2.1 Introduction

This chapter reviews literature related to the investigation of social media activity in a live triage and identifies potential research opportunities in digital forensics. In this chapter, **Section 2.2** defines the term digital forensics, including the areas of study (**Section 2.3**), challenges and advancements in this field; **Section 2.4** discusses social media communication. **Section 2.5** discusses digital investigations, reviews the existing models for digital investigations, proposes a research working model, and highlights the principles for digital investigations. **Section 2.6** discusses the importance of live triage to digital investigations. **Section 2.7** outlines and reviews the admissibility requirements for digital evidence, the value and reliability digital evidence. **Section 2.8** provides a definition of terms used in this research. Finally, a chapter summary is provided in **Section 2.9**.

2.2 Digital Forensics

The advances in technology have improved how users communicate, research items or areas of interest, and carry out personal or business transactions. It has also given rise to a variety of electronic crime [Crown Prosecution Service (CPS), n.d.] and the use of computer technology to facilitate traditional crimes like money laundering, fraud, kidnapping, child exploitation, terrorism, and other unlawful activities.

Digital forensics (also referred to as forensic computing or computer forensics), has been identified as an integral part of criminal investigations, civil litigations, and the investigation of IT security incidents [Goodison et al., 2015; Police Executive Research Forum (PERF), 2018]. According to the Digital Forensic Science Strategy [NPCC, 2020], about 90% of crime is recognised as having a digital element. This figure was arrived at following a series of oral evidence collected by the Science and Committee [2019]’s inquiry (extracts from the transcript can be seen below).

Examination of witnesses

“Q126

Lord Borwick: To follow up on Lord Renfrew, we have received evidence that about 70% of crimes have a digital element.

[...]

Professor Peter Sommer: Unless Mark knows differently, I do not think the statistics are being collected. To give you an example, I was instructed in a particularly gruesome murder case, and the digital evidence was that the accused said he was not there at the place of the murder, and he could prove it because there were computer logs on another computer in another location. It was a tiny part of the overall case. I do not think anybody would be collecting those statistics, but it is to illustrate that it is all pervasive. I do not know where the 70% figure comes from. It might be 90%.

Mark Stokes: In terms of investigations across all types of crime, we believe - and it is anecdotal I am afraid, and we do not have clear statistics - it is nearer to 90% of crime that has a digital element, in the broadest sense of that: CCTV, communications data, social media data, cyberattacks.

[...]

Mark Stokes: We are only involved in the ones that have digital evidence. I would conclude, even if there is a bit of attrition in there, that you are talking about a large number of criminal trials, which is all I can talk about, and obviously most fraud cases will be digital these days. Probably about 90% of murders and complex rape cases will have a digital element in them”.

- House of Lords Select Committee on Science and Technology [2018]

The term forensic computing is often restrictive and difficult to define [Hannan et al., 2003]; hence the reference by practitioners and researchers alike to digital forensics [Reith et al., 2002], as it broadens the scope of definition and covers the forensic investigation of all digital devices with the capacity to store or retain data.

2.2.1 Defining digital forensics

There have been various attempts by researchers and practitioners to define digital forensics; some of which have come from an investigative perspective, viewing it as the process of “*unearthing*” and analysing data of value from computer system [Mandia and Proise, 2001]; a scientific perspective viewing digital forensics as “...*the application of science and engineering to the legal problem of digital evidence*” [Pollitt, 1995]; and according to DFRWS [2001], the use of scientific methods to produce digital evidence from digital sources. This definition by DFRWS [2001] expands the scope of digital forensics to include criminal, corporate or military investigations.

2.2. DIGITAL FORENSICS

McKemmish [1999]’s definition approaches digital forensics from a mixed point of view; combining a scientific approach (applying computer technology) with an investigatory component (computer crime investigation). McKemmish [1999] identifies four “*key elements*” - identification, preservation, analysis and presentation; that ensures that the result of the investigation will be acceptable in court. These are discussed in **Sections 2.2.1.1 - 2.2.1.4.**

2.2.1.1 Identification

In digital forensics, finding the device(s) that may contain evidence relevant to a case is paramount; Casey [2011][p.189] describes this process a “*survey*” of evidence. This identification process occurs at the start of an investigation [McKemmish, 1999] and involves understanding the type of digital evidence that may be available. For example data files, web browser history, call logs from a mobile device; where and how the data is stored, and the recovery process.

2.2.1.2 Preservation

Preservation is a crucial step in any digital investigation; it involves protecting the evidence from modification or alteration. It is important that evidence submitted to the court is recovered in a manner that does not modify the state in which the data was discovered [McKemmish, 2008]. However, there may be circumstances where changes to the evidence are inevitable thus, in line with the Association of Chief Police Officers (ACPO) guidelines [ACPO, 2012], such changes should be minimal, well documented and justified.

2.2.1.3 Analysis

McKemmish [1999] describes this stage as the “*main element*” of digital forensics. Analysis involves processing the digital evidence by extraction, examination, and interpretation (into human readable form), and is often used to broadly describe the examination and analysis of digital evidence.

Casey [2011] however, makes a distinction between these two activities; arguing that the extraction of digital evidence and interpretation of information from the evidence prepares it for analysis where “*scientific methods*” and “*critical thinking*” can be used to explain how an incident occurred, who was involved, and the probative value of the evidence to the case.

It is important to make this distinction as extracting and viewing the information retrieved from the evidence alone may not be sufficient; it is necessary to substantially ‘connect the dots’ and highlight the relationships that exist between a suspect, the

digital evidence and an incident (incident in this context could be a crime, or corporate misconduct).

2.2.1.4 Presentation

This is the process of reporting or presenting the findings to a court or to a relevant body such as a disciplinary panel. Presentation includes providing detailed information about the forensic processes used in the production of the digital evidence [TWGECSI, 2001]. At this stage of an investigation, questions may be raised about the tools and processes to ensure that the evidence has been correctly interpreted; the competency of the digital investigator, and the verifiability of the evidence.

The definition proffered by McKemmish [1999] is deemed appropriate for this research as:

- ⇒ It highlights the processes involved in “*applying computer technology*” to computer crime investigation (using digital forensic tools to examine digital evidence).
- ⇒ It implies the importance of linking the recovered evidence to an offender (analyse, connect the dots).
- ⇒ It recognises the importance of maintaining evidential integrity and concise documentation (preservation).
- ⇒ It implies the importance producing correctly interpreted and relevant evidence in court (presentation).

Summary

Digital forensics is multidisciplinary, thus, it is important for investigators to be familiar with the devices and technologies they are investigating [DFRWS, 2001]. The core function of digital forensics identified from the above definition is the need to produce court-worthy digital evidence. This could be referred to as “*forensically sound*” evidence and implies that the necessary precaution was taken in the production of the said evidence [McKemmish, 2008]; this minimises the chances of producing incomplete, irrelevant or questionable evidence.

“Forensically sound” is a term used extensively in the digital forensics community to qualify and, in some cases, justify the use of a particular forensic technology or methodology [McKemmish, 2008]. For instance, ensuring that a disk image is an exact replica of the original i.e. no alterations or modifications (unintentional or otherwise) has been made to the image and that a comparison can be made for verification purposes, using their hash values (an irreversible one way encryption algorithm that converts a data block into a unique fixed length bit string; any

2.3. WEB BROWSER FORENSICS

subsequent change to that data block returns a different hash value). This ensures that digital evidence is collected in a manner free from bias [Casey, 2004] and that evidential integrity is maintained.

2.2.2 Areas of research within Digital Forensics

Digital forensics has been described as multidisciplinary and has a number of sub-disciplines [Casey, 2011] such as OS forensics, malware, mobile devices, Internet of Things (IoT). Typically, the forensic investigation of digital devices can be achieved using a range of conventional digital forensics tools; and may involve the acquisition of images of the user devices (computers, mobile phones, tablets etc.), examination of the image files, analysis and presentation of findings [Reith et al., 2002] as shown in the existing digital forensics process models.

Different aspects of these user devices can be analysed depending on the case although there may be overlaps. For example, the examination and analysis of user generated files on a computer would involve researching the OS to understand what file system is in use [Carrier, 2005]; how user generated files are created, and how the data is stored within that file system. A sub-discipline of interest to this research is web browser forensics [Oh et al., 2011]. Web browsers can be found on a range of user devices and in a digital forensic investigation, they can provide a wealth of useful information about a user’s Internet activity. The evidential value of data recovered from web browsers is discussed in **Section 2.3**.

2.3 Web browser Forensics

Web browsers play a large part in enabling access to the Internet [Oh et al., 2011]. In **Section 1.1**, web browsers were described as essential to client-server interaction on the Internet where the client (web browser) communicates with the server using protocols such as HyperText Transfer Protocol (HTTP) over a TCP/IP network. A web browser can be defined as “*an application used to view, interact with and navigate web pages*” [Oxford Reference, 2016]; it provides a means for users to access and retrieve resources from the World Wide Web (www). Examples of popular web browsers include Google Chrome, Firefox, Safari, Opera, Chromium, Vivaldi, and Microsoft Edge browser. Although the data location and file names vary based on the browser and the OS [Magnet Forensics, 2017; Mozilla Support, n.d.], web browsers tend to store similar data of interest related to navigation history, form data (e.g. logins), cache, cookies, downloads, and session information files.

There are several aspects of web browser activity that are of interest in digital

forensics; these include basic activity such as web searches using search engines, accessing and navigating web pages, or using other web services. An investigator may be interested in the reconstruction of a user's browser activity [Shafqat, 2016] as traces are usually left on disk when web browsers are used. In addition to basic browser activity, other data of interest to investigators comes from user social media interactions.

Web browsers in spite of their benefits, present a number of challenges to digital investigators. Web browser installations are regularly updated/upgraded and sometimes these changes may affect the availability and type of artefacts that can be recovered [Google, a; Mozilla Support, n.d.]; examples of such changes include data formats, the use of multiple browser profiles, and the addition or removal of tables from SQLite database files. There is also the challenge of private/incognito browsing mode however, Shafqat [2016] demonstrates the possibility of recovering some of the artefacts from private browsing by dumping and analysing the RAM.

2.4 Social Media Communication

The emergence of social media improved the ease of communication between users. It has also provided a platform for corporate marketing as a lot of organisations now have a social media presence. It has created opportunities for social connectivity among individuals of like/similar interests, and the formation of personal or business relationships through services like sharing personal/group events and photos, instant messaging, micro blogging, and advertising.

Social media affords digital investigators a means of gathering evidence that can be used to tie a suspect or suspects to a crime and to each other (criminal enterprise relationship attribution). In order to understand social media communication in the context of this research, it is necessary to provide a definition of Internet enabled (web based) communication.

2.4.1 Communication

Dance [1967] suggests that communication can be broadly interpreted as the “elicitation of a response from another, without making a distinction between the interaction of animate and inanimate matter or between humans and animals”. Littlejohn [1992] goes further and explains that communication can be defined by looking at the conceptual components of the act, for example speech, symbols, interaction, and intention. Thus, communication in a generic sense can be described as *‘an activity that involves the exchange of information between two or more entities, through*

2.4. SOCIAL MEDIA COMMUNICATION

writing, speaking, or by using signs/symbols’.

When applied to web based communication, this exchange of information could take place at different levels - it could be the interaction between computing devices when a request is sent [Berners-Lee et al., 1996] (e.g. ‘fetch the requested service using port 80’); it could also be the human interaction such as sending and receiving messages; requesting and accessing web pages via a browser; posting, downloading or uploading items to and from a social media platform.

Communicating on the Internet with the aid of computers can be referred to as Computer-Mediated Communication (CMC) [Ellison and Childs, 1999] and it involves the conversion of data into electronic/digital formats that can be transmitted over the TCP/IP stack using the client-server interaction method. Data is broken down into packets and transmitted through various routes, and then reassembled when it gets to its destination [December, 1996].

The data transmitted could be in the form of messages, files, etc., all of which could contain text, audio, or video content; it could also be in the form of posting (or uploading) content that is retrospectively viewed (or downloaded) by other social media users. Certain types of communication could be considered illegal depending on the jurisdiction [Wall, 1999], this is because the definition of what constitutes a crime varies across jurisdictions and acceptable content, within reason, may be subject to different interpretations.

The Internet has been described by Armstrong and Forde [2003] as “*an open unregulated, largely unpoliced and unpoliceable interconnected communications infrastructure*”. It presents users with the ability to control the level and type of information they disclose. It also presents an opportunity for a fraction of users to engage in activities such as the use of false personas or identities; deception and other forms of disagreeable activities. According to Ellison and Childs [1999], some of the reasons why CMC differs from traditional communication methods is because it sometimes encourages users to act in ways they would not be able to repeat offline (i.e. in face-to-face interactions).

Defining communication in terms of criminal activity implies the motive behind a criminal act; the intent to commit a criminal act, or the intent to incite another Internet user into doing the same. Therefore, in the context of this research, communication can be defined as ‘*the exchange of information between two or more entities, using web based technologies as a medium for transmission*’. When applied to social

2.4. SOCIAL MEDIA COMMUNICATION

media, this definition covers all content (text, audio, video, image, etc.) exchanged between individuals on a social media platform.

Zeng et al. [2010] describes social media as a “*conversational, distributed mode of content generation, dissemination, and communication among communities*”. The use of social media artefacts as digital evidence is an on-going research area in digital forensics [Arshad et al., 2019] that needs to be explored in detail because of the wealth of useful information that can be recovered. There are several aspects of social media usage that have been and can be researched. For example, social media usage has been approached from a network forensics perspective [Shaw et al., 2016], reviewing the vulnerabilities of social media platforms and the challenges social media platforms pose to digital investigations.

Data from social media can be used in non-criminal investigations. Hubert [2014] proposes techniques that can be applied by corporate entities when investigating employee misconduct, especially considering that such investigations can not compel third party cooperation unlike in law enforcement.

2.4.2 Social media platforms as web applications

Shklar and Rosen [2009][p. 4] defined a web application (web app) as a “*client-server application that uses a web browser as its client program, and performs an interactive service by connecting with servers over the Internet (or Intranet)*”. Web apps allow website visitors to interactively submit and retrieve (dynamic) data to and from a server, over the Internet, using a web browser. This interaction is achieved through the use of HTTP [Scambray and Sima, 2011]. In the context of this research, social media platforms can be described as web apps as they perform this function.

A web app can thus be described as a *software program an entity uses as a medium to perform actions, which allows the interactive submission and retrieval of dynamic data through a web browser using HTTP*. A few examples include Facebook, Instagram, Twitter, Google Drive. These social media platforms use a combination of underlying technologies and frameworks such as AJAX, React.js [Facebook Inc., 2019] to manage the rendering of dynamic content. Social media platforms also offer standalone applications that can be downloaded and installed however, these are outside the scope of this research as these desktop (and mobile) applications do not access the social media platforms through a web browser.

2.4.2.1 Case examples

This section highlights some cases involving social media:

2.4. SOCIAL MEDIA COMMUNICATION

Trafficking of minors

A woman in Oklahoma, was arrested for trying to sell her two children for \$4000 on Facebook. Apparently, she needed some money to bail her boyfriend out of jail.
- Taylor [2013]

Criminal solicitation of murder

A teenager charged with rape tried to hire a hit man (for \$500) through Facebook to “knock off” his accuser. In the process, he inadvertently made contact with an undercover detective who eventually testified in court.
- Matyszczyk [2011]

2.4.3 Challenges posed by social media communication

As with every technology, social media also presents challenges to digital forensic investigators. Some of the challenges include the use of proprietary data formats across devices, the type of artefacts written to disk, and the ability of a small percentage of the user community to use the platform for reprehensible or unlawful activities [Bello and DiBlasio, 2013; Moore, 2014; Select Committee on Communications, 2014] as evidenced by the number of social media related prosecution and convictions over the years [Agency, 2015; Haroon and Carter, 2010; Wood, 2018] for trolling, murder and drug offences.

Investigating social media activity may also be required in cases involving corporate misconduct [Taylor et al., 2014] for example, theft of Intellectual Property (IP), libel and disinformation on social media. Investigators are also faced with the lack of defined guidelines or standards [Cusack and Son, 2012]; this often results in such investigations being conducted in the same manner as a generic web browser focused investigation. To address the lack of standardisation and guidelines, Taylor et al. [2014] suggests the use of ACPO Guidelines (in the UK) as a starting point when investigating social media activity. This suggestion appears to give the investigator the flexibility to adapt the investigation to suit the circumstances surrounding the case.

Another challenge associated with the forensic investigation of social media activity is that the communication content may not always be immediately available or accessible on disk; further measures may be required to access or retrieve it. There are also possible privacy implications [Casey, 2013] because artefacts recovered may contain information about other unrelated users/activities. Hence, the need to

contextualise the evidence to understand its relevance to the case.

2.5 Digital Investigations

In order to gain an understanding of the digital investigations process, the definitions of digital forensics were reviewed in **Section 2.2**. This section reviews the digital investigations process and outlines steps to be followed by forensic investigators as well as certain guidelines that must be adhered to in order to maintain the reliability of the investigation process and evidence integrity.

2.5.1 Defining Digital Investigations

Investigation can be defined as *“the act or process of examining a crime, problem, statement, etc. carefully, especially to discover the truth”* [Cambridge Dictionary, 2019]. The decision to conduct a digital investigation depends on the primary objective of the organisation [Schatz, 2007] for instance, business continuity may be the primary objective of a corporate entity investigating an intrusion on their networks while for law enforcement the primary objective is usually the prosecution of offenders.

Digital investigation has been defined by Carrier [2006] as *“a process that formulates and tests hypotheses to answer questions about digital events or the state of digital data”*. It is important to note that there are several processes involved in determining the “state of digital data” and proposing a probable answer about digital events that have occurred. For instance, the data has to be observed, then a hypothesis is formulated and tested [Carrier, 2006] to corroborate the observed events.

Digital investigations are often after the fact however, there may be instances where the only option immediately available to an investigator is the acquisition of data from a live device [Wiles and Reyes, 2007]; such scenarios include running applications from portable, mounted or network devices, and the use of encryption. It is important to note that prior to the hypothesis testing stage, the data must be examined to determine a sequence of events or a pattern.

Carrier [2006]’s definition therefore, can be further extended to include *“the extraction and interpretation of meaningful data that relates to an event either to prove or disprove a fact”*; meaning that the event has to be clearly understood in order to formulate and test a hypothesis. This data must be obtained from digital sources [DFRWS, 2001] and the result is the digital evidence that is used in criminal or civil proceedings.

2.5. DIGITAL INVESTIGATIONS

Casey [2004] describes the goal of investigations as “*uncovering and presenting the truth*”. It answers the question of person(s) involved, intent/motivation, nature of the event, how the event occurred, the location of digital artefacts, and time frame of an event. Thus, trying to provide the reasonable answers to the fundamental questions of Who, What, When, Where, Why, and How of an investigation. There is the possibility that these questions may not be fully addressed by the digital evidence recovered and may require corroboration from other sources such as witness statements or physical evidence.

For clarity and to further narrow the scope of this research, digital investigations would be defined within the context of producing forensically sound, court-worthy evidence. Digital investigation is therefore defined as ‘*the process of recovering, preserving, and processing digital artefacts in order to produce evidence that meets the requirements for admissibility in criminal proceedings*’. This definition is deemed suitable as it includes the key elements of digital forensics identified by McKemmish [1999], and focuses on investigations where the objective is the production of evidential material that meets the prescribed court standards.

In general, a ‘live’ and/or ‘post-mortem’ (after-the-fact) investigation may be required depending on the severity of the case. A live investigation is necessary where time is of the essence and some actionable intelligence is required [Montasari, 2016] in order to proceed with either an interview with the suspect or to refine the scope of an investigation. In such scenarios, the evidence is acquired from an active (powered on) device, following ACPO Guidelines to ensure that potential risks to the state of the digital evidence are documented and justified. A post-mortem investigation involves the examination and analysis of an image of a digital device in a lab environment.

The interaction between a user and a digital device is a bi-directional process that can be explicitly described by Locard’s Exchange Principle [Chisum and Turvey, 2000] which states that: “*Any action of an individual, and obviously, the violent action constituting a crime, cannot occur without leaving a mark...*” (Locard 1934, pp. 7-8 as cited in [Chisum and Turvey, 2007][pp. 23-24]). In a physical forensics crime scene investigation, which is based on Locard’s Exchange Principle [Lee et al., 2001], an example could be an offender leaving tyre tread impressions and collecting traces of soil that could be used to connect the offender to a crime scene.

In the digital realm, and within the confines of crimes involving the use of a digital

device (which may or may not involve physical violence), this principle infers that user actions such as powering on a device, using the device and powering it off would leave a trace (artefacts). Traces left could be in the form of generic system event logs, application installation and usage logs, or user browser history found on the device. Such traces can subsequently be used in profiling and identifying the user(s) during an investigation.

It is important to consider the fact that certain factors (human or otherwise) may contribute to the reported ‘current state’ of a target digital device. This is known within the physical forensic sciences as *evidence dynamics* and has been defined by Chisum and Turvey [2000] as “*any influence that changes, relocates, obscures or obliterates evidence regardless of intent*”. In digital forensics, evidence dynamics can be used to describe the changes that could occur any time between the creation and transfer of evidence up to the point where the case is concluded and the evidence is destroyed [Brown, 2009; Casey, 2004, 2010]; for example, poor storage conditions for digital evidence.

2.5.2 Digital Investigation models

Several investigation models have been proposed to harmonise the various phases involved in digital investigations. This is because different scenarios require different processes although there could be overlaps [Carrier, 2006]; for example, a criminal investigation would have processes like identification, collection, preservation, examination, analysis, and presentation which could also be applied to a corporate or military investigation. A few of these models are briefly discussed in this section.

Based on the proposed working definition of digital investigation [Section 2.5.1], and the fact that certain phases of an investigation could apply to criminal and non-criminal scenarios, it is important to review the existing models of digital investigations and identify processes and phases that are key to the successful completion of a digital investigation.

2.5.2.1 Linear models

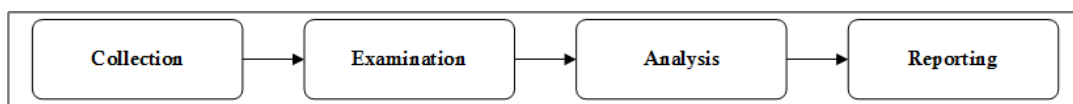


Figure 2.1: U.S. Department of Justice model

Figure 2.1 shows the process model proposed by the US Department of Justice Technical Working Group for Electronic Crime Scene Investigation (TWGECSI) as

2.5. DIGITAL INVESTIGATIONS

a guide for first responders [Mukasey et al., 2008; TWGECSI, 2001]; it represents a four-step linear digital investigation process. The collection stage involves searching for, recognising, collecting, and documenting digital evidence. The examination stage explains the origin and significance of the evidence while the analysis stage determines the relevance of the evidence to the case, and its reliability.

TWGECSI [2001][p. 3] makes a distinction between the examination and analysis stages; it suggests that the examination is a technical review done by the digital forensics practitioners while the analysis rests on the digital investigators; although in some scenarios, both stages may be conducted by the same person or group. The reporting stage is the final stage of the digital investigation process and involves the presentation of the evidence in court. According to TWGECSI [2001][p. 3], the reporting stage may involve expert testimony from the investigator where questions about the methods employed during the investigation and the expertise of the examiner/analyst or investigator may be answered.

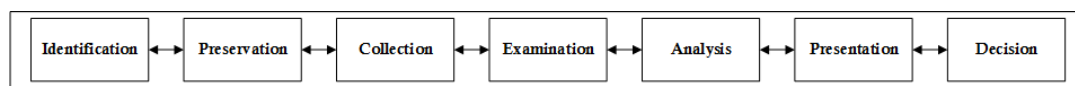


Figure 2.2: DFRWS’s model

The DFRWS [2001] model is a linear process matrix where each stage is described as a class with associated “*candidate techniques or methods*” with room for feedback from preceding classes. For example, the identification class consists of event/crime detection, system monitoring; followed by the preservation stage, with case management, and ensuring evidential integrity and continuity through a chain of custody.

The collection stage involves the use of approved hardware and software for evidence acquisition; examination involves the use of validation, filtering and pattern matching techniques for the processing of digital evidence. Analysis involves using the evidence to establish links to a suspect. The presentation stage is where the evidence is interpreted and may involve expert testimony.

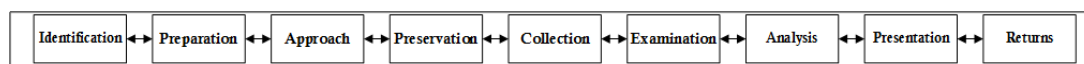


Figure 2.3: Reith et al.’s model

Reith et al. [2002]’s model is a generic enhancement of DFRWS [2001]. It makes reference to an approach strategy after the identification and preparation stages with the intent to “*maximize the collection of untainted evidence while minimizing impact to the victim*”. This involves considering the type of technology in place, for example,

determining whether it is necessary to shut down a device or if a live acquisition of evidence would be more appropriate.

With regards to an approach strategy, Carrier [2003]’s suggests that an approach strategy should precede the investigation process. This model considers the implications of collecting evidence from networked computers and the possibility of returning the source device to the owner at the conclusion of the case. Reith et al. [2002] developed this model according to the traditional (physical) forensic investigation strategy practised by law enforcement.

2.5.2.2 Non-linear models

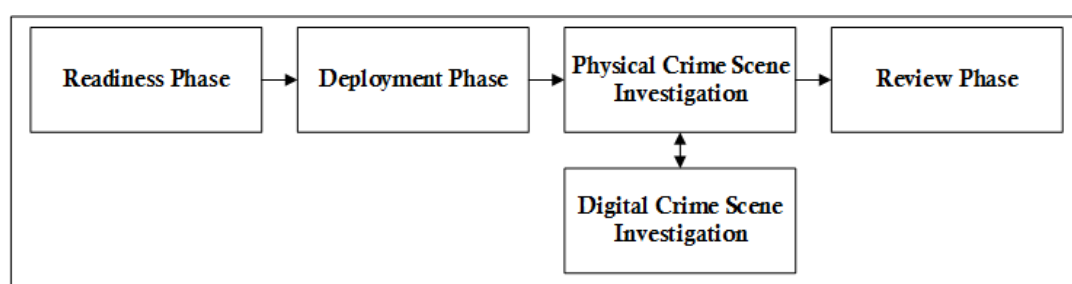


Figure 2.4: Carrier’s model

Carrier and Spafford [2003] identified certain requirements that digital investigations have to meet; suggesting that digital investigation models must be generic, practical, flexible, applicable to criminal and non-criminal investigations, and based on long existing and established physical forensic investigation models. These requirements were intended to cover corporate investigations, incident response, and investigations by law enforcement.

Based on these requirements, and the ‘high-level’ physical crime scene investigation processes i.e. securing the scene, identifying sources of evidence, documenting the scene, searching the crime scene for additional evidence and crime scene reconstruction, Carrier and Spafford [2003] developed a 17 phase model consolidated into five specific groups as shown in Figure 2.4. This model is referred to as the Integrated Digital Investigation Process (IDIP) and in it, Carrier and Spafford [2003] provides a detailed explanation and makes a distinction between the physical crime scene and its resulting physical evidence; and the digital crime scene and its resulting digital evidence.

According to Carrier and Spafford [2003], the physical crime scene is the physical environment (e.g. home, office, car) where the physical evidence of crime (e.g. computer, hard disk, USB) can be found and the digital crime scene is the virtual environment

2.5. DIGITAL INVESTIGATIONS

created by the computer's Operating System (OS), software and hardware, where the digital evidence (digital data) exists. Isolating and securing the physical and digital crime scenes may involve, cordoning off the premises and preventing access to the crime scene, and using a faraday bag/cage to prevent remote connections to a digital device. Isolating the crime scene would prevent further access by any party other than the investigating team, to the evidential media.

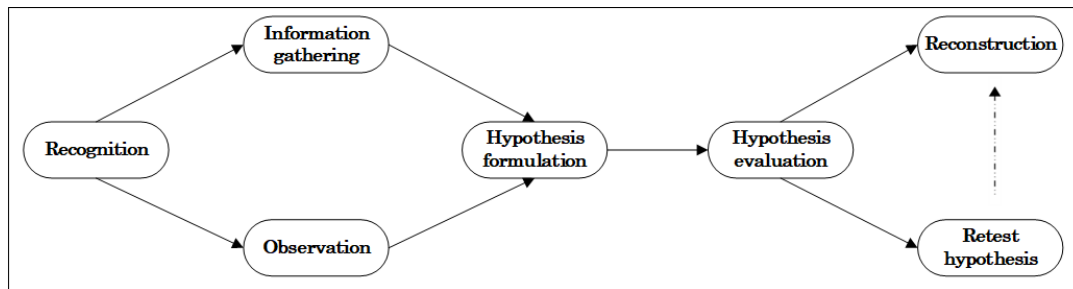


Figure 2.5: Casey's model

Casey [2010] advocates the application of scientific methods to digital investigations to ensure that the results are based on fact. Figure 2.5, is a representation of a digital investigation process proposed to achieve this. The recognition phase of this process as shown above involves notification of a crime, information gathering and observation. This could be further strengthened by interviewing witnesses or victims; identifying and filtering out irrelevant artefacts; recovering deleted data and hidden partitions; verifying evidence integrity and authenticity, and making use of other intelligence.

This model includes the hypothesis formulation and evaluation stages which provides the investigator with opportunities to work out a probable, factual explanation for the observed sequence of events. A hypothesis is formulated when there is intelligence or evidence, and the term probable is used prior to the testing of the hypothesis. According to Casey [2010], the success of a digital investigation is dependent on how rigorous the hypothesis testing is. Merely seeking to prove a theory increases the possibility of reaching the wrong conclusions thus, experiments should be aimed at disproving the hypothesis. This model gives room for feedback and a review of preceding stages if the results are inconclusive, or if any previously unidentified material evidence is discovered that may enable the revision and retesting of the hypothesis.

As there is no formally standardised digital investigations model, it is invariably up to a digital forensic investigator to decide what model to follow as long as the evidence is legally acceptable. Oftentimes, the chosen model may be an internal

model specified as part of an organisation's Standard Operating Procedures (SOPs) e.g. implementing ISO standards [International Organization for Standardization (ISO), 2012]. This is a good starting point because internal models although narrow and organisation centred, are starting to formalise but there is more freedom to adapt the processes compared to traditional forensic science disciplines.

A number of the models discussed above with the exception of Carrier and Spafford [2003] and Casey [2010]'s models, begin with an identification or collection phase however, the state of affairs prior to evidence identification or collection is not discussed in detail. Mandia and Prorise [2001] in their incident response model proposed certain phases like a pre-incident preparation and response strategy formulation. These two phases although initially designed for incident response and intrusion detection can be incorporated into a digital investigation model. This is because they address issues such as setting up adequately equipped digital forensics laboratories; competency of staff (training and certification for investigators) and an approach for live acquisition. This may include pre-preparation where the investigator considers the nature of the crime, how to proceed (e.g. employing live acquisition for encrypted systems or devices, and dealing with networked computers and servers).

2.5.3 Principles of digital investigations

In addition to following a structured digital investigations model, investigators are compelled to abide by certain principles that would ensure the admissibility of digital evidence in court. This section highlights the various principles that have been proposed over time to aid digital investigators [ACPO, 2012; IOCE, 2000; McKemmish, 1999; TWGECSI, 2001]; despite the differences in legislation and jurisdiction, the following similarities were identified:

1. **No data alteration or modification should occur:** Investigators are advised not to work with the original evidence if it can be avoided [ACPO, 2012]. To prevent tampering, modification or alteration of digital evidence, it is advisable to use a write blocker to create a working copy which is an exact replica of the original evidential device. It is also advisable to apply cryptographic hashing techniques and maintain a chain of custody. As long as the data although presented in a different format (e.g. conversion from machine readable to human readable) is not modified, it satisfies one of the requirements for admissibility.
2. **Keeping an audit trail:** In a situation where it is necessary to access the original device, for example, evidence acquisition from a live device,

2.5. DIGITAL INVESTIGATIONS

any changes made due to accessing the device must be minimal and the investigator must keep a log of all actions taken [ACPO, 2012; IOCE, 2000]; document any changes and effects (chain of custody). This would answer questions as to the reliability or validity of the investigative process or the evidence itself [McKemmish, 2008]; and ensures the clarity of the forensic report.

3. **The accuracy of the evidence:** Any evidence presented may be re-tested at the discretion of the court for clarification purposes [McKemmish, 2008]; as long as the digital investigator has satisfied the requirement of accuracy, the risk of reporting incorrectly or producing incomplete evidence in court is minimised.
4. **Ensure that the result of the analysis complies with the specified rules regarding the submission of evidence to court:** This principle answers questions such as the reliability and relevance of the evidence to the case; whether the evidence is authentic and valid, and whether the same results can be produced if a different tool or technique is applied [McKemmish, 1999]. It stresses the importance of the repeatability of the investigative process especially the examination and analysis of evidence. Industry acceptance for example, peer review of the forensic tool; documentation of the process and tool(s) used; identifying errors rates and inconsistencies (if any), are an important part of ensuring that independent third-party validation is achievable [McKemmish, 2008]. If this condition is duly satisfied, then the question of evidence admissibility is answered.
5. Anyone handling or processing digital evidence at any stage of the digital investigation process must be **competent either via qualifications or experience** [ACPO, 2012; IOCE, 2000]: If not trained to carry out a specific task, the investigator must refer to a suitably qualified colleague/investigator and document reasons why the task had been referred and what stage had been completed prior to the referral.
6. **Adherence to these guidelines:** This principle answers the question of where responsibility lies and compliance to the specified rules of evidence in that jurisdiction. For example, the IOCE [2000] and ACPO [2012] principle 4 states that the case officer has overall responsibility for ensuring compliance to the applicable principles. Tasks however, should be delegated to investigators who are qualified (through experience and/or certification) and every investigator should be responsible for their work and for their professional development. If this (along with the previously mentioned principles) is

satisfied, the evidence can then be said to be forensically sound and thus admissible.

This section (**2.5.3**) has identified and highlighted, based on existing work, the principles that guide digital investigators. These principles give a clearer understanding of evidence handling and processing ‘code of conduct’. These principles are not a standalone guide but need to be implemented at each stage of the forensic digital investigation. This section has also highlighted that some investigations require the acquisition from a live environment; this is discussed in **Section 2.6**.

2.6 Digital Forensics Triage

Previous sections in this chapter made references to data acquisition from live environments. This section discusses the use of triage in digital forensics, including the challenges investigators face with regards to obtaining evidence from a live machine; the opportunities it presents, and the evidential value of data recovered during a digital triage.

2.6.1 Existing triage methods

Several models for digital investigations have been successfully implemented within the digital forensics community (**Section 2.5.2**). However, these models are by design, largely aimed at a complete, ‘after the fact’ forensic investigation. It was observed that some digital investigations which may require a triage, do not have adequate representation in these models.

Digital triage processes and models have also been proposed within the digital forensics community. Cantrell and Dampier [2012] introduced a Partially-Automated, and Crime Specific Digital Triage Process Model which proposed the use of “pre-examination” or screening techniques to assist digital forensics examiners with the prioritisation of evidence prior to a full forensic analysis. Cantrell et al. [2012] is an implementation of the automated stages of the Partially-Automated, and Crime Specific Digital Triage Process Model and is aimed at enabling the quick profiling of computers during a triage.

Other models include a Tiered forensic methodology model for Digital Field Triage by non-digital evidence specialists [Hitchcock et al., 2016], designed to enable non-digital forensics specialists carry out triage investigations; a Formal Two Stage Triage Process Model (FTSTPM) for Digital Forensic Practice [Montasari, 2016], and the Computer Forensics Field Triage Process Model [Rogers et al., 2006]. These models are aimed at maximising efficiency by providing investigators with case-relevant

2.6. DIGITAL FORENSICS TRIAGE

artefacts in a timely manner.

Triage can be defined as *“the process of quickly examining patients who are taken to a hospital in order to decide which ones are the most seriously ill and must be treated first”* [Cambridge Dictionary, 2019] and has historically been in practice in the field of medicine [Robertson-Steel, 2006].

In the realm of digital forensics however, this definition can be used to describe two primary scenarios; a post-mortem (off-site) triage or live (on-site) triage, depending on the nature of the investigation:

- Post-mortem: the investigator/forensic analyst needs to make a decision on what devices or images need to be submitted to, or examined first in the forensic lab to assist with prioritising existing or incoming caseloads [Cantrell et al., 2012; Cantrell and Dampier, 2012].
- Live: the investigator/forensic analyst needs to make a determination on what case-relevant evidence can be retrieved quickly on-site (at the scene) from already identified and secured active devices (i.e. powered on) in order to guide investigators, interview suspects, generate new leads or to direct the search and seizure of other devices identified as related to the case.

These are briefly discussed in **Sections 2.6.2** and **2.6.3**.

2.6.2 Post-mortem triage

Post-mortem (PM) triage also known as off-site triage refers to lab-based triage activities. Parsonage [2009] proffers a process based definition which infers that triage can be used as an efficient way of ensuring that limited resources are adequately used to prioritise evidence based on “the need for, or likely benefits of an examination”. Roussev and Quates [2012] also propose a process based definition where data (in this case digital evidence) is screened in order to identify case-relevant artefacts and also “help build an overall understanding of the case”. These definitions can be used to describe activities in a digital forensic lab where digital devices (and recovered artefacts) may need to be categorised based on the potential availability of data of interest, or relevance to the case.

PM triage involves a quick, time and resource effective means of identifying and assessing devices containing case-relevant artefacts. For example, Garfinkel [2006] introduced Bulk Extractor as a viable technique for prioritising devices based on the case-relevance of the evidence it contains. This process may also lead to the identification of other items/persons of interest to an investigation. This activity is

conducted in a digital forensics lab after digital devices have been seized and imaged in accordance with the established guidelines or principles for the jurisdiction (e.g. ACPO Guidelines in the UK), but prior to a detailed examination/analysis of the image.

As at the time of writing, the current research on PM triage has leant towards modelling and automation of the activity. For example, Yang et al. [2016] describes a Dual-Triage Digital Forensic Process Model (DTDFPM) which covers both PM and live triage. The PM part of the model sorts through the evidence and prioritizes devices going by the probability that they contain evidence relevant to the case.

PM triage when implemented properly would help forensic labs manage their caseload and ensure that the examination and analysis of digital evidence is conducted in a time and resource efficient manner. Casey et al. [2009] argues that the lack of established guidelines for triage examinations introduces the risk that a forensic triage, which is focused on identifying and classifying “specific known artefacts” may result in other pertinent evidence being overlooked rendering the process inefficient. To address this, Casey et al. suggests the need for guidelines and updated tools to mitigate this risk.

This section provided a brief background of PM triage and its usefulness in digital forensic investigations. This was aimed as an introduction to the triage process and also as a way to highlight the differences between PM and Live (on-site) triage processes. As PM triage is outside the scope of this research, it will not be discussed further. For more information including a review of PM triage models, see: [Garfinkel, 2013; Jusas et al., 2017; Parsonage, 2009; Roussev and Quates, 2012].

2.6.3 Live triage

Live (on-site) triage has been briefly referenced in previous sections in this chapter. It is a technique that can be used by investigators to retrieve quick and actionable intelligence at a crime scene. The intelligence it provides can be used to direct an investigation before the digital devices are transferred to a lab for further processing.

A number of techniques have been proposed with regards to the use of a live triage in digital forensics investigations [Gielen and Bolzoni, 2014; Montasari, 2016]. These techniques identify the requirement for a live triage during digital forensic investigations as a means of finding intelligence that can be used to:

2.6. DIGITAL FORENSICS TRIAGE

- generate new leads
- assess the severity of a crime
- identify other persons or items of interest

Rogers et al. [2006] developed the Computer Forensic Field Triage Process Model (CFFTPM) which can be applied to a wide spectrum of investigations. The model is considered a foundation for modelling the live triage process. CFFTPM is focused on the on-site analysis of digital device(s) and aims to obtain information that can be used during the “search execution phase” of the investigation, or whilst a suspect is being interviewed. It highlights the need to consider live triage as part of the digital forensics process.

The CFFTPM is not automated, thus it can only be used by competent users. Whilst the timely retrieval of case-relevant information is important, the requirement for examiner competency with CFFTPM may result in additional resources being expended on-site. This view is expressed in Hitchcock et al. [2016] which highlights a number of problems this poses. For example, an analyst on-site would mean one less resource in the lab. This may inadvertently lead to an increase in the backlog of devices in the forensic lab because the on-site analysis becomes a priority and thus, existing cases may need to be re-prioritised.

Hitchcock et al. [2016] posits that training non-digital forensics specialist staff to supplement investigators/analysts on-site would help with the provision of information that can be acted on pending a full digital forensics analysis, and would help digital forensics units maximise the use of limited resources. The Digital Field Triage Model (DFT) is proposed by Hitchcock et al. [2016] as a way to introduce non-digital forensic specialist staff to the skills required for live triage, the front-line prioritisation of devices, and then advising on which devices an analyst needs to examine further. Having staff with basic live triage training would give them “tool-neutral” training and would ensure that analysts required in the lab are able to focus on lab-based work.

The challenge of appropriately managing limited resources and utilising non-digital forensics specialists for live triage activities was also addressed by the introduction and implementation of semi-automated steps in the triage process model proposed by Cantrell et al. [2012]; Cantrell and Dampier [2012]. It was argued that the introduction of semi-automation would improve ease of use by non-digital forensics specialists and would speed up the live triage process. This also makes it possible for these processes to be scripted into a case-specific framework.

Live triage is conducted on an active device such as personal computers, servers, mobile devices, etc. In the context of this research, a live triage may be carried out at the crime scene when an investigator makes a determination that the need for quick actionable intelligence outweighs the need to perform a lab-based, detailed forensic analysis on all potential sources of digital evidence discovered at the scene [Wiles and Reyes, 2007]. Performing a live triage could be for a number of reasons such as the time critical nature of the case e.g. kidnapping or missing person(s); imminent threat to life; the practicality of seizing a device; the use of encryption on the device, or the extraction and preservation of volatile artefacts [Bashir and Khan, 2013].

The use of live triage has raised some concerns with regards to the possibility of data being written to the device being investigated. When employed during an incident response investigation where the objective is usually business continuity and making sure that compromised systems are returned to a ‘good working state’, creating files or writing to the target device may not be seen as problematic [Gielen and Bolzoni, 2014]. However, during digital forensic investigations where the state of digital data is expected to remain unmodified, all actions taken during a live triage including their potential impact must be documented, and justified, in adherence with the relevant guidelines for handling digital evidence. There are also concerns about potentially overlooking pertinent evidence [Casey, 2013]; however, this concern can be addressed by conducting a full forensic analysis afterwards.

Although there is on-going research in the use of and applicability of live triage processes in digital forensics, additional work is required to capture and address how triage tools can be used to supplement the limitations of existing tools and processes, and to manage targeted capture of specific volatile or case-based artefacts that can provide investigators with actionable intelligence.

2.6.4 Extracting features from social media artefacts

Digital investigators often face a challenge when it comes to the quick identification, retrieval and contextualisation of meaningful, case-relevant information from digital devices in time critical investigations, and prior to a detailed analysis of a seized device. This is due to the proliferation of devices and the volume of data investigators need to process. In this context, and in a live triage, feature extraction can be used as a viable approach to process the social media artefacts at the crime scene and quickly retrieve content of interest.

2.6. DIGITAL FORENSICS TRIAGE

Features have been defined by Garfinkel [2006] as “*pseudo-unique identifiers*”, which are used as the basis for analysing individual hard drives and correlating digital evidence from multiple hard drives. These pseudo-unique identifiers as described by Garfinkel have the following properties:

- i. “sufficient entropy such that within a given corpus it is highly unlikely that the identifier will be repeated by chance”. Repetitions may occur however, when it does, it is attributable to a specific action such as copying a file from one device to another
- ii. recognisable using pattern matching techniques e.g. Regular Expression (RegEx)
- iii. do not require parsing or semantic analysis
- iv. do not change over time
- v. can be correlated with specific documents, people or organisations

In the context of this research, the re-occurrence of a given feature infers a user’s repeated access to a particular web resource such as a social media profile, shared documents etc. (See [Garfinkel, 2006, 2013] for more on feature extraction).

As discussed in **Section 2.5.1**, “every contact leaves a trace”. Features can be extracted from such traces (e.g. from the absolute path of a URL, data from the cache, session data files) and can be used to infer what a user has done. In the context of this research, feature identification and extraction will be defined as *an intelligence gathering process through which artefacts containing key information about a user’s activities are identified, and retrieved using pattern matching methods such as RegEx, and can be corroborated by other pieces of evidence on the digital device.*

2.6.5 Summary

This section has discussed Digital Forensics triage and made distinctions between PM and Live triage processes. It has also highlighted the gaps in current triage research, such as but not limited to the need for tool-neutral triage training, and more focus on intelligence gathering. It has highlighted that extracting key information from a live device would help investigators make the best use of limited time during an investigation. It is pertinent to note that a live triage tool is not a replacement for a detailed examination however, it may be used to gather intelligence and subsequently advise on whether or not further analysis will be required.

During the course of this research, it was observed that there is a gap in the modelling

of live triage frameworks for investigating social media activity. When combined, the different aspects of these research have been useful in highlighting the need for frameworks/models/processes for social media investigations. They also emphasise the potential for investigators to obtain a substantial amount of evidence related to social media activity - this includes data from the web browsers and corroborating data from the OS. The important part however, is the ability to extract from these artefacts meaningful information which can then be used to infer a user's intent and activity. This extraction of meaningful information especially in the context of a live triage as related to this research, is through the identification of pseudo-unique identifiers in the data, which can then be used for inferring user activity.

This section has shown that digital investigators have a vested interest in reliably recovering and analysing artefacts from social media, in order to obtain inculpatory or exculpatory evidence for use in legal proceedings. It also highlights that evidence recovered through a live triage, should be subject to the same rules and principles of digital investigation and evidence handling.

2.7 Digital Evidence

As discussed earlier in **Section 2.5.1**, the goal of a digital investigation is to connect a crime to a perpetrator with the intention of prosecuting the offender, or enabling the enforcement of disciplinary actions. This can be achieved when there is evidence that links the perpetrator to that crime. This section provides a working definition of digital evidence; describes its properties; discusses the admissibility of evidence, and challenges posed to the collection and usability of digital evidence.

Digital evidence, often referred to as electronic evidence has a number of context-based definitions (i.e. criminal and non-criminal investigations). According to Casey [2004], digital evidence is “*data stored or transmitted using a computer, which supports or refutes a theory on how an event occurred*”. It addresses critical elements of an offence such as the intent of a culprit and has also been defined as “*captures of data transmitted over communication links, for example log files generated by the OS*” [Schatz, 2007]. Another definition from Mukasey et al. [2008] describes digital evidence as “*information or data of investigative value that is stored on, received or transmitted by a computer*”.

Digital evidence is useful in the investigation of a range of computer crimes such as fraud, possession or distribution of unlawful materials; as well non computer-related crimes like murder, kidnapping or extortion where digital evidence may be necessary

2.7. DIGITAL EVIDENCE

to tie a suspect to a victim or a crime [Casey, 2004]. For the purposes of this research, a working definition which is derived from Mukasey et al. [2008] and Casey [2004] will be used. Digital evidence in the context of this research, can be defined as any “*data of investigative value that is stored or transmitted using a digital device, which can be used in a court of law to prove or disprove a theory about how an event occurred*”. This definition is most appropriate as it approaches digital evidence from the role it plays in a legal context (i.e. addressing relevance to a case) and is broad enough to cover evidence recovered from a range of digital devices.

Digital evidence can be used in assessing alibis and profiling [Casey, 2004]. It reveals how an incident occurred, and the parties involved by establishing a link between a user and certain actions for instance, the interaction between two or more parties or visits to specific websites. Digital evidence provides the timelines necessary for the reconstruction of events and it is also known to exhibit the following characteristics:

- Latency – it must be processed to derive meaningful data.
- Fidelity – a copy of the original data will suffice as long as the copying process does not invalidate the evidence.
- Volatility – the evidence is at risk of being altered or deleted accidentally. Intentional change could be a deliberate attempt by an individual to distort or invalidate the evidence [McKemmish, 2008] however, this risk can be minimised by ensuring a chain of custody [Schatz, 2007] and the comprehensive documentation of processes.

According to Hosmer in [DFRWS, 2001][p. 10], the fundamental question surrounding digital evidence is identity i.e. linking the data to the suspect. Since in its natural state digital evidence has no informational value in itself (unlike witness statements or printed documents), it has to be translated and interpreted by the investigator through the application of certain rules and techniques [Sammes and Jenkinson, 2007][p. 9] for it to be understood and subsequently linked to an individual.

Oftentimes, secondary evidence may be obtained which would lead to the identification of the primary device on which additional data resides. Potential sources of digital evidence include but is not limited to hard drives, Internet or Communications Service Provider, portable storage devices, mobile devices, games consoles, and smart (IoT) devices.

Digital evidence has to satisfy two tests [Murr, 2007; Sommer, 1999]; first, it must

satisfy the specified rules of evidence in that jurisdiction e.g. Police and Criminal Evidence Act (PACE 1984), Regulation of Investigatory Powers Act (RIPA 2000), Investigatory Powers Act (IPA 2016) in the UK. Second, digital evidence must be believable and convincing to the court.

2.7.1 Rules of evidence

The admissibility and weight of digital evidence in court relies on the satisfaction of the applicable rules of evidence in a specific jurisdiction. There is no agreed standard for handling and processing digital evidence however, digital evidence, if tendered in court must exhibit certain characteristics [Casey, 2004, 2005; Murr, 2007; Sommer, 1999]; these rules that guide the admissibility of evidence are described below:

- **Relevance** – the evidence put forward must be material to the fact. This means that it must be related to the case for which it is being tendered.
- **Reliability and repeatability** – the evidence must be verifiable and the methods applied to the investigation must produce the same results when tested independently by a third party.
- **Authenticity** – the evidence must be genuine; not modified or forged. This can be achieved by generating a hash of the original device at the acquisition stage and comparing the value with that of the working copy. Casey [2011] notes that digital evidence may be altered prior to the hashing process so it is important that the person (or persons) handling the digital evidence is trustworthy, with strong supporting documentation.

In the event of a live acquisition, authenticity may be backed up with a properly documented and justified acquisition process. Authenticity also involves explicitly linking the evidence to a specific individual (e.g. establishing that a device belongs to or is being used by the suspect).

- **Validity** – the evidence must be complete. It must tell the whole story of how an event occurred. Incomplete evidence or evidence tainted with errors would be invalid and inadmissible. In exceptional cases however, where the amount of data or the volume of the device makes copying or imaging impractical, selective imaging of all the relevant evidence may be considered as an alternative measure. This evidence must then be backed up by a chain of custody (and contemporaneous notes where necessary) showing the accuracy of the content and process.
- **Credibility** – the evidence must be believable, beyond reasonable doubt, and trustworthy.

2.7. DIGITAL EVIDENCE

2.7.2 Digital evidence: Challenges

“... the variability of applications drives researchers and investigators alike to continuously try to hit a moving target”. - David Baker

As discussed in **Section 2.7**, digital evidence is concerned with linking an offender to a crime. Linking a crime to an offender may pose a challenge [Marcella and Menendez, 2007] for example, the use of cryptographic and concealment techniques [DFRWS, 2001]; evolution and complexity of digital devices [7safe, 2014; DFRWS, 2001; TWGECSI, 2001]; content diversity and volume [ACPO, 2012]; compression capabilities, and storage capacity of digital devices [McKemmish, 1999; Sommer, 1999] may affect the ability to acquire images of the digital device.

Other challenges include spoliation techniques [Marcella and Menendez, 2007] such as the destruction or alteration of evidence prior to prosecution for example, remotely connecting to a device in order to wipe it. The Internet and procedural issues like the lack of standardised protocols, procedures and terminology [DFRWS, 2001] can also be challenging. Evidence from the Internet may change with little or no notice for instance, changes to browser data storage format and the type of data stored on disk e.g. cookies, history, form fill data; this often means that investigators take longer to understand and decode the data before they are able to extract meaningful information from it. Internet users have learnt to be ‘tech savvy’ as a result of the advances in some of the technologies. This shows the need for academics and practitioners to keep up to date with these technologies [Berghel, 2003; McKemmish, 1999] by exploring new ways to overcome these challenges.

Marcella and Menendez [2007] advocates reviewing and assessing new technology from an offender’s perspective i.e. how the technology can be used to hinder investigations, or used to conceal a user’s identity and actions; because technology has the tendency to be used in a manner that deviates from its original purpose. For instance, cryptography is concerned with protecting data and communication, however, it can be used to thwart an investigation by making the entire device inaccessible to investigators. Encryption tools, evidence eliminating tools, hard disk erasers/scrubbers are widely available, often for free, on the Internet and have given rise to an increase in computer-related crimes [Marcella and Menendez, 2007] as perpetrators are able to use them to attempt to evade detection or prosecution.

2.7.3 Reliability of digital evidence

Reliability, in everyday terms can be defined as “*the quality of being able to be trusted or believed because of working or behaving well*” [Cambridge Dictionary, 2019]. In Casey [2002], reliability is referred to as “the consistency of a measuring or recording process”. Digital evidence could be user created (e.g. photos, videos, contacts, messages) or system created (e.g. logs, configuration files). These artefacts need to be assessed as trustworthy and accurately attributed to specific events in order to be admissible in court.

Casey [2011] posits that focus on the digital evidence itself rather than the process through which it was created is a more effective way of assessing the reliability. This involves inspecting the evidence to identify any signs of modification or damage e.g. deletion, corrupt or damaged files etc. Due to the nature of this research, reliability can also be assessed through the repeatability of the evidence examination and analysis process, to produce the same result in the same conditions. This ensures that the artefact is what it appears to be, for example, a URL indicating that a user uploaded a photo to a social media platform, should be the same (structure) if the same action (uploading a photo) is repeated, in the same conditions by a third party.

In the context of this research, reliability can be described as *the degree to which a specific artefact can accurately be correlated or attributed to a specific event*; this means that for artefacts from social media to be deemed reliable, they need to be able to attribute actions to a user and need to be corroborated with other evidence, providing context for what has been observed. For example, these URLs show the user added ‘*profile X*’ as a friend and visited ‘*profile X*’ five times; these JSON artefacts support this inference as it implies that the user and the owner of ‘*profile X*’ have a social relationship.

2.8 Working definitions

This section reiterates the working definitions and key terms presented in this chapter. These definitions will be used in this research in relation to the investigation and contextualisation of artefacts from user social media activity.

Digital forensics: *the application of computer technology to computer crime investigation, with a view to identify, preserve, process, produce and present court-worthy evidence.*

2.8. WORKING DEFINITIONS

Digital investigation: *the process of recovering, preserving, and processing digital artefacts in order to produce evidence that meets the requirements for admissibility in criminal proceedings.*

Digital evidence: *data of investigative value that is stored or transmitted using an electronic device, which can be used in a court of law to prove or disprove a theory about how an event occurred.*

Feature extraction: *an intelligence gathering process through which artefacts containing key information about a user's activities are identified, and retrieved using pattern matching methods such as RegEx, and can be corroborated by other pieces of evidence on the digital device.*

2.8.1 Key terms

Artefact: From a digital forensics point of view, and based on Locard's Exchange Principle, which states that *every contact leaves a trace*; an artefact in the context of this research can be described as *any data of evidentiary interest which has been, or is recoverable from a digital device.*

Entities: The Oxford Dictionaries [2019] defines entity as *a thing with distinct and independent existence*. In the context of this research, an entity could be any or all of the following; where 'all' means that the all the conditions below are met:

- a unique individual (a user) who has initiated, or is party to the exchange of information.
- an OS or user that performs an action.
- a web app (e.g a social media platform) through which a user can interact with a web server or another individual.

Action: An action in the context of this research is the *process of doing something on a digital device*. This could be sending/receiving messages, or interacting with other entities using social media. Artefacts are a collection of records of user actions.

Computer: Oxford Dictionaries [2019] defines computer as *an electronic device for storing and processing data, typically in binary form, according to instructions given to it in a variable program*. Thus, a computer can be broadly described as any device capable of storing and processing information.

World Wide Web: The World Wide Web (www) invented by Sir Tim Berners-Lee [World Wide Web Foundation, 2019] is an information retrieval service on the

Internet. Access to content/resources is achieved using a client-server model, where the client (web browser) requests information (resource) from a server through the use of a resource address/identifier (URL).

2.9 Summary

This chapter has reviewed related research and provided an overview of digital forensics. It highlighted the advances as well challenges faced in this discipline from data complexity and availability to the extraction of meaningful information from digital evidence. This chapter discussed social media activity and highlights how it presents an opportunity for misuse; it also shows that there is potential for investigators to recover data that can be used for the prosecution of offenders.

It discussed the importance of a live triage investigation and highlighted the need for more user-friendly, automated methods; the need for live triage frameworks for social media investigations; and the importance of feature extraction as a viable technique to be used in a live triage investigation and for user attribution. Finally, the rules of evidence and how the reliability of social media artefacts can be measured through corroboration and contextualisation are discussed.

Chapter 3 discusses the problem definition, the research question, objectives and the methodology.

3 Research Methodology

3.1 Introduction

This research aims to evaluate whether artefacts other than the content of user communication on social media can be used to attribute actions or relationships to a user. To do this, the research must be broken down into smaller tasks. This Chapter provides an overview of the problem definition as highlighted by the related reading in **Chapter 2**; proposes a research question; defines research objectives; and discusses the potential considerations made in relation to how this research is conducted.

It discusses the tools to be used in the completion of this research and the overall experimental methodology (experiments applied to specific aspects of the research will be discussed in **Chapters 4 and 5**).

The remainder of this chapter is structured as follows: **Section 3.2** discusses the problem definition; **Section 3.3** discusses the research question; **Section 3.4** discusses the research objectives; **Section 3.5** discusses the research considerations; the tools used during this research are discussed in **Section 3.6**, with a tool summary table at the end of the section; **Section 3.7** discusses the social media experimental samples, and the terms of acceptable use of these platforms are discussed in **Section 3.8**. The baseline setup for the experiments are discussed in **Section 3.9**. The experimental methodology is discussed in **Section 3.10** and **Section 3.11** provides the chapter summary.

3.2 Problem Definition

Following a review of relevant literature, knowledge gaps and limitations in the use of live triage in social media related investigations were identified [**Section 2.6**]. The reviewed material highlighted two separate streams of work:

- i. the analysis of social media artefacts, and
- ii. the use of live triage in digital investigations.

The existing research on the use of digital triage were often either focused on the use of live triage for incident response or malware investigations [**Section 2.6.3**], generic browser artefact recovery for social media activity [**Section 2.3**], and the

use of Post-mortem triage to prioritise such evidence [Section 2.6.2].

Triage is a growing field in digital forensics however, there has been considerable effort and focus on post-mortem (PM) triage. Live triage as discussed in [Section 2.6.3] is a technique that can be used by investigators to retrieve quick actionable intelligence at a crime scene. It may be argued that triage is not a core part of the forensic process by definition [Section 2.5.2] as it does not meet the requirement for admissibility in court by default. This to some extent can be said to be true, however, the evidence recovered during a live triage should be forensically sound, and in compliance with the relevant guidelines (e.g. ACPO); if required as supporting evidence in court.

3.2.1 Challenges

The advancements and variety of digital devices available to users produces a vast number of user and system activity logs. The volume of data generated and stored on these devices introduces the data complexity challenge. For example, a user could have one or more laptops or desktop computers, with terabytes of storage space including external storage devices; all with varying data formats. It also brings about a lack of standardisation as seen in the case of IoT devices, as proprietary data formats are often used for storing data. Other technical challenges include the ever changing technical landscape, provider limitations, and getting digital forensics tools validated with artefacts.

Every time a user communicates on the Internet, digital traces are left behind on the digital devices involved and these traces can be used in user identification or profiling, event reconstruction, and user attribution [Section 2.5.1]. The constant evolution of web technologies for example, patches, security/platform updates; changes to the User Interface (UI) of web applications such as Twitter, Facebook, and the browsers' method of enhancing user experience has subsequently made investigating web based technologies a challenge [Section 2.3]. This challenge extends to the investigation of social media activity and highlights the need to develop a generalisable technique that can be used in identifying, recovering and analysing these artefacts at the scene of an incident, especially in time critical cases.

It is pertinent to note that due to several factors, for example, the underlying technologies and frameworks used to build a web app, or the way browsers handle artefacts created by these web app, the content of user communication (i.e. textual or graphic content) may not always be immediately available or accessible on disk [Section 2.4.3]. Thus, the investigator may need to rely artefacts (e.g. URLs,

3.2. PROBLEM DEFINITION

browser cache data, etc.) found on disk when retrieving evidence in a live triage.

Jurisdictional constraints and local privacy laws need to be considered whilst acquiring data in a live triage and measures should be taken not to include the personal information of uninvolved entities [Section 2.4.3]. It is important to note the history of the system from which data is being collected (e.g. previous owners of second-hand devices). When acting on the information retrieved from during a live triage, as per the rules of evidence, it is essential to establish the reliability of the artefacts – are they a true representation of what occurred? Could they have been created due to an unrelated action or event? Were the devices wiped and sanitised prior to change of ownership? These questions will help with the exclusion of false positives from the results of a live triage.

Social media platforms have made efforts to secure the content of communication on their platforms; sometimes through the use of encryption [MercoPress, 2019] which may result in difficulties accessing communication content during an investigation. Thus, there is a need to find a balance between user privacy and data availability for law enforcement.

3.2.2 Limitations

A number of limitations were identified during this research. For example, the focus on the use of live triage in incident response/malware investigations [Section 2.6.3]; where the goal is business continuity and there may be little to no requirement for following ACPO principle 2. In addition, some live triage methods used for incident response may require pre-installation of agents on the target device and may not be ideal for a forensic investigation. Although the information retrieved through a live triage may not always be used the single source of evidence in court, it can provide valuable intelligence. It can be used to identify other potential sources of digital evidence; for example, during the search and seizure procedure, the digital forensics team can identify other devices that may be connected to the case, leading the investigator to determine the depth of analysis required.

The volume of data, and the complexity of file formats and storage environments sometimes makes it difficult to quickly spot activities of interest. This sometimes makes it impractical to rely on conventional forensic tools for a live investigation; although some of these tools may have the capability for live acquisition, additional effort would be required to process the evidence and to obtain the data of interest.

3.3 Research Question

As discussed in **Section 3.2**, there are limitations and challenges faced by digital investigators. In summary, the key problems identified in this research are:

1. The increasing volume of data, proprietary formats, and the proliferation of digital devices are a challenge.
2. Web browsers are constantly and rapidly going through updates/upgrades; changing versions (and across devices).
3. Changes to social media platforms (web apps) occur at a different pace and style to browser updates/upgrades. These changes may not be immediately obvious and may not be version changes.
4. When social media is used through a web browser, it leaves traces on the user's hard disk.
5. The content of user communication may not be immediately accessible for retrieval and processing without additional effort.
6. Artefacts from social media activity in themselves may not always be reliable and may need to be corroborated. For example, data about unrelated persons created as a result of background processes.

Taking these central problems to the research focus into account, and recognising that attributing actions to a user, and identifying other entities involved is useful in digital investigations; the following research question is proposed:

“In the absence of the contents of user communication, can social media artefacts be used to reliably attribute actions to individual users? Can they be used to link other entities involved in the communication?”

Where the following key terms in the research question are defined in the context of this thesis:

1. **absence of the contents of user communication:** communication was defined in **Section 2.4.1** as the exchange of information between two or more entities, using web based technologies as a medium for transmission. When investigating social media activity, contents of user communication refers to the information exchanged between two users, (text, images, videos, data files) on a social media profile. In the context of this research, the absence of these contents implies that the information is not readily available due to a number of reasons. For example, it may not have been written to disk at the time of recovery; it may have been written to an unstructured

3.4. OBJECTIVES

part of the disk and would require time and effort to locate; it has been partially or completely overwritten.

2. **social media artefacts:** this, in the context of this research, are the artefacts created as a result of social media activity. This may be user initiated e.g. sending and receiving content on a social media platform, or as a result of background processes running during user activity.
3. **reliably attribute actions:** reliability has been defined as the degree to which an artefact can be accurately attributed to a specific event [Section 2.7.3]. This research question attempts to determine if artefacts other than the content of user communication can be used to attribute actions to a user during a live triage investigation. To meet this requirement, baseline experiments need to be conducted and used as a means of evaluation.
4. **link other entities involved:** social media platforms have a lot of users, often in billions. For example Facebook, in Q1 2018, disabled about 583 million fake accounts, and took down millions of spam, graphic violence and other content involving adult nudity and sexual activity. The abuse of usage terms has resulted in these platforms implementing measures through the establishment of forums like Global Internet Forum to Counter Terrorism (GIFCT) [2019] to combat the use of their platforms for unlawful activities.

Users tend to form social relationships with other individuals with similar interests. In UNITED STATES OF AMERICA v. ASHRAF AL SAFOO [U.S.D.C., NORTHERN DISTRICT OF ILLINOIS, 2018], it was shown that social media was being used by a group of individuals to disseminate materials that promote violence. Due to the nature of these messages, the user accounts were suspended. It highlights the persistence of criminals who attempt to regain access to these platforms by hijacking the accounts of legitimate social media users or creating accounts under fake identities. This indicates that users with criminal intent would form social relationships and work in groups with like minded individuals. This presents an opportunity for investigators to use social media artefacts to find evidence of association. This helps the investigator determine the connections between individuals and link them to other associates based on their social relationships.

3.4 Objectives

The following objectives are based on the review of related literature in **Chapter 2** and the key problems identified in the Problem Definition [Section 3.2]. In order

to answer the research question, the objectives need to be broken down into five activities. First, communication content needs to be characterised, followed by a categorisation of social media artefacts. Then a framework will be developed for the recovery and analysis of social media artefacts and a PoC will be developed to test the framework. The reliability of the social media artefacts will be tested and finally, the artefacts will be used to determine the relationships between entities.

1. **Characterise communications content.** The contents of user communications was described as the message(s) exchanged between two entities [Section 3.3]. For example, a private message on Facebook, a tweet or a shared file. In relation to the research question, this *content* may not be immediately accessible on disk but user action can be inferred based on other artefacts such as URLs and browser cache or session information. This addresses problems 1 and 5 [Section 3.3]. This activity involves devising a method to differentiate the content exchanged on a social media platform from other artefacts created as a result of social media activity.
2. **Categorise artefacts generated from the use of social media.** As discussed in [Section 2.3], accessing social media through a web browser leaves traces of user activity on the user's hard disk. These artefacts will be investigated to establish their relevance and how they can be used to determine user actions and events that occurred. This will address problems 2 and 4 [Section 3.3]. This will involve categorising the artefacts into *case specific* and *context* artefacts to determine which artefacts are directly related to user action and the artefacts that can be used to contextualise the action.
3. **Develop a framework for the recovery and analysis of social media artefacts.** This activity relies on the outcome of Objective #2 because it requires the results from the categorised artefacts in order to determine the baseline for artefacts of interest. The framework will address the challenge posed by data complexity, providing quick actionable information that can guide an investigation in a time effective manner. It will also provide information on areas that may need to be prioritised during subsequent, detailed forensic analysis. This objective requires a means of demonstrating the feasibility of the framework, so a PoC will be developed to test the framework. This will address problems 2, 3, 4, and 5 [Section 3.3].
4. **Assess the reliability of social media artefacts.** Reliability was defined as *the degree to which a specific artefact can accurately be correlated or attributed to a specific event* [Section 2.7.3]. This activity requires testing that the artefact recovered can be used to attribute actions to a user, and

3.5. CONSIDERATIONS

that it can be corroborated with other evidence. This will address problems 4 and 6 [Section 3.3].

5. **Determine the relationships between communicating entities.** This activity involves the use of the artefacts recovered by the framework to identify persons involved [Section 2.7.3] and determine the associations between persons or items of interest. This will address problems 4 and 5 [Section 3.3].

To achieve these objectives, a series of experiments will be conducted. These will involve a range of activities such as: defining an experimental environment; selecting social media platforms, and test users; creating social media accounts for the experiments, and simulating a range of user social media activity. This is followed by the manual analysis of the disk; the results will be used to set baseline for the framework. These activities are discussed further in **Sections 3.7, 3.9; Chapters 4 and 5**. The tools used for the experiments and analysis are discussed in **Sections 3.6**.

3.5 Considerations

In addition to the technical challenges discussed in this thesis, digital investigators also face a number of non-technological challenges. Morris [2013] posits that there are various limitations that impact on the conduction and/or outcome of an investigation, with regards to maintaining evidential value and the admissibility of artefacts. Experimental, evidential, legal, procedural, and reliability constraints were identified as relevant to this research and are discussed in this section.

3.5.1 Experimental Constraints

The experimental constraints in relation to this research were identified as the limited number of social media platforms selected; the limited user activity performed on social media; the experimental setup; network traffic artefacts, and the use of existing digital forensics tools. This is because the experiments were performed in a Windows OS virtualised environment with a limited number of social media platforms and web browsers. In addition, a limited subset of user activities were performed; thus the focus was primarily on the artefacts that existed on the virtual machines.

There were concerns that the experimental setup confined the research to the selected social media platforms, and user activities performed on them through web browsers. It is noted that there might be other artefacts created as a result of social media activity through a different operating system, desktop application, different web

browsers, or through other types of user activities not covered in this research.

There were also concerns that the real world application of this research may be limited, as the experiments were not fully representative of the various ways in which social media can be used, or the artefacts that can be recovered. However, it is argued that the experiments demonstrate a viable method for identifying the artefacts of interest from social media activity, that may be recovered during an investigation. The experimental methods created in this research take the principle of repeatability into account, as a way to ensure that the results can be independently verified. Thus, it can also be argued that the experimental method can be used to establish a baseline, and can consequently be applied to other social media platforms not covered in this research.

3.5.2 Evidential Constraints

Digital forensics is concerned with identifying, preserving, processing and producing digital evidence [Section 2.2.1]. In the context of this research, this refers to the artefacts from social media activity recovered from a target device.

This research is focused on the recovery of digital evidence related to social media activity from live environments. To achieve this, it is important to first identify and understand the type of evidence that is of interest to an investigator. This involves performing experiments on social media, then recovering the evidence of interest through a post-mortem forensic digital forensic analysis, using existing tools designed to preserve the state of the target device, to ensure that evidential integrity is maintained. This is followed by the recovery of the identified evidence from a live environment.

This research identified that there is the possibility of cross-application of methods and principles from PM triage to live triage scenarios. For example, in the context of a PM examination of digital devices, triage was defined a process of screening recovered digital evidence to identify its case relevance and evidential value [Section 2.6.2]. Although this approach is from a post-acquisition perspective, this principle can be applied to live triage investigations - i.e. the screening of a digital device to determine if the evidence therein has any probative value.

There were also concerns that the results may differ across operating systems however, as the focus is on browser artefacts, it can be argued that individual browsers have a defined structure that persists across operating systems thus, the techniques for recovery can be adapted. It was noted that in a live triage investigation, some system

3.5. CONSIDERATIONS

files may not be accessible as they are locked down by the operating system however, a method was devised to create a copy of these locked files for subsequent analysis outside the live environment. Whilst doing this, utmost care must be taken to handle the evidence correctly, document the process, and ensure that the action can be justified.

3.5.3 Legal Constraints

Every investigator must take care to ensure that digital evidence is collected in a forensically sound manner, and also, that it is handled appropriately by maintaining an audit trail (chain of custody). This will ensure that the evidence is not inadvertently compromised, and that the integrity is maintained, in order to meet the requirements of admissibility.

Live triage evidence in the context of this research, is aimed at providing valuable intelligence to investigators. Although it may not be intended for court [Section 3.2], it is important to ensure that the process of recovering evidential material does not compromise the target digital device from which evidence is being retrieved. If peradventure modification to the target device is unavoidable, e.g plugging in a USB device to run a tool; this action needs to be fully and clearly documented (audit trail).

Digital evidence can be used to show that a suspect had the intent to commit a criminal act [Casey, 2013]; or whether the evidence found was as a result of unrelated activity unknown to the suspect. It is important to understand the legal implications of evidence recovered from a live triage investigation. Due to the limited scope of artefacts recovered during a triage, it may not be fully representative of the evidence pertinent to a case and as such would not meet the admissibility requirement of *completion* [Section 2.7.1], but it may be used to support other evidence found during a detailed forensic analysis.

In order to ensure that live triage evidence can be admitted to court if required, it is important to understand the sequence of events leading up to the creation of the artefacts of interest; and to use established and appropriate methodologies to recover the evidence, with minimal impact to the state of the target system in compliance with ACPO principle 2. Another legal constraint identified is related to the type of evidence that can be recovered. For example, due to the way browsers handle data (e.g. caching), it is possible that identifying information from other social media users may be available on disk, and recoverable as part of the web browser artefacts.

Investigators need to be able to identify and possibly exclude these artefacts through a process of corroboration, and only submitting evidence of probative value to the court; i.e. evidence that provides information that can be used to prove or disprove the theories about how an event occurred.

3.5.4 Procedural Constraints

In digital forensics, the procedures for handling digital evidence in the UK were provided in the form of ACPO guidelines. Although the guide is now a few years old, and ACPO has been replaced by the National Police Chiefs' Council (NPCC), these guidelines help investigators employ a culture that ensures proportionality, and that the evidence collected meets the requirements for admissibility in court [Section 2.5.3].

Live triage investigations involve direct interaction with the target device during an investigation. Although evidence handling principles [Section 2.5.3] highlight the importance of not altering the state of the target device, an exception is made for scenarios where modification is unavoidable; provided all actions are documented and justified. It can be argued that live triage investigations fall into this category. It is consequently important for the digital investigators to be adequately trained, with an understanding of digital forensics processes and methods. The methodologies and the impact of tools used in live triage investigations should be documented, justified and implemented by a competent investigator [Section 2.5.3]. This should also be clearly explained in the forensic report submitted to a court.

Considering that computer crimes are likely to cross jurisdictions [Seals, 2015]; and that there might be differing standards on how digital evidence is handled, the principles of digital evidence from ACPO, International Organization on Computer Evidence (IOCE) and Technical Working Group Electronic Crime Scene Investigation (TWGECSI), were reviewed in **Chapter 2** and the similarities were highlighted in **Section 2.5.3**.

3.5.5 Reliability Constraints

In the course of this research, the principle of *repeatability* was taken into consideration [Section 2.7.1]. This involves providing a generalisable means of verifying experimental results. This means providing adequate information on the process such that if the evidence recovery method is implemented by a third party, the results can be reproduced either manually or through automation.

Repeatability was considered as specified by ACPO principle 3. However, due to the

3.5. CONSIDERATIONS

nature of a live investigation, repeated access to a live device is not recommended as part of the evidence verification process. It was determined that live triage evidence may be verified through logging and detailed documentation of the evidence retrieval process. A full forensic analysis may also be used to verify live triage evidence, as it will provide deeper context e.g. recovering image files and content of communication that can be used to bolster the live triage evidence.

As noted in the problem definition [Section 3.2], web technologies are constantly and rapidly evolving. It is important to note that due to the nature of this research, and considering the time constraints, the decision was made to focus on a generalisable process of intelligence gathering through a live triage; rather than focusing on individual artefacts. This would address the repeatability requirements of the evidence recovery process.

3.5.6 Ethical Constraints

Ethics in research involves conducting research in an acceptable manner; with integrity, through the avoidance of harm to anyone, and the resolution of personal conflicts with honesty, fair dealing and truthfulness [Cranfield University, 2019]. Cranfield University’s Research Ethics Policy defines a set of requirements to be met by the researcher with oversight from the supervisor.

To demonstrate that this research is conducted in an ethical manner, the evidence shown in Table 3.1 is used to measure compliance. In addition, a research proposal was submitted and ethical approval from Cranfield University Research Ethics System (CURES) subsequently granted.

Table 3.1: Ethical considerations and compliance

Requirements	Status
Risks of dataset contamination	Due to the nature of social media platforms (e.g. Twitter), where additional content is delivered in the background, there is a risk of unintentional contamination of the dataset. Such content with personal information may be recoverable through the PoC framework however, this will be mitigated by redacting such data where necessary or excluding them from published results.
Risks to the general public	<p>The research data collected is from test user accounts created by the researcher. This ensures that the researcher is able to control, to a certain degree, the type of data created and collected.</p> <p>To mitigate any risk to the general public, the user accounts created during experiments will be deleted from the social media platforms at the conclusion of the research.</p>
Benefits to the general public	The data collected is from user accounts created by the researcher thus, there are no benefits to the general public.
Designing Research to mitigate data collection from live subjects	<p>The nature of the research does not require live subjects, so several virtual machines (VMs) were created as test environments to simulate ‘real life’ user activity. Communications were established between the test user accounts; and a traffic analysis tool was used to monitor background data, to highlight any content delivered in the background which may not be visible to the user in the web browser.</p> <p>The use of virtualised test environments and test user accounts was aimed at minimising the unintentional collection of live third-party data. The VMs used during the experiments will be securely destroyed at the conclusion of the research.</p>

3.6 Tools

A range of existing digital forensics tools were used for the experimental and analysis stages of this research. This was done to enable the manual identification and recovery of evidential material of interest, from a target system in a powered off state. The manual analysis was done by hand, with minimal automation and reliance on the tools. This section discusses the process employed in the selection of the tools used at the experimental stage, and for the manual analysis of artefacts. The two categories of tools used (experimental and analysis tools) are discussed in **Sections 3.6.1** and **3.6.2** respectively, while the tools summary is presented in Table 3.2.

3.6. TOOLS

3.6.1 Experimental Tools

Experimental environment

During this research, the requirement for a controlled environment and the ability to retain snapshots at the end of each experiment was identified as a viable means of preserving varying stages of the experiments. It was determined that a virtual machine (VM) would provide the required flexibility and control needed as a virtual disk can be deleted or reproduced without the need to rebuild or reinstall an operating system should an error occur. It also means that the basic configuration stays the same across all machines used.

Barrett and Kipper [2010] identified several categories of virtualisation such as: full, hardware-assisted, and operating system virtualisation. Of all the identified categories, full virtualisation possesses the characteristics required for conducting the experiments which were envisaged within this research. A VM is a software-emulation of a physical computer; it runs as a software application on the host operating system, uses the host hardware, and allows the execution of a guest operating system in an isolated environment on the host.

The virtualisation tool of choice for this research is VMware Workstation Pro installed on a Windows 10 PC. Other alternatives such as VirtualBox, Windows VirtualPC were considered during this research. VMware Workstation was determined to be more mature and stable [VMware, 2019] in comparison with VirtualBox, and offers a wider range of device and driver support. Windows VirtualPC is not supported on Windows 10 and as such was discounted along with VirtualBox.

Logging

Procmon v3.50 [Russovich, 2018] is a tool used for passive logging on a Windows computer. It logs events in a sequence so that a chain of events can be followed just the way they occurred. It will be used to log the files, services and processes called during the experiments (e.g. plugging in a USB, installing a web browser). The procmon logs will then be examined to identify activities of interest; it is expected that the logs will highlight activities that may take place during a live investigation.

Fiddler v5.0 [Telerik, 2018] is a free, cross-platform web debugging proxy for web browsers. One of its key features relevant to this research is HTTP/HTTPS traffic recording. The captured traffic contains the full HTTP request and response; including other metadata, and can be stored in an archive (.saz) file for subsequent analysis. The data from Fiddler can be used to provide insight about traffic from

social media activity; enabling a comparison between what was observed during the experiment, and what was written to disk.

3.6.2 Analysis Tools

A range of existing digital forensics tools were used for the analysis of social media artefacts created during the experiments.

Hexadecimal Editors

A hexadecimal editor (hex editor) is used as a primary and general tool for the manual analysis of the virtual hard disks (.vmdk) from the experiments. The hex editor of choice for this research is WinHex v18.5 [X-Ways Software Technology AG, 2018]; it will be used for the extraction of artefacts that require further analysis with third party tools (e.g. SQLite database files, registry hives). For this research, a simultaneous keyword search function was required; WinHex has this feature which allows for specific keywords to be searched. The keywords are entered one word per line and the results indicate the location where the keyword is found.

Although useful, this function can return a number of false positives depending on the keyword entered; so care must be taken when choosing keywords to search for. It is important to note that the keyword search will only pick up data (hits) that are visible to it; this limitation means that WinHex will not have captured the content(s) of encrypted, or compound files. WinHex's built-in data interpreter was also deemed useful; it allows data analysis by calculating the value of a selection of bytes, for example, timestamps, file size, or the length of an item of interest such as a string of text. It is also useful for understanding the format/structure of data of interest.

Browser Artefacts

The analysis of web browser artefacts required tools that could be used to view, and export the data from the range of file formats available. To this end, the following tools were selected; DB Browser for SQLite [DB4S Project, n.d.] is a graphical open source tool that can be used to create, view, and edit SQLite database files. It can be used to issue SQLite queries; search records in a web browser SQLite database file, and export the results to CSV for further analysis.

MZCacheView v1.81 [NirSoft, 2018c] and ChromeCacheView v1.80 [NirSoft, 2018a] are Windows based tools, designed for the visual inspection of cache contents from Firefox, and Google Chrome respectively. For each item in the cache, the HTTP header information such as URL, file size, content type, last modified time, fetch count, last fetched etc. are returned. The cache content can be exported to a working

3.6. TOOLS

directory for further inspection/analysis. The header information can also be copied, and pasted, or saved as a text file.

Python 3 [Python.org, 2019] is a cross-platform, high-level programming language with built-in libraries and an easy to learn syntax. Existing parsers written in Python [Bancel, Jean-Rémy, 2015; Benson, 2019], as well as custom scripts will be used to decompress, and parse files of proprietary formats recovered from the web browser.

JSONViewer [Hay, n.d.] was selected as a tool for reading and visualising the JSON data recovered from web browsers; making the data clearer and highlighting artefacts of interest within the JSON files.

System artefacts

To analyse artefacts created as a result of system activity which can be used to infer user behaviour, the following tools were selected; RegRipper v2.8 [Carvey, 2018] is an open source forensic tool written in Perl. It provides dual functionality in the form of a Graphical User Interface (GUI), and Command Line Interface (CLI). It can be used to extract data of interest from the Windows Registry, and allows the parsing of individual registry hives. It consists of a range of parsers that can be run against registry hives; the results from RegRipper contains keys, values, and data of interest which can be saved to a text file for further analysis. RegRipper also produces a log file for each Registry hive parsed.

FullEventLogView v1.37 [NirSoft, 2018b] presents a way for the system log to be examined in an efficient manner. It is a Windows based tool that displays information such as the event time, event ID, description etc; and provides both GUI and CLI functionality, allowing flexibility of usage. The GUI version of FullEventLogView was used; it provides a 'Properties' view which presents a quick view of the event log type, identifiers and associated descriptors. The results can be exported to a range of file formats - CSV, HTML, text etc., allowing for text searches and data filtering.

PrefetchForensics v1.0.4 [Woan, 2013] is a tool that can be used to extract valuable information from Windows Prefetch files. It provides CSV and HTML options for exporting the results which can be further analysed to extract artefacts of probative value.

Validation Tools

FTK Imager Lite v3 [Access Data, 2010] is a digital forensics tool used to preview the data on a digital device; and can also be used for evidence acquisition. This tool can be downloaded as a compressed file; it requires no installation and can be executed once the contents are decompressed. FTK Imager Lite supports mounting of *.vmdk* files as local drives which can then be imported as physical drives for examination. It was selected as a validation tool to inspect the contents of the virtual disks (*.vmdk* files) shown in WinHex.

RegDecoder [Case and Marziale, 2011] is an open source digital forensics tool for the examination of Windows Registry hives; with parsers that can be run against hive files. Unlike RegRipper (GUI) where hives are individually analysed, RegDecoder allows all hive files to be loaded at the same time for processing. It was selected as a validation tool to corroborate the results from RegRipper. The results from RegDecoder can be exported to CSV for further analysis.

The tools used, whilst ideal for a detailed analysis on a disk image, were not fully compatible with a live triage. Although some of the tools do not require installation before use, it was observed that access to live data was sometimes unobtainable. Where a digital investigator is able to access live data, the data often requires extraction and then follow-on analysis with a third-party tool. For example, there is the capability to mount a ‘live’ hard disk or partition in FTK Imager Lite however, as this tool is not designed for a live analysis, it means the data of interest will need to be exported and analysed.

As discussed in this section, although these tools are not designed for a live triage; during the course of the experiments and initial manual analysis, it was determined that there is scope for the use of some of these tools in a live triage, given the right conditions. This process is discussed in **Chapter 6** as part of the framework design process.

Table 3.2: Tools Summary - Data examination and acquisition tools (in alphabetical order)

Tool	Description
Chromagnon [Bancel, Jean-Rémy, 2015]	Used for the analysis of Google Chrome artefacts.
ChromeCacheView v1.80 [NirSoft, 2018a]	Used as an individual artefact analysis tool for Chrome cache artefacts.
DB Browser for SQLite [DB4S Project, n.d.]	Used for individual artefact analysis. This was used for the extraction and analysis of the web browser SQLite database artefacts.
DCode™ v4.02a [Wilson, 2009]	Used for the interpretation of timestamps.
Fiddler v5.0 [Telerik, 2018]	Used as a general examination tool. Fiddler was used during the experiments to log all traffic between the experimental system and the Internet. It is an independent program that can intercept any program/process running on a user computer. Fiddler acts a proxy and all traffic between a browser and a web server flow through Fiddler.
FTK Imager v3 [Access Data, 2010]	Used for general examination. FTK Imager was used for the extraction and multi-tool validation of artefacts.
JSONViewer (Allan S Hay) [Hay, n.d.]	Used for individual artefact analysis; for viewing JSON artefacts.
MiTeC Windows Registry Recovery v1.6.1.0 [MiTeC, n.d.]	Used for individual artefact analysis of Windows registry and for multi-tool validation.
MZCacheView v1.81 [NirSoft, 2018c]	Used an artefact specific tool in the analysis of the Firefox cache artefacts.
NirSoft FullEventLogView v1.37 [NirSoft, 2018b]	Used for individual artefact analysis of Windows event logs.
Prefetch Forensics v1.0.4 [Woan, 2013]	Used for individual artefact analysis of Windows prefetch artefacts.
Procmon v3.50 [Russinovich, 2018]	Used as a general examination tool. Procmon was used during the experiments to monitor activity on the experimental system. This includes file system activity and user initiated activity such as software installation.
Python 3 [Python.org, 2019]	Used for scripting, and to automate the conversion of files e.g. from LZ4 format to JSON. Python was also used to extract features from extracted URLs.
RegDecoder [Case and Marziale, 2011]	Used for individual artefact analysis of Windows registry and for multi-tool validation.
RegRipper v2.8 [Carvey, 2018]	Used for individual artefact analysis of Windows registry and for multi-tool validation.
WinHex v18.5 [X-Ways Software Technology AG, 2018]	Used as a general examination tool. This was used to view the contents of the virtual disk and for the extraction of artefacts to be analysed with third-party tools (listed below). It was also used for simultaneous keyword search. WinHex simultaneous search function allows a list of search terms (one per line) to be searched at once.

3.7 Social Media Selection

Real-life data are of paramount importance in any scientific study. Unfortunately, this is a challenge in digital forensics due to factors such as privacy, or the sensitivity of a case intended to be researched; ethical issues with the use of data from live subjects; in addition, rapid changes in technology may restrict what can be done even if live data is obtained with consent. This means that researchers in digital forensics often have to create controlled environments in which real live activities can be simulated. To this end, Facebook, Google Drive and Twitter were selected to simulate user social media activity for this research.

The data from social media activity can be said to be representative across different Operating Systems although the location where they are stored is usually browser dependent [Section 2.3; Section 3.2.1]. The web browsers selected for this research, Firefox and Google Chrome are discussed in Chapters 4 and 5.

Social media can be described as a means through which communication can be propagated [Section 2.4]; i.e. the generation or creation of content, and sharing the content with other entities using a social media platform such as Facebook, Twitter, and Google Drive. For the purposes of this research, Google Drive is classed as a social media platform as it presents a platform for propagating communication through content sharing and collaboration. The selection of social media platforms for this research was based on market research and a review of social media statistics [Figure 3.1] to determine the size of their user base (Facebook $\Rightarrow$ 1.69 billion in 2020; Google Drive $\Rightarrow$ over 1.5 billion active users in 2020; Twitter $\Rightarrow$ 353 million monthly active users in Q3 2020), the growth rate, and the services they provide.

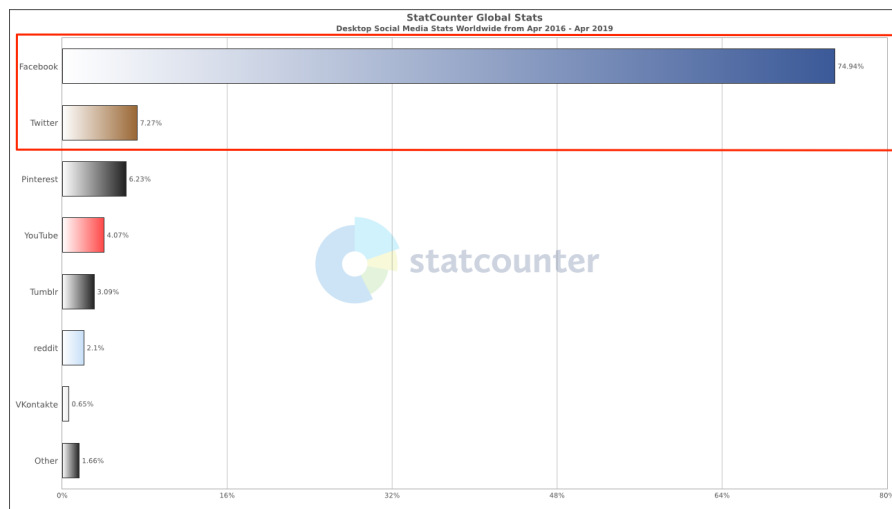


Figure 3.1: Social Media Stats

3.7. SOCIAL MEDIA SELECTION

3.7.1 Facebook

Facebook is a web application that provides social media services to Internet users and as at Q2 2020, had 2.70 billion monthly active users (MAUs) [Facebook, 2020b]. Some of Facebook’s features include: news feed; private/instant messaging; searching for, adding, or removing ‘friends’; posting on walls (user profile space), and timeline; liking, reacting, and commenting on posts; starting or joining groups, and events; and a marketplace that enables users to advertise, buy, or sell items.

Facebook as at the time this research was conducted, is the largest social media platform globally. It was selected due to its large user base, and the increase in the number of criminal activities linked to the platform [Section 1.2]; [McGuire, 2018; RSA, 2016].

3.7.2 Google Drive

Google Drive is a cloud-based, document storage and sharing service for Internet users. It provides web and desktop services, giving users the ability to synchronise files across multiple devices. Users with Gmail accounts are provided with Google Drive storage by default; with 15GB of free storage. Features provided include: document creation within Google Drive, or uploading existing documents; support for multiple file types such as archive, audio, image, markup/code, plain text, video; a range of Adobe, Microsoft, and Apple files. Google Drive users can add other users as collaborators, giving them the ability to edit and comment on documents [Google, 2013, 2016]. Up to 30TB of additional storage space can subsequently be purchased from Google if required.

The features available to users, especially content storage and collaboration, sets Google Drive apart from traditional social media platforms; they also make it an attractive candidate for misuse as the platform could be used for disagreeable or unlawful activities [Paganini, 2014a,b; Raphael, 2021].

3.7.3 Twitter

Twitter is a real-time information network that allows users to connect to each other globally. Twitter is described as *“a service for friends, family, and coworkers to communicate and stay connected through the exchange of quick, frequent messages. People post Tweets, which may contain photos, videos, links, and text. These messages are posted to your profile, sent to your followers, and are searchable on Twitter search”* [Twitter Help Center, 2019].

Twitter users can create, share, or like content which in some cases may be disagreeable

[Bello and DiBlasio, 2013], manipulated, inaccurate, or unlawful. Such content may go viral and subsequently spiral out of control.

In order to regulate the use of their platforms, service providers have defined several rules, stipulating what they deem unacceptable and how users are expected to comply. These are discussed in **Section 3.8**. User social media activities are discussed in the context of the experiments in **Chapters 4 and 5**.

3.8 Social Media: Terms of Acceptable Use

This section highlights the applicable terms of service/acceptable use for the selected social media platforms reviewed as at the end of January 2021. Common themes identified across all three platforms (Facebook, Google Drive and Twitter) include:

- the protection of minors
- regulating gory/obscene content
- policing harassment/spamming
- prohibiting impersonation/deception
- curbing exploitation (sexual or otherwise)
- preventing the trading of restricted good and services
- forbidding interference with systems and services

This review activity was performed for a number of reasons. First, it is important to comply with each service provider’s rules and policies. Second, it is a requirement that this research be conducted in an ethical way, in compliance with Cranfield University’s research ethics policy [Cranfield University, 2019]. This ensures that no harm or distress is caused to the public. Finally, it is important that each platform is used as intended by the service provider.

3.8.1 Facebook Community Standards

Facebook as at the time this research was conducted, has 27 policies that make up the Community Standards, outlining what is considered acceptable and unacceptable behaviour on the platform.

According to Facebook [2020a], these policies are based on feedback from the user community and “the advice of experts in fields such as technology, public safety and human rights”.

There are six categories which include Facebook’s compliance and board oversight. All six categories were reviewed in the for this research:

3.8. SOCIAL MEDIA: TERMS OF ACCEPTABLE USE

- I. Violence and criminal behaviour
- II. Safety
- III. Objectionable content
- IV. Integrity and authenticity
- V. Respecting intellectual property
- VI. Content-related requests and decisions

The policies aim to prevent and protect against a wide range of activities and behaviours such as:

- **[Part I: Sections 1 and 3]** potential offline harm that may be related to content on Facebook and the facilitation, promotion or admission of certain criminal or harmful activities which may lead to offline harm or threats to public safety.
- **[Part I: Section 5]** fraudulent activity that can harm people or businesses, through the removal of content that intentionally deceives, defrauds or exploits others for money or property.
- **[Part II: Sections 9 and 11]** unwanted malicious contact; content that shares or solicits personal identifiable and/or financial information about other individuals.
- **[Part III: Section 6]** cruel/insensitive content.
- **[Part IV: Sections 17, 18, 20, 22]** impersonation and identity misrepresentation; content that is designed to deceive or behaviours designed to enable other prohibited behaviours and activities.
- **[Part V: Section 24]** content that violates someone else's intellectual property rights.

Examples of unacceptable behaviour specified in Facebook's Community Standards include:

- threats that could lead to the loss of life (and other forms of high-severity violence) targeting people or places.
- statements of intent to commit violence; or the advocacy of violence.
- content created with the intention of outing an individual as a member of a designated and recognisable at-risk group.
- depicting, promoting, advocating for or encouraging participation in a high risk viral challenge.
- promoting or advocating for harmful miracle cures for health issues.
- hacking when the intent is to hijack a domain, corrupt or disrupt cyber systems, seek ransoms or gain unauthorised access to data systems.

3.8. SOCIAL MEDIA: TERMS OF ACCEPTABLE USE

- content created to deceive others whilst generating a financial or personal benefit to the detriment of a third party or entity.
- repeatedly contacting someone in a manner that is targeted and unwanted.

Researcher Compliance

- The accounts used in this research were:
 - used responsibly and only communicated with each other.
 - used to search for and view publicly available content.
- The accounts used in this research were **not used to**:
 - promote content that incites or facilitates threatening behaviour/violence.
 - initiate unwanted or malicious contact with publicly accessible user accounts or groups.
 - initiate unwanted or malicious contact with private user accounts or groups.
 - promote content designed to deceive or defraud other users.
 - engage in purposes other than the intended use described in **Chapters 4 and 5**.

For extracts of applicable policies, please see Appendix C.1.

3.8.2 Google “The Terms”

According to Google [2020c], in order to use Google Drive, users must accept the Google Terms of Service, and the Google Drive Additional Terms of Service. Both documents are collectively referred to as “The Terms” and confirm a set of reciprocal expectations between users and Google.

Google Terms of Service

The Terms of Service are grouped under the following topics:

1. Your relationship with Google
2. Using Google services
3. Content in Google services
4. Software in Google services
5. In case of problems or disagreements
6. About these terms

Google grants users permission to use there services provided that these and any service-specific terms [Google, 2020d] are adhered to. Examples of the basic rules of conduct specified by Google include:

3.8. SOCIAL MEDIA: TERMS OF ACCEPTABLE USE

- User compliance with “applicable laws, including export control, sanctions and human trafficking laws.”
- Users are required to “respect the rights of others, including privacy and intellectual property rights.”
- Users are not engage in threatening or harmful behaviour.
- Users are not to “abuse, harm, interfere with, or disrupt the services.”
- Users are to ensure they have the necessary rights to upload and share content and that the content is lawful.
- Users are to ensure that they meet the minimum age requirements to own and manage a Google Account, and any service-specific age requirements [Google, 2020b].
- Users are not to copy, modify, distribute, sell or lease any part of the services or software.
- Users are not reverse engineer or attempt to extract any of the source code without written permission from Google or an applicable law allows this.

Google Drive Terms of Service

These terms, also referred to as “Google Drive Additional Terms” are grouped under the following parts:

1. **Your Content:** which affirms a user’s ownership and control of content uploaded, shared or stored in a Drive account. It also reiterates Google’s “limited purpose license to operate and improve the Google Drive services” as described in the Terms of Service [Google, 2020e].
2. **Program Policies:** which is broken down into 20 categories and describes what is deemed appropriate behaviour and use of the platform.

The program policies are applicable to Drive, Docs, Sheets, Slides, Forms, and new Sites. They aim to prevent abuses that threaten Google’s ability to provide these services to users [Google, 2020a].

In some cases, Google may make some exceptions based on “artistic, educational, documentary, or scientific considerations, or where there are other substantial benefits to the public from not taking action on the content.”

Examples of prohibited behaviour under the program policies include:

- accessing another user’s account without permission.
- creating, uploading, or distributing content that exploits or abuses children.

3.8. SOCIAL MEDIA: TERMS OF ACCEPTABLE USE

- attempts to bypass these policies or subverting restrictions placed on an account.
- not keeping an account active by using it at least once every two years.
- promoting activities, goods, services, or information that could cause serious harm to others.
- threatening, harassing and hateful content.
- impersonation, misrepresentation and misleading content.
- explicit or non-explicit intimate images shared without the subject's consent.
- spamming, distributing malware or interfering with the service.
- storing or distributing unauthorised images of minors; or graphic content depicting violence.

Researcher Compliance

- The accounts used in this research were:
 - used responsibly and only communicated with each other.
 - used to store content created as part of this research. Content shared as part of this research was not made available to the general public.
- The accounts used in this research were **not used to**:
 - incite threatening, harassing behaviour/violence.
 - initiate unwanted/malicious contact with other users, or to access other user accounts on Google.
 - promote content that exploits children or depicts violence.
 - promote content designed to deceive or defraud other users.
 - engage in purposes other than the intended use described in **Chapters 4 and 5**.

For extracts of applicable policies, please see Appendix C.2.

3.8.3 The Twitter Rules

Twitter has a comprehensive set of rules and best practices that define what is considered acceptable or unacceptable behaviour on the platform. According to Twitter Help Center [2020], these rules prohibit unsavoury behaviour and ensures that users can freely and safely participate in the public conversation.

The rules are currently grouped under five headings which include Twitter's approach to policy development and the enforcement of these policies.

1. Safety
2. Privacy

3.8. SOCIAL MEDIA: TERMS OF ACCEPTABLE USE

3. Authenticity
4. Enforcement and Appeals
5. Third-party advertising in video content

The Twitter rules aim to prevent and protect against a wide range of activities and behaviours such as:

- Automated proactive following/un-following of other accounts, indiscriminately “following hundreds of accounts to get attention”, or unsolicited and repeated replies to multiple accounts.
- Sharing personal identifiable information about other individuals without permission.
- Targeted harassment of another user; the promotion or glorification of violence, sexual exploitation, or cruel/insensitive content.
- Impersonation and identity misrepresentation; content that is designed to deceive or behaviours designed to enable other prohibited behaviours and activities.
- Sharing content that violates someone else’s intellectual property rights.

Examples of unacceptable behaviour prohibited by Twitter’s policies include:

- content that makes violent threats against an identifiable target.
- sharing or threatening to publicly expose someone’s private information.
- posting explicit content without the subject’s consent.
- sharing content that promotes the sexual exploitation of minors; or graphic content depicting violence.
- impersonation, misrepresentation and misleading content.
- circumventing a suspension, or enforcement actions.
- interfering with computers and networks in order to disrupt services or to gain unauthorised access to data.

Researcher Compliance

- The accounts used in this research were:
 - used responsibly and only communicated with each other.
 - used to search for and view publicly available content.
- The accounts used in this research **did not**:
 - incite or promote content that facilitates threatening behaviour/violence.
 - initiate unwanted or malicious contact with publicly accessible or private

user accounts.

- promote content designed to deceive or defraud other users.
- deviate from the intended use described in **Chapters 4 and 5**.

For extracts of applicable policies, please see Appendix C.3.

3.9 Environment Setup

3.9.1 Virtualisation Vs Physical Devices

In **Section 3.6.1**, VMware was selected as the virtualisation tool for this research. VMware 9 was installed at the early stages of the research and rolling updates were subsequently applied bringing the version up to VMware 15. VMware allows the creation of full clones which are not dependent on the parent disk, or linked clones which are duplicates of the parent disk and depend on the parent disk in order to function. Virtual disk cloning although considered as an option, was not implemented.

Whilst choosing to conduct experiments in a VM, certain problems associated with the use of VMs were taken into consideration. They include possible lack of drivers for some applications [Barrett and Kipper, 2010]; the potential loss of data; contamination due to the unintentional corruption of a VM or contamination due to interaction with the host OS. To mitigate issues that may occur as a result of conducting experiments in a VM, periodic tests were carried out on physical PCs to confirm consistency with results obtained from a VM.

It was determined that preserving the state of the system after each experiment was a requirement; to meet this, a copy of the *.vmdk* file is created after each experiment and saved in a working folder. This was a way to keep track of the changes made to the system as a result of user and OS activities. It was also a useful means of comparison of artefacts recovered with the digital forensics tools listed in Table 3.2 during the manual identification processes.

With a physical disk, this process would require imaging the disk after each experiment which due to the time constraints on the research, was not a practical solution. It is also easier to copy and move an existing virtual machine to another computer (for example, from Windows 10 to Mac OS and back to Windows) unlike a physical disk which would either require a lot more effort or would not function.

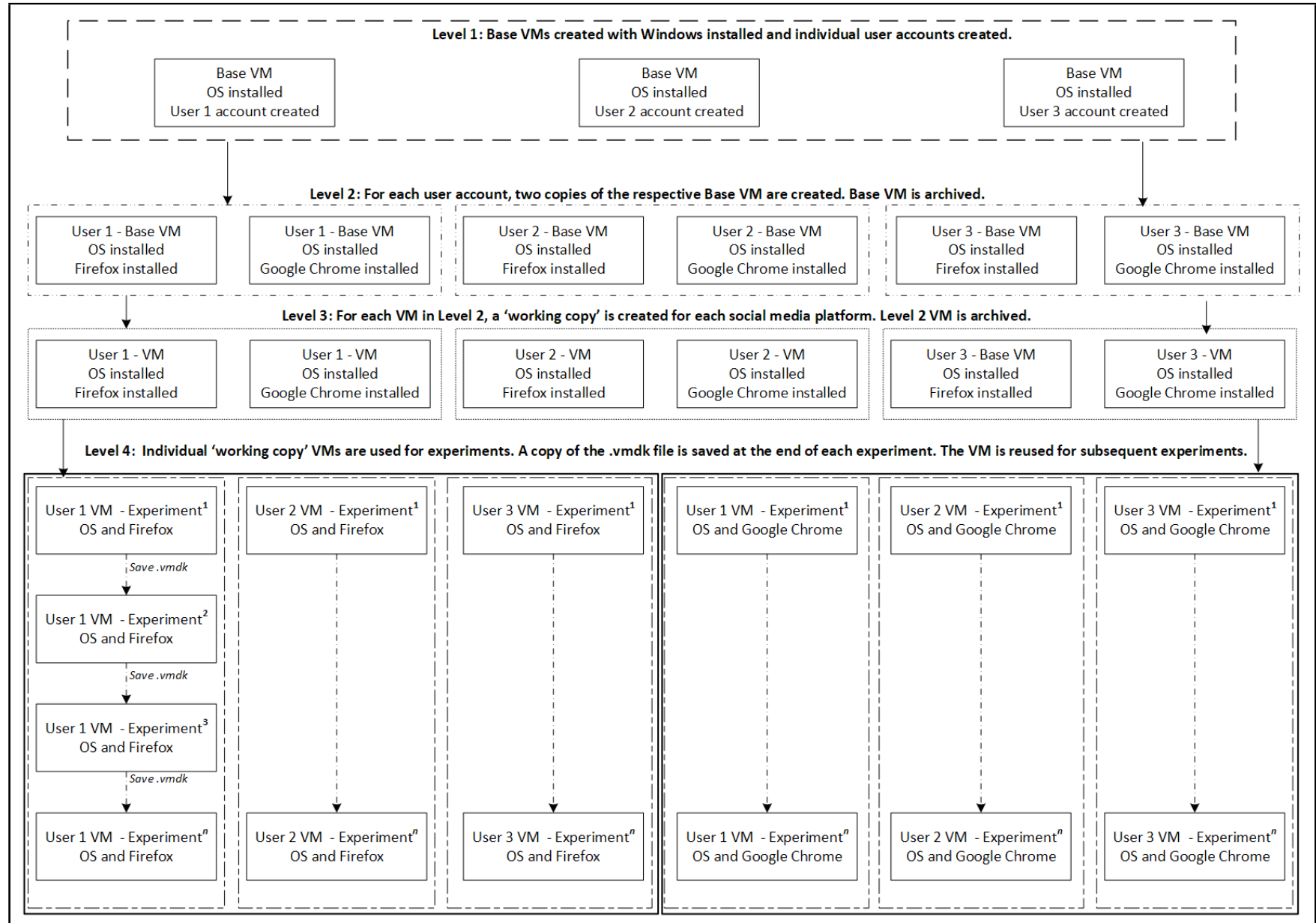


Figure 3.2: Virtual Machine Layout

3.9.2 Baseline Setup

This section discusses the creation process, structure and purpose of the VMs. Figure 3.2 is a graphical illustration of the VM structure used in this research and depicts the four main levels involved in the creation and use of the VMs.

At **Level 1**, Baseline VMs were created for three users “Josh”, “Obed”, and “Seth”. Three users were selected to reflect any noticeable differences between two individuals (i.e. one-to-one contact) and/or multiple parties communicating within the context of artefacts other than communications content. Although the dataset from three users represents a small number of relationships and does not fully represent the complexity of how users may communicate, it provides fast, reliable information which is required in a live triage.

Windows 7 was installed and an account was created for each user on individual VMs. Using the correct character set (ANSI, ASCII) and grep, the usernames were searched for using WinHex to ensure that they were not in a clean Windows 7 installation by default. This formed the basic configuration per user and produces three Baseline VMs.

To ensure consistency of the experiments and results, the VMs were not upgraded to Windows 10. However, as Windows 7 neared end of life, a Windows 10 physical computer was to be used for testing the framework. Initiating an upgrade during the experiments would involve downloading and running the Windows 10 upgrade tool; this may introduce unknown variables outside the scope of this research.

At **Level 2**, for each user account created in Level 1, two copies of the user Baseline VM are created. Copying involves a copy-paste-rename activity. Once completed, Firefox is installed on one VM and Google Chrome on the other. This process produces six Level 2 VMs. The VMs created in Level 1 are archived at this stage.

Level 3 VMs are the experimental working copies. For each VM produced in Level 2, a copy is made for experiments on individual social media platforms. For example, “*Obed VM $\Rightarrow$ Firefox $\Rightarrow$ Twitter*”. Level 2 VMs are archived after working copies are successfully created.

Level 4 or working copy VMs are the ones used actively for experiments. At the end of each experiment, the VM is powered off and the .vmdk (virtual disk) file is copied to a working folder. This is a way to keep track of the simulated real life activity, identifying what changes occurred during an experiment; and simulating

3.10. EXPERIMENTAL METHODOLOGY

the imaging a physical disk at the end of each experiment.

The VMs created from Level 3 onwards (i.e. the working copies) form the starting point for each group of experiments; i.e. the first set of Level 3 VMs (three per browser, six in total), are to be used for social media platform 1 (Twitter) for example:

- *Josh VM* $\Rightarrow$ *Firefox* $\Rightarrow$ *Twitter*
- *Obed VM* $\Rightarrow$ *Firefox* $\Rightarrow$ *Twitter*
- *Seth VM* $\Rightarrow$ *Firefox* $\Rightarrow$ *Twitter*
- *Josh VM* $\Rightarrow$ *Chrome* $\Rightarrow$ *Twitter*
- *Obed VM* $\Rightarrow$ *Chrome* $\Rightarrow$ *Twitter*
- *Seth VM* $\Rightarrow$ *Chrome* $\Rightarrow$ *Twitter*

The second set are to be used for social media platform 2 (Facebook), and the third set are to be used for social media platform 3 (Google Drive). This makes a total of 18 VMs at Level 3.

At Level 4, individual copies of the VMs are taken at the end of a range of activities as discussed in **Section 3.9.1**, to preserve the state of the system at the end of each experiment. The archived baseline VMs (Level 1), the copied VMs (Levels 2 & 3), *.vmdk* files, and extracted artefacts were stored on a Network Attached Storage (NAS) device.

3.10 Experimental Methodology

This research is multi-phased and iterative in nature thus, predicting each stage of the research is dependent on the results from the preceding stage, and these are determined after a study of artefacts recovered in each completed stage. The experimental methodology consists of two phases; phase one involves using traditional digital forensics solutions to study the artefacts, while phase two involves the application of a general automation framework as a solution.

3.10.1 Traditional digital forensics solution

A range of digital forensics tools were used to examine the disk images and to recover and study artefacts of interest. This activity was carried out with a view to establish a baseline for the identification and recovery of features of interest [**Section 3.6**].

Browser Behaviour

This stage of the methodology involves studying browser behaviour, including their default installation location, directory structure, and file formats. This stage also

involves the identification of the type (e.g. URL, image, text) and location of artefacts left behind as a result of social media activity [Section 2.3]. For instance, where data from actions such as downloads or entering information in text fields are stored.

Following existing digital forensics methods, digital evidence was recovered from images of target devices. This was done to ensure that the state of the digital device was unaltered and made it easier to focus on the identification and recovery of the data from each experiment; the identification of the components to include in an automation framework, and also to have a baseline for comparison for data recovered through automation.

Known artefacts

Known artefacts in the context of this research can be described as *artefacts that have been previously identified, and are well known within the digital forensics community (e.g. browser history artefacts), and can be attributed to specific activities*. This stage of the methodology involves the extraction of known artefacts, and provides an idea about what is expected or what can be inferred when social media platforms are used. The artefacts of interest are those from which features can be extracted, and can be used as a source of intelligence to identify users, infer actions and potentially link other users.

The recognition and extraction of features from social media artefacts would provide an understanding of the format of artefacts from different social media platforms. For example, some social media platforms may have the user ID or username in plain text as part of the URL while others may have an encoded version of the same information. Individual experimental methodologies used for the social media experiments are discussed in detail in **Chapters 4 and 5**.

Identify features of interest

This stage involved exploring related research [Section 2.4.2] where the client/server relationship of social media platforms was reviewed. Client based applications interact with the server through a dedicated application installed on the user's desktop, while web-based interaction with the server is via a compatible web browser.

In the context of this research, identifying features of interest refers to the identification of certain attributes or properties of a piece of digital evidence which may be used to infer what action has occurred [Section 2.6.4]; and facilitates the definition of features that can be used in linking entities. For example, attributes indicating the

3.10. EXPERIMENTAL METHODOLOGY

activities of a user on social media (e.g. ‘/login’, ‘/follow’, ‘/photo’).

The use of a traditional digital forensics approach is aimed at understanding the evidence; it involves interpreting and categorising features from social media artefacts, in order to provide a background for defining the scope of the research. It helps achieve the objective of characterising communications content (**Objective 1**). This stage also involves the manual identification of features from social media artefacts with focus on data of probative value [**Section 3.6.1**].

This was achieved by creating written and video logs of the URLs seen in the address bar during the experiments, as well as data from the captured traffic; then comparing the visually identified data to what had been written to disk. It was important to define a process of identifying these attributes in digital evidence through testing, to ensure consistency with previously published results [Arshad et al., 2019; Huber et al., 2011; Shaw et al., 2016]. A multi-tool approach [**Section 3.6**] is used to validate the results.

System artefacts

Due to the nature of this research, it was necessary to identify other artefacts that can be used to contextualise and corroborate social media artefacts, which are largely web browser artefacts. System generated artefacts can be useful for authenticating device ownership, and in contextualising use activity; for example, highlighting events that occurred prior to, during and post user browser activity.

Identifying potential sources of corroborating evidence involved considering several parts of the system where digital evidence may be located. This includes Volume Shadow Copies (VSC), unallocated space, registry, event logs, prefetch, and page-file/swapfile.sys. For the purposes of this research, it was decided that the analysis of some of these files may introduce additional variables outside the control of the researcher. For example, analysing VSC during a live triage may require mounting the shadow file, and depending on the number of shadow copies available, each one may need to be mounted, before the data it holds can be analysed.

Mounting the shadow copies would cause further interactions with the target system, which is at variance with a live triage (i.e. ensuring minimal interaction with the target device). Another option considered with regards to the analysis of VSC is the time it takes to process, and the volume of data it returns when a ‘strings’ search is performed. This requires time and effort in order to extract meaningful data, and defeats the purpose of a live triage.

In addition, Windows OS locks some system files (e.g. pagefile.sys) when a computer is in use, making these files inaccessible for direct analysis during a live triage. The decision was made to analyse the system artefacts which may be used for correlation. Three sets of system artefacts (Windows registry, event logs, and prefetch) were considered of interest due to the type and value of artefacts they contain; and for the purposes of this research would be described as context artefacts.

It is important to note that with the registry and prefetch, evidence of browser installation may not always be immediately accessible, depending on how long the system has been in use. It may have been overwritten, or the First In First Out (FIFO) method may have been applied, especially for prefetch; however, registry keys may still exist after an application has been uninstalled.

Windows Registry Artefacts

Microsoft describes the registry as *“a hierarchical database that contains data that is critical for the operation of Windows and the applications and services that run on Windows”* [Microsoft, 2018]. The registry consists of hive files which store data in key:value pairs.

Read, write, and delete are three basic functions or operations that occur at a system (or file) level on a computer. Registry keys are opened or created, queried, written to or deleted during OS or user activity. Figure 3.3 is an illustration of a sample of registry operations depicting user activity. For example, when a user executes an application such as the installer file for a web browser, a registry key is created for the application. The data from the registry is considered valuable as it can be used to determine what browsers are installed, the default browser, frequency of use, and whether a browser is being accessed from a removable device.

To explain the registry operations in Figure 3.3 at a high level:

1. A user runs an executable file, the **RegOpenKey** function is called. If a key for the said application already exists, it is opened.
2. If a key does not exist, the **RegCreateKey** function is called to create a key for it, and open the key.
3. When the key is opened, the **RegQueryKey** and **RegQueryValue** functions are used to query, and retrieve key:value information for the specified (open) registry key.
4. The **RegEnumKey** and **RegEnumValue** functions are used to enumerate the data stored in the key (subkeys and their values). **RegEnumKey**

3.10. EXPERIMENTAL METHODOLOGY

retrieves the name of one subkey each time it is called; while **RegEnumValue** retrieves one indexed value (name and data block), for each associated subkey each time the function is called; This continues until there are no more keys, subkeys and values to be returned.

5. At the end of a transaction (activity), the **RegCloseKey** or **RegFlushKey** function can be used to close the key.
 - **RegCloseKey**, may take several seconds to flush the information in the transaction logs (cache) to disk.
 - **RegFlushKey** is used by an application to explicitly write registry information to disk, and is resource intensive.

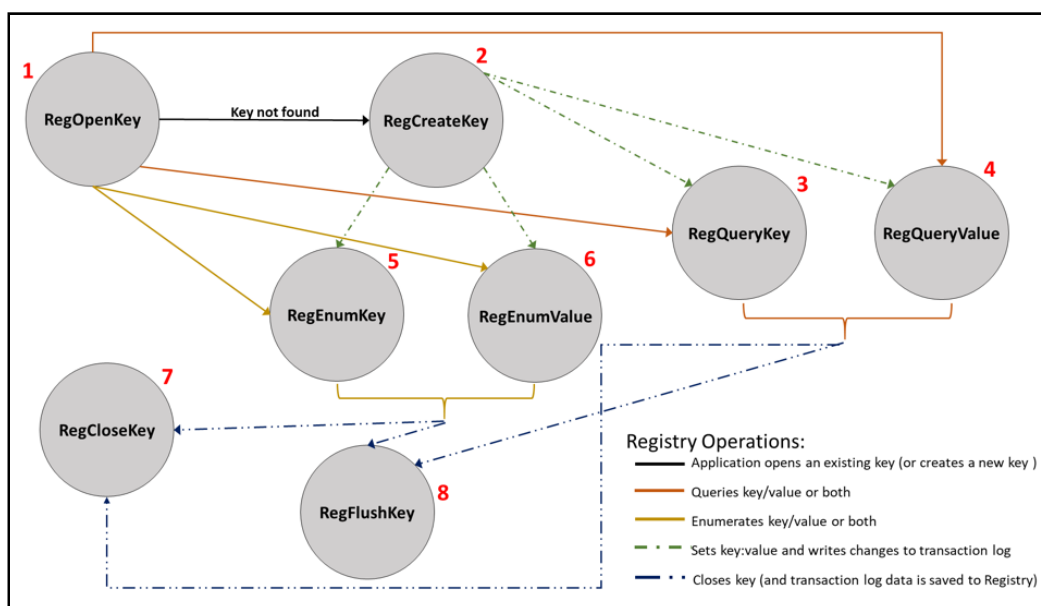


Figure 3.3: Example illustration of Registry operations create, open, query, enumerate key activity (based on data from Procmon).

With regards to social media activity, evidence recovered from the registry was focused on determining system information; user local account information (login, logout, installed applications, etc). This type of information is of forensic interest in a live triage investigation, as it will help an investigator authenticate device or local machine account ownership and access; determine possible social media account ownership, and user's behaviour on the device. It would also highlight that a user may have multiple web browsers installed; helping an investigator focus the search for browser artefacts.

At the time of writing, there is limited information on the full structure of the registry available from Microsoft. Open source developers, researchers, and digital forensics practitioners have made some information available [Carvey, 2011; Morgan, 2009];

these can be used to identify and recover information/artefacts of interest from the registry. Windows Registry activity in the context of this research is discussed in **Chapter 6** and **Appendix B**.

Event Logs

Event logs are a record of system, security, application activities automatically recorded by the Windows OS without manual intervention [Microsoft Windows Dev Center, 2018]. Event logs are located in %SYSTEMROOT%\Windows\System32\winevt\Logs and have an .evtx extension on post-Windows XP computers. Windows OS identifies, and categorises events by assigning an Event ID, and other descriptors [Microsoft, 2012; Ultimate IT Security, n.d.] that enable investigators determine the usability of the event log artefacts.

Event Logs contain a wealth of information that can be of significant value to a digital forensic investigation. Depending on the nature of the investigation, the information retrieved, ranging from audit events (logon/logoff); setup logs (system drivers and applications); application events (application access/usage/errors) etc. may be used to scope an investigation and to corroborate case specific artefacts. For example, logon/logoff events may be used to corroborate events that occurred during a specific time frame.

Although the event logs contain interesting data, there is the challenge of managing the volume of data it contains in order to retrieve usable information. In the context of this research, it was necessary to recover and analyse the artefacts in the event logs. This was aimed at understanding the information it contains, and recognising the type of information that may be of interest in a live triage to corroborate other intelligence obtained.

To do this, a number of the identifiers and descriptors assigned to an event can be combined, and used to scope an investigation. For example, the *EventID*, *TargetUserName*, *TargetLogonId*, and *LogonType* may be used to group individual logon/logoff sessions. This information would be useful for specifying a time frame to focus on during the intelligence gathering process of a live triage investigation because all access is recorded including system uptime, start and shutdown times.

Prefetch

Windows uses pretetching to speed up application start up processes, and improve user experience. It is a technique the Windows OS uses to identify and preload frequently used applications into memory. Information on the applications accessed by a user

3.10. EXPERIMENTAL METHODOLOGY

can be saved in a number of files in the prefetch folder [Magnet Forensics, 2014], which can be found by default in %SYSTEMROOT%\Prefetch. Prefetch files use the following naming convention: `executablename-hash.pf` (e.g. `FIREFOX.EXE-E60C0AA7.pf`). The hash in the filename is a hash of the path from which the application was executed.

To determine if prefetch is enabled on a computer, the value of `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\PrefetchParameters` registry key can be used. The possible values of this key are: 0 = Prefetch is Disabled; 1 = Application Prefetching is Enabled; 2 = Boot Prefetching is Enabled; and 3 = Application and Boot Prefetching are Enabled.

A prefetch file is created the first time an application is executed, and subsequent use of the application would increment the counter that records the number of times the application has been used. The quick and useful information provided by prefetch files, makes their examination a common practice in live investigations involving malicious programs/activities [Porter, 2014]. Prefetch tracks program execution and may retain records of applications that are no longer present on the device; for example, if a ‘clean up’ or ‘wipe’ tool has been used to remove certain artefacts of interest.

The volume of information recovered from prefetch can be overwhelming however, in the context of this research, the prefetch files of interest are those that can be used for corroborating the social media artefacts i.e. the browser(s) installed, last used time and frequency of use; and also to determine if the evidence recovered from disk during a live triage is consistent with what the investigator expects to find. For example, when a browser is updated, what are the potential implications to a digital forensic investigation. Prefetch artefacts are also useful in highlighting instances where a secondary storage device is being used as a source for launching applications.

The outcome of every stage in this methodology made it possible to identify an appropriate path forward in the research; in a sense, a hop-like technique is implemented where a process is completed, then analysed, and the results are used to determine, inform and build on the next stage.

3.10.2 Framework conceptualisation

Having identified the artefacts of interest, it was necessary to devise a means of recovering these during a live triage. Manual processes are time and resource intensive,

and unsuitable for live triage investigations. Live triage often employs the use of automation methods such as scripts, tools and procedures for extracting, analysing, and reporting these as part of an intelligence gathering process [Section 2.6].

The framework developed as part of this research takes individual groups of artefacts as components; made up of core, case-relevant, and corroborating evidence. The framework will be used to parse the data on disk, and recover evidence related to specific framework components. At the completion of the PoC framework design, it was tested in a live environment as a means of demonstrating its viability.

The development of a PoC framework enables the retrieval of previously identified ‘known’ features through a live triage. The development process involves determining the framework structure; the components that make up the framework, and why they are required; and the programming language that can be used to develop a PoC to test the theoretical framework. The methodology used in developing the framework is discussed in Chapter 6.

3.11 Summary

This chapter presented an overview of the problem definition for the research. It discussed the research question and the objectives, and defined the key tasks required in order to meet the objectives, and address the research question. It highlights the importance of context and corroboration when establishing a link between the evidence and a suspect. It discussed the methodology and tools used for the analysis of experimental data, and provided the rationale for experimental sample size [Sections 3.7; 3.9.2]. It also introduces the framework conceptualisation process.

It highlights a requirement for a solution that provides investigators with the required intelligence in a timely manner. It offers a high level overview of the artefacts relevant to the case, suggesting the possible inclusion of live triage into the initial stages of a forensic investigation, as a means of mitigating the challenges with the recovery of evidential data identified as part of this research. This chapter also provides an overview of the framework developed as part of this research, as a solution for investigators undertaking live triage.

The next chapter discusses the analysis and results of social media experiments conducted in Firefox.

4 Firefox: Social Media Artefacts

4.1 Introduction

This chapter discusses the methods used for the experiments conducted using Firefox. It discusses the individual experiments on Twitter, Facebook and Google Drive. It discusses the analysis strategy and the experimental results. The results presented in this chapter provide insight on the artefacts that are of interest in an investigation involving social media activity.

In this chapter, the following font conventions are used:

Table 4.1: Font conventions

Item	Font typespace
Paths and Filenames	<code>monospaced typewriter</code>
SQLite database table names	<i>slanted text</i>
Artefact and Hex extracts	typewriter
Quotes and definitions	<i>italics</i>

The rest of this chapter is structured as follows: **Section 4.2** discusses the experiments conducted on Twitter, the analysis of the data, and the results. **Section 4.3** discusses the Facebook experiments, analysis, and results. The Google Drive experiments are discussed in **Section 4.4**, followed by the analysis and results. **Section 4.5** discusses how these experiments address the research question; followed by an overview of the results from Firefox in **Section 4.6**. **Section 4.7** contains the chapter summary.

4.2 Twitter

This section discusses the experiments and results from Twitter activity using Firefox. The results discussed in this section are from the manual recovery processes, using a range of the traditional digital forensics tools discussed in **Section 3.6**.

4.2.1 Experiments

This section details the data generation and collection process. Experiments were conducted using each of the three user accounts created on the experimental VMs [Section 3.9.2]. The experiments for Twitter consists of a range of activities that were carried out to simulate user social media activity; and the actions described

in this section were repeated for each user on Firefox. In addition, as discussed in **Section 3.6.1**, Procmon was used to log system activity during the experiments, while Fiddler was used to capture HTTPS traffic.

Mozilla Firefox is a free and open source cross platform web browser, and as at 2019 was the second most popular browser (See: GlobalStats statcounter [2019]) based its market share. Firefox uses a variety of SQLite database files to store user browser information [Pereira, 2009]. Due to the rapid changes and releases in Firefox versions, rolling upgrades were applied to simulate real-life user activity. The new versions were reviewed to identify any changes that could be of forensic relevance or impact on the research. For example, the addition of new tables to the SQLite database, removing a database or tweaking functions.

In its Frequently Asked Questions (FAQs) for new users, Twitter describes how the platform can be used [**Sections 3.7; 3.8.3**]. Based on this description, basic activities that can be carried out on Twitter can be extrapolated and placed into categories for the experiments. These categories (Registration and Authentication; Searching and Viewing Content; Sharing Content; Following) are representative of one of the varying ways Twitter can be used.

Registration and Authentication

A registered account is required in order to get the full benefits of the platform; i.e. the ability to connect with, and exchange messages with other users. To simulate the account registration process, Twitter accounts were set up for the Windows users created for this experiment [**Section 3.9.2**]. Twitter account creation requires a valid email address and/or a mobile phone number. Gmail accounts were created for use in the verification of the social media accounts set up for this research. Creating a Twitter account allows a user to personalise the account by adding profile photos and banners, and amending the account settings. This experiment will highlight artefacts created as a result of account registration on Twitter.

Searching and Viewing Content

During the experiments, it was observed that unregistered users are able to visit publicly accessible Twitter profiles via search engines; they are then able to search for other registered users, view trending topics and public content posted on Twitter. Public content in the context of this usage scenario refers to any content posted on Twitter accounts that are open to the public; for example, TV channels, animated characters, etc.

4.2. TWITTER

To simulate this activity, searches were made on Twitter for public user accounts, including those of the additional users created during this experiment. This activity also involved navigating through Twitter by clicking tabs like sports, news, entertainment etc., and viewing the content on those pages. This experiment will highlight the artefacts created as a result of searching and viewing content on Twitter; this covers the activities of registered (including registered, but not logged in), and unregistered users.

Sharing Content

Content dissemination is another way Twitter is used. This is primarily achieved through “Tweeting” or “Retweeting” using the Tweet button or the retweet button, depending on what the user is hoping to achieve. A Tweet is described as “*any message posted to Twitter which may contain photos, videos, links, and text*” [Twitter Help Center, 2019]; while a Retweet is forwarded, and reshared Tweet. As part of content sharing, users may choose to exchange direct messages known as DMs; this is a way of communicating privately. Tweets can also be sent with hashtags to highlight trending topics; when written, keywords are prefixed with the hash symbol (#) for example #haircare, #freebies.

It is important to note the following: At the time of writing, Tweets have a limit of up to 280 characters, once posted, they cannot be edited but can be deleted. Content sharing on Twitter is restricted to registered users only; individuals who access Twitter without an account cannot Tweet, Retweet or comment on Tweets. For this experiment, Tweets were exchanged between the experimental user accounts.

Due to the numerous ways in which Twitter can be used, it was necessary to select and simulate a subset of activities as described in Twitter’s “New User FAQs”. This was done to establish a baseline for how this social media platform can be used. For this experiment, the test user accounts were used to post Tweets on own profile, post on another test user’s profile, reply Tweets, and share content through DMs.

Following

Following, according to Twitter, means choosing to subscribe to another user’s Tweets such that whenever they post an update, it shows up on the subscribing user’s profile [Twitter Help Center, 2019]. Followership is designed as a two way process with users following each other. However, there are scenarios where a user does not follow a subscriber back; for example, a football club with millions of followers. For this experiment, the test users sent follow requests to each other, and also followed public Twitter accounts (TV channels, comics strips). This was done to determine user

social relationships and associations on Twitter.

4.2.1.1 Availability and Recoverability of Artefacts

During the experiments, it was necessary to pause the VM and copy the virtual disk to a working folder [Section 3.9.1]. This activity was carried out to preserve the state of the VM during the experiment; this essentially puts the VM on hold, and allows the experiment to continue without the need to restart services, applications and sessions. It was also a way to compare what was written to disk during live activity, with what was recovered once the VM had been powered off.

Due to the nature of user social media activity, there were multiple use cases or scenarios that could occur, and were considered. However, the outlined activities were selected as a baseline, due to the time constraints that would be imposed on the research if all possible scenarios were to be covered in the experiment.

It is important to note that due to the nature of web browsers, and social media platforms as discussed in Section 3.2, it was necessary to verify the availability, recoverability and validity of previously recovered artefacts by repeating a range of the outlined activities at the end of the research.

4.2.2 Results

This section contains the results for Twitter activity using Firefox. These results can be regarded as case-specific artefacts inferring user activity. The results were recovered using a range of the digital forensics tools discussed in Section 3.6.

Browser History

WinHex was used to extract the Firefox profile folder (for the History); in Windows versions XP onwards, this is located by default in: %APPDATA%\Mozilla\Firefox\Profiles; where %APPDATA% is the environment variable name for the user's AppData (Roaming) directory, i.e. %SYSTEMROOT%\Users\<username>\AppData\Roaming. This folder contains a subdirectory with an auto-generated, 8 character, alphanumeric SALT prefix [Mozilla, n.d.] with a *.default* extension.

Firefox creates individual profile folders for each Windows user where Firefox is installed and accessed; it was also observed that additional profile folders may be created for a user in the event of a refresh of a Firefox profile, or the installation of a Release version. From Firefox 67+, multiple profiles may also exist where a user has different versions of Firefox installed e.g. Beta, ESR, Developer Edition [Mozilla Support, n.d.]. Examples of profile folder names include:

4.2. TWITTER

- tt3cd5j0.default
- ch47urgp-[timestamp].default (created when a Firefox profile is Refreshed)
- 4bpkj23t.default-release

Artefact recovery from the browser history was aimed at determining the type of information available, and how it can be recovered in a live triage. The analysis of browser history data highlights the type of artefacts that are of interest to this research. In order to analyse the generated data, individual tools were used for specific data types.

From history profile, URL artefacts were recovered from the SQLite database files using DB Browser for SQLite. As discussed in **Section 2.6.4**, attributes or properties from URLs can be used to give context to, or infer user action. In addition to the SQLite database files, the history profile directory also contains other files of interest containing session information, login credentials, etc. as shown in Figure 4.1.

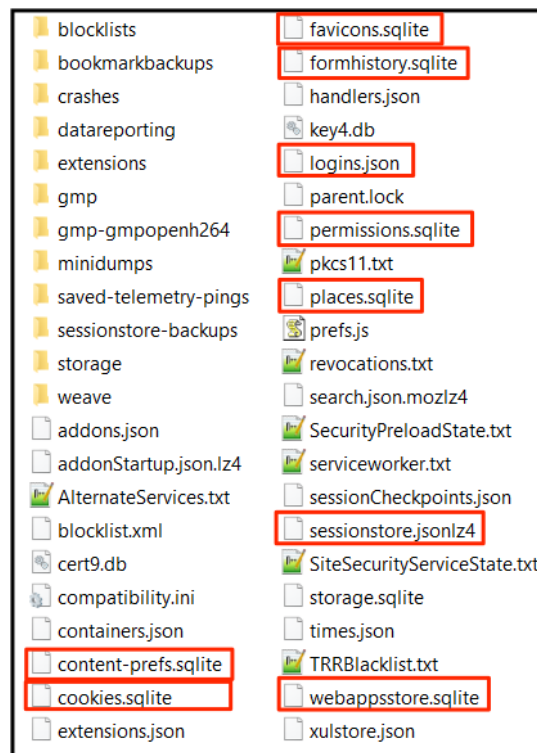


Figure 4.1: Contents of the Firefox History Profile directory

The highlighted files in Figure 4.1 were of interest in this research, and the data they contain were analysed. The goal is to identify properties (features) that will be useful in inferring user action, and in relationship attribution during a live triage investigation. The SQLite databases hold data of interest and forensic value. These

are briefly discussed below:

content-prefs.sqlite

At the time of writing, this database contains three tables; *groups*, *prefs*, and *settings*; and stores user settings for websites. The data contained in **content-prefs.sqlite** can be used as an indicator of, and to corroborate website visit data found in **places.sqlite**.

cookies.sqlite

When a user visits a website, some textual information is stored on the user's computer by Firefox, on behalf of the website. This information stored is known as a cookie, and is usually for the benefit of the website. It helps a website remember a user's login state, identify a user on subsequent visits, and provide personalised content to users.

Firefox stores cookies in the *moz_cookies* table; the data of interest can be found in the host, and associated timestamp columns. Data was recovered using the following query:

```
SELECT datetime(creationTime/1000000, "unixepoch") as createTime,
datetime(lastAccessed/1000000, "unixepoch") as lastAccessedTime,
datetime(expiry/1000000, "unixepoch") as expiryTime,
name, value, host, isHttpOnly, isSecure
FROM moz_cookies
WHERE lastAccessed != ""
AND moz_cookies.host LIKE "%twitter%"
ORDER BY moz_cookies.lastAccessed;
```

It is important to note that cookies are not necessarily an indication of a page visit, as some cookies may be created as a result of third-party advertising on websites; other corroborating evidence should be used to establish the forensic value of a cookie.

favicons.sqlite

This database contains information about the icons for a user's Favourites/Bookmarks. It consists of four tables; *moz_icons*, *moz_icons_to_pages*, *moz_pages_w_icons*, and *sqlite_stat1*. Manual inspection and analysis identified *moz_pages_w_icons* as of interest. Although Twitter was not marked as a favourite, or saved as a bookmark; when analysed, the 'page_url_hash' column in the *moz_pages_w_icons* table contained URLs related to the activities conducted during the experiment. For

4.2. TWITTER

example, searching for users, clicking on notifications, amending account settings, and viewing the profiles of other users.

This type of information can be used to corroborate other history related information, inferring a user's activities within Twitter. Extracts of artefacts recovered from *moz_pages_w_icons* can be seen in Figure 4.2.

page_url	
https://twitter.com/i/moments	navigation and content view
https://twitter.com/c...h	profile view
https://twitter.com/settings/account	updating account settings
https://twitter.com/following	navigation and content view
https://twitter.com/Dilbert_Daily	profile view
https://twitter.com/i/notifications	navigation and content view
https://twitter.com/?logged_out=1&lang=en	
https://twitter.com/search?q=g...%20c...&src=typd	typed searches for users
https://twitter.com/search?q=c...%20c...k&src=typd	
https://twitter.com/search?q=c...s&src=typd	
https://twitter.com/i/flow/signup	
https://twitter.com/i/moments?category_id=2	navigation and content view
https://twitter.com/DailyDilbert	profile view
https://twitter.com/settings/safety	updating account settings

Figure 4.2: Extracts from *moz_pages_w_icons* indicating user activity on Twitter

formhistory.sqlite

This database contains two tables; *moz_deleted_formhistory*, and *moz_formhistory*. The *moz_deleted_formhistory* table did not contain any data at the time of the analysis, so the analysis was focused on the *moz_formhistory* table.

It maintains the autofill/autocomplete data used in Firefox and stores the text entered in the search bar, or in forms on web pages. This data is stored in sets of name value pairs, so that when such web pages are revisited, a user is shown options of previously entered values when making an entry e.g. login credentials (email address and password pair).

When the results were analysed, they appeared to corroborate the activities carried out during the account registration and authentication experiments [Section 4.2.1] and the data recovered matched the data entered during the experiments. The data of interest was found in the fieldname, value, firstUsed, lastUsed, and the timesUsed columns, where:

- Fieldname contains the name of the field in a form filled out by the user e.g. username
- Value contains the entry made by a user

- FirstUsed contains the date and time of first use
- LastUsed contains the date and time an entry was used last
- TimesUsed is a counter indicating the number of times an entry has been used

Figure 4.3 is an extract from *moz_formhistory* indicating user login credentials in recovered from *formhistory.sqlite*.

fieldname	value	timeFirstUsed	timeLastUsed	timesUsed
Username	s [REDACTED] 6	19/08/2018 08:26	19/08/2018 08:32	2
Passwd	d [REDACTED]	19/08/2018 08:26	19/08/2018 08:32	2
ConfirmPasswd	d [REDACTED]	19/08/2018 08:26	19/08/2018 08:32	2
session[username_or_email]	s [REDACTED] 6@gmail.com	01/12/2018 17:20	16/05/2019 20:21	2

Figure 4.3: Extracts from *moz_formhistory* user details (credentials) for Twitter

permissions.sqlite

This database consists of two tables; *moz_hosts*, and *moz_perms*. It holds information about the permissions granted to websites. For example, permission to set cookies, open pop up windows or display notifications. This information can be recovered from the origin, and type columns in the *moz_perms* table, where origin contains the URL and type indicates the type of permission granted e.g desktop-notification. These results corroborates the user granting Twitter permission to display notifications during the experiments.

places.sqlite

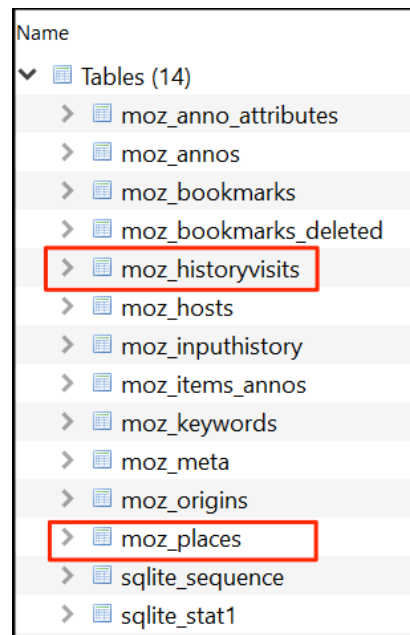


Figure 4.4: Tables in the *places.sqlite* database

4.2. TWITTER

This database contains fourteen tables; *moz_anno_attributes*, *moz_annos*, *moz_bookmarks*, *moz_bookmarks_deleted*, *moz_historyvisits*, *moz_hosts*, *moz_inpuhistory*, *moz_items_annos*, *moz_keywords*, *moz_meta*, *moz_origins*, *moz_places*, *sqlite_sequence*, and *sqlite_stat1* as shown in Figure 4.4.

Places.sqlite is used by Firefox to store information such as browsing history, input history, and bookmarks [MDN, 2019]. For the purposes of this research, the *moz_places* and *moz_historyvisits* tables are considered of interest as they contain visited URLs, with dates and times of visit, and the number of times the web page was visited. Website visit information is stored within the *moz_historyvisits* table while associated URL information is stored within the *moz_places* table.

Due to the volume of information stored in these two tables, a targeted query was used to recover Twitter specific information:

```
SELECT datetime(visit_date/1000000, "unixepoch") as visitDate,  
url, title, rev_host, from_visit, visit_type, visit_count  
FROM moz_places, moz_historyvisits  
WHERE visit_date != ""  
AND moz_places.url LIKE "%twitter%"  
ORDER BY visit_date
```

The recovered artefacts were exported to a CSV file. This was done to make data filtering and ordering easier, when identifying properties that could be used for relationship attribution. This also made it easier to corroborate these artefacts with other data recovered from aforementioned SQLite databases. The results from *places.sqlite* appeared to corroborate the activities carried out during the experiment and validates the data recovered from *favicons.sqlite*.

webappsstore.sqlite

The *webappsstore.sqlite* database contains a single table, *webappsstore2*. This database retains web storage data (similar to cookies but larger in size) for websites [MDN, 2019]. The data is stored in key:value pairs; and often persists in the event of a browser crash, or after the browser history/cookies have been cleared. The *originKey*, *key*, and *value* columns were of interest, where:

- *OriginKey* contains the reverse hostname, protocol and port number of the website storing the data (e.g. *moc.rettiwt.https:443*)
- *Key* contains the key set by a website
- *Value* contains a value assigned to a key by a website; the value varies and could be a string or an integer

It was interesting to note that the value for the `__typeahead__userHash` key contained information about the experimental user accounts, and that of other Twitter users that were not part of this experiment (> 500 were identified). Due to GDPR, security, and ethical concerns, this data has been excluded from the research results.

An extract of the type of information that may be recoverable from *webappsstore2* is shown in Figure 4.5. It also depicts data recovered from the `__typeahead__userHash` key for one of the experimental user accounts. It is important for an investigator or analyst to identify the potential availability, and evidential value of these types of artefacts; they provide information that may be used for determining social relationships and associates, identifying who is following whom. These results also corroborate the follow activities initiated during the experiments.

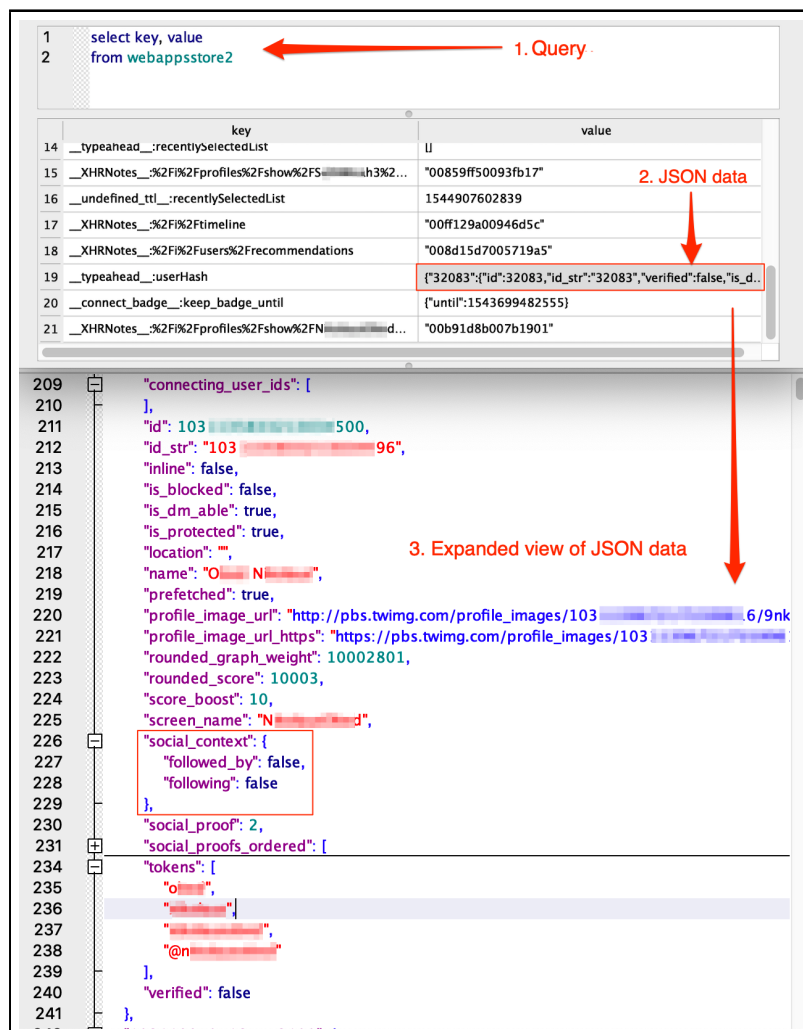


Figure 4.5: Extract from `webappsstore.sqlite` shows account information for user ‘Obed’

Due to the volume of data recovered during the analysis, features such as the

4.2. TWITTER

@user_handle, *user_id*, and *profile_id*, extracted from `places.sqlite` or other relevant databases, were used to narrow down the results to manageable, and usable size. For the purposes of determining a user's connection to additional profile/account information recovered during an investigation, the values within "social_context" as seen in Figure 4.5 may be used to infer a relationship.

sessionstore.jsonlz4

From Firefox 56+, the sessionstore file format was changed from .js to Mozilla's version of LZ4 compression (.jsonlz4). This file is used to store browser session data, including any open tabs or windows, page views, and state (minimised, maximised). It can be used to restore a browser session in the event of an unexpected termination, or a forced restart after an update. In the event of a user initiated browser exit, it can be used to reopen previous windows and tabs when the browser is restarted.

The `sessionstore` file contains a lot of information about a user's activity. To recover this data, WinHex was used to extract the file, and the *python-lz4* module was used to decompress the file, and convert it to JSON. This was done to make session data accessible, and ready for analysis. The contents of the decompressed file was subsequently viewed using JSONViewer [Hay, n.d.].

The decompressed data is stored in key:value pairs, and has some of the following extracted fields which contain artefacts of interest: page URL, page title, referrer URL where available; the time a tab or window was opened, accessed or closed; session start/last updated time, and cookies associated with the session. These artefacts appear to corroborate activities carried out during the experiments such as visiting user profiles on Twitter, and viewing followers. Figure 4.6 is an extract of the data of interest recovered (line breaks inserted).

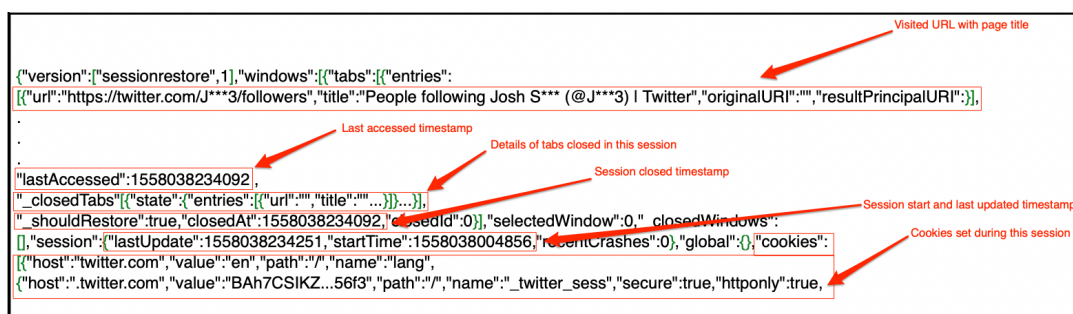


Figure 4.6: Extract from `sessionstore` infers an action was taken to view the users following Josh

When analysed further, the data in `sessionstore` was used to corroborate the user activities inferred by the other artefacts recovered from the SQLite databases. For

example, the visited and favicon URLs, and cookies. Some Tweet content was also recovered from `sessionstore` as shown in Figure 4.7.

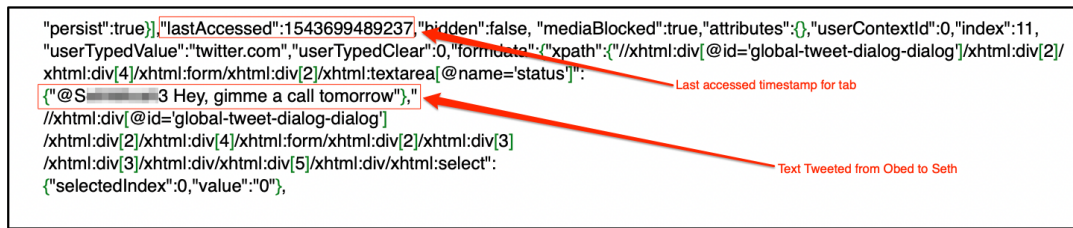


Figure 4.7: Tweet content recovered from `sessionstore`

A keyword search was carried out using the simultaneous search function in WinHex. As discussed in **Section 3.6.2**, the keyword search will only return hits that are visible to it. With this in mind, and in other to aid the characterisation of communication content, it was deemed necessary to identify other locations on disk where the content exchanged or viewed on Twitter may be found. The extracts of messages exchanged were identified in `unallocated space` and `pagefile.sys`. The results from the keyword search corroborated the experimental activities, and is shown below:

Josh sends a Tweet to **Obed**

```

1912717296 32 61 72 65 61 5B 40 6E 61 6D 65 3D 27 73 74 61 2area[@name='sta
1912717312 74 75 73 27 5D 22 3A 22 40 4E XX XX XX XX XX tus']": "@N*****
1912717328 XX XX XX XX XX 20 48 65 79 2C 20 68 6F 77 27 73 ***** Hey, how's
1912717344 20 69 74 20 67 6F 69 6E 67 20 6D 61 74 65 3F 20 it going mate?

```

Seth replies **Obed's** request for a call [Figure 4.7]

```

3053660000 00 01 00 63 02 60 61 72 65 61 5B 40 DB 00 20 3D c `area[@Û =
3053660016 27 C5 01 FF 0A 75 73 27 5D 22 3A 22 43 68 65 65 'Å ÿ us']": "Chee
3053660032 72 73 2C 20 77 69 6C 6C 20 64 6F 21 22 2C 94 00 rs, will do!','

```

Viewed Tweet on **SYFY's** account

```

5420590224 33 68 74 74 70 73 3A 2F 2F 74 77 69 74 74 65 72 3https://twitter
5420590240 2E 63 6F 6D 2F 53 59 46 59 2F 73 74 61 74 75 73 .com/SYFY/status
5420590256 2F 31 30 33 30 38 36 31 38 39 32 33 30 35 32 38 /103086189230528
5420590272 31 30 32 35 32 65 53 59 46 59 20 6F 6E 20 54 77 10252eSYFY on Tw
5420590288 69 74 74 65 72 3A 20 22 44 65 73 70 69 74 65 20 itter: "Despite
5420590304 74 68 65 20 63 61 73 75 61 6C 74 79 20 72 61 74 the casualty rat
5420590320 65 2C 20 70 65 6F 70 6C 65 20 73 74 69 6C 6C 20 e, people still
5420590336 77 61 6E 74 20 74 6F 20 62 65 20 70 69 6C 6F 74 want to be pilot
5420590352 73 20 69 6E 20 74 68 65 20 23 53 74 61 72 57 61 s in the #StarWa
5420590368 72 73 20 67 61 6C 61 78 79 2E 22 3A 14 08 98 C3 rs galaxy.": ~Ã

```

Session data is also backed up by Firefox using the files listed below. This makes it possible for previous session activity to be examined. The backup

4.2. TWITTER

files are located in `%APPDATA%\Mozilla\Firefox\Profiles\<profilefolder>\sessionstore-backups`. The following files were identified in the `sessionstore-backups` directory:

- `previous.jsonlz4`
- `recovery.baklz4`
- `upgrade.jsonlz4-[datetimestamp]` (this backup file contains the session state before a browser version upgrade)

logins.json

This file is used to store user login information; this includes the encrypted version of the username and password. The data of forensic interest in this file in the context of this research include the hostname where the login credentials are used; the created and last used time; date when the password was changed and the number of times the credentials have been used.

This data corroborates the authentication data set when the Twitter account was created during the experiments. It can also be used to corroborate the other artefacts recovered from sessionstore (e.g. hostname, date and time of access) or formhistory.sqlite (login credentials). This data can be used, in addition to context and case-specific artefacts, to link a user to a particular Twitter account (especially if the same credentials such as an email address or username is used on multiple sites).

Browser Cache

The Firefox cache is located in a separate profile folder `%LOCALAPPDATA%\Mozilla\Firefox\Profiles` in Windows XP onwards; where `%LOCALAPPDATA%` is the environment variable name for `%SYSTEMROOT%\Users\<username>\AppData\Local\Mozilla\Firefox\Profile`. The cache directory shares the same naming convention and an identical name with the History profile directory. When a user visits a website, Firefox temporarily stores, images, scripts and other files from the visited website; this enhances efficiency and allows the browser to check if those resources are locally available (on disk), and up to date before downloading them again if that website is revisited.

The Firefox cache is made up of cache entry files in the `./cache2/entries` directory; these individual files are indexed in the `./cache2/index` file. To analyse the contents of the cache, the profile directory was extracted using WinHex, and as part of the analysis, NirSoft [2018c] MZCacheView was used to view the contents in the cache.

Based on the cached contents of interest, the cache index file was opened in WinHex and the cache name for the contents of interest were compared against the entries in the cache index file.

The artefacts of interest during the analysis were those indicative of a user's direct interaction with other test users, for example, cached profile information such as profile banners, or profile photo URLs and images. These results appeared to corroborate the actions taken during the experiments e.g. visiting profiles, searching for users etc. The cache data was also used to corroborate the artefacts recovered from webappsstore and sessionstore.

As mentioned in **Section 4.2.1**, Fiddler was used during the experiment to monitor network traffic. The HTTP request and response headers captured during the experiments were analysed, to determine the underlying activities that may have occurred during profile views. These were then compared to the data in the cache; the cache data supports the actions taken during the experiments as shown in Figures 4.8 - 4.13. It is important to note that during profile views on Twitter, the URL in the address bar does not show the picture or banner URL; only the user name is shown hence, the comparison to determine any additional artefact created as a result of profile views.

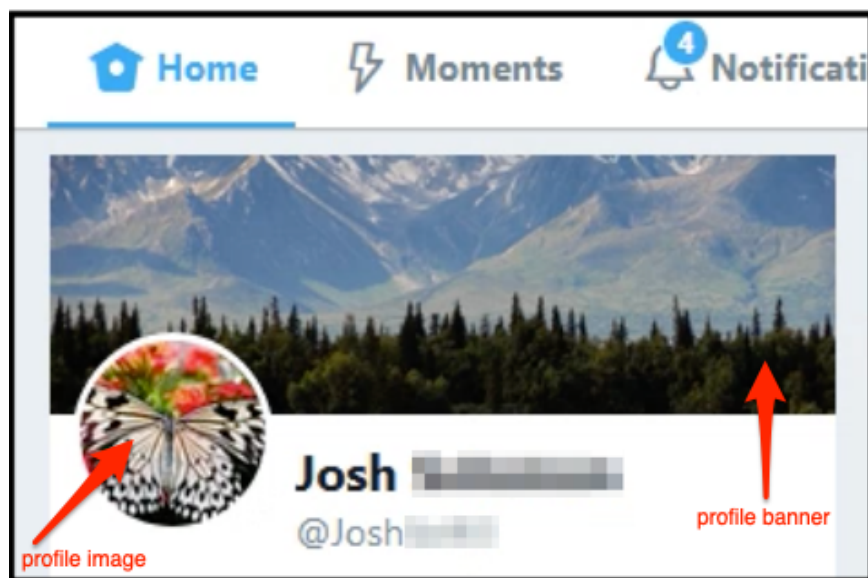


Figure 4.8: Twitter profile image and banner for test user Josh as seen on Twitter

4.2. TWITTER

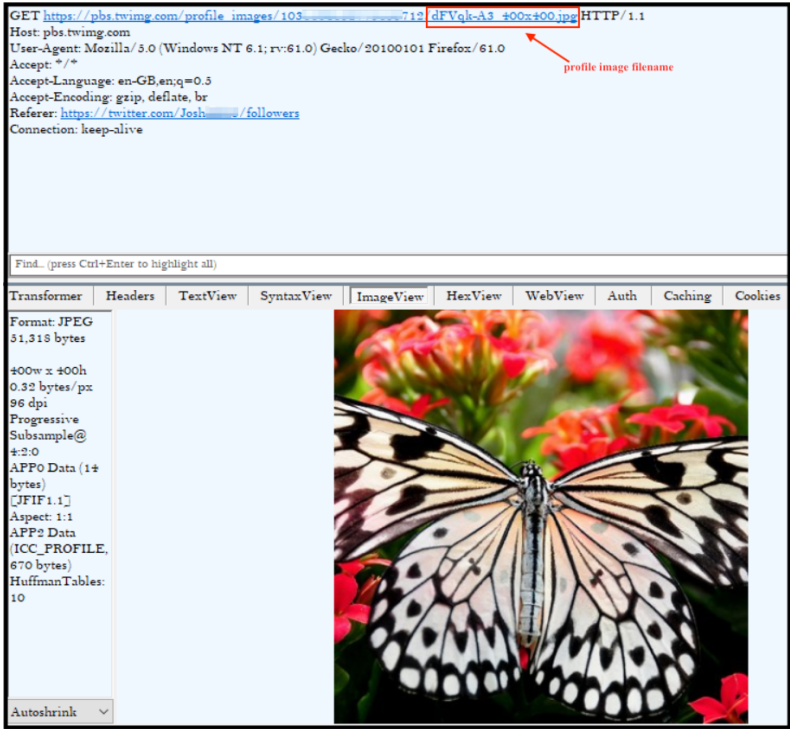


Figure 4.9: Twitter profile image for test user Josh as captured by Fiddler

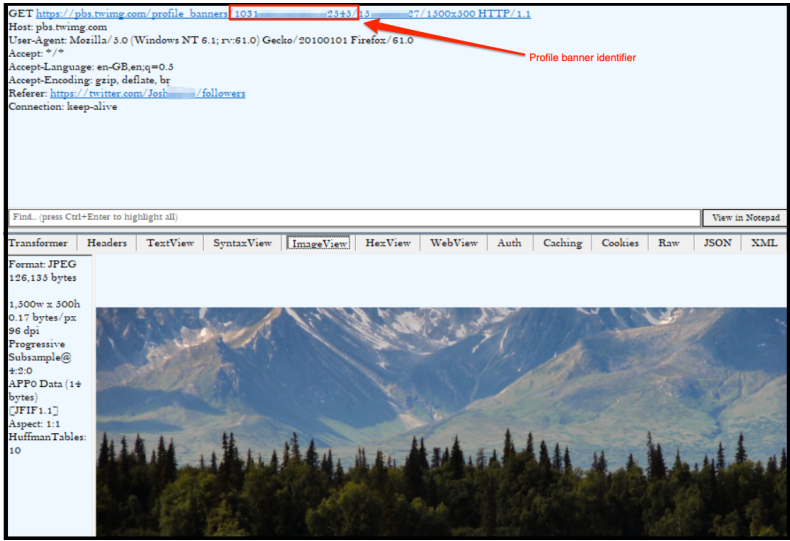


Figure 4.10: Twitter profile banner for test user Josh as captured by Fiddler

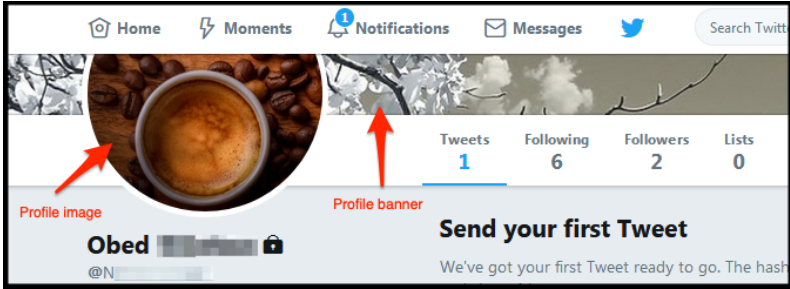


Figure 4.11: Twitter profile image and banner for test user Obed as seen on Twitter

4.2. TWITTER

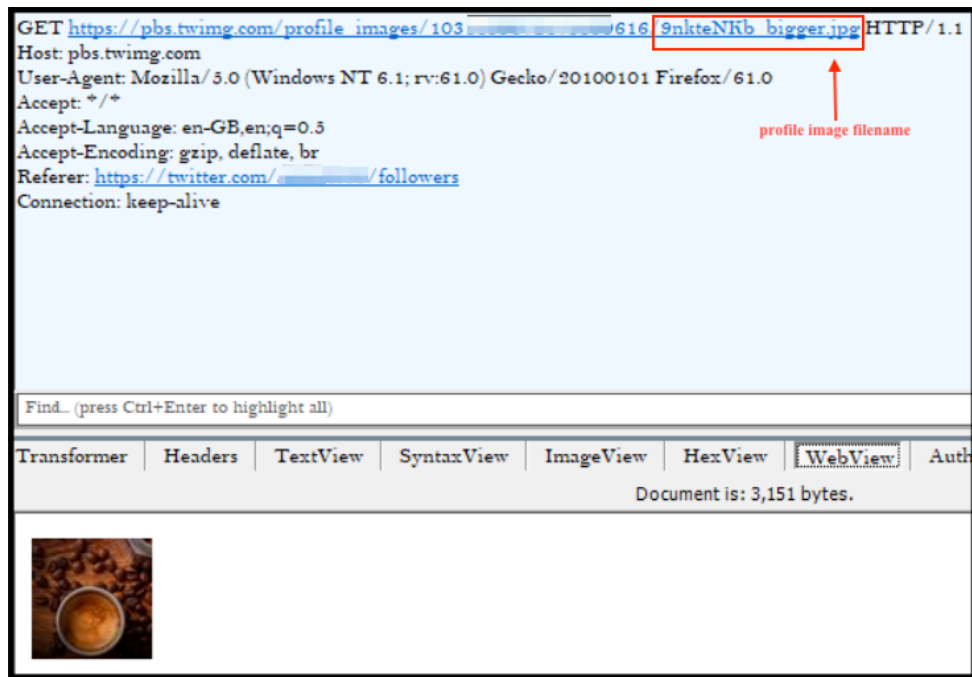


Figure 4.12: Twitter profile image for test user Obed as captured by Fiddler

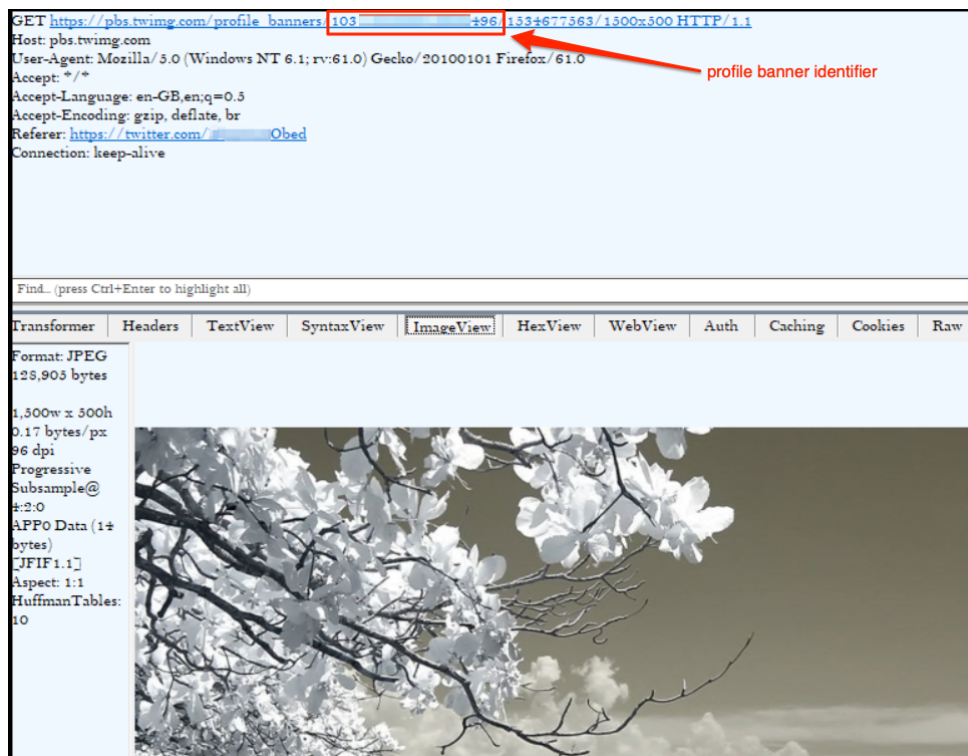


Figure 4.13: Twitter profile banner for test user Obed as captured by Fiddler

4.2. TWITTER

Based on the URLs in the HTTP request and response headers captured by Fiddler, an additional search was made in the cache, using the simultaneous search functionality in WinHex. This was done to verify the artefacts recovered using MZCacheView as shown in Figures 4.8 - 4.13, confirming the user actions recorded during the experiments.

```
00051344                                3A 68 74 74                                :htt
00051360  70 73 3A 2F 2F 70 62 73  2E 74 77 69 6D 67 2E 63  ps://pbs.twimg.c
00051376  6F 6D 2F 70 72 6F 66 69  6C 65 5F 69 6D 61 67 65  om/profile_image
00051392  73 2F 31 30 [REDACTED] [REDACTED] 37 37 35 31  s/10*****7751
00051408  35 35 37 31 32 2F 64 46  56 71 6B 2D 41 33 5F 34  55712/dFVqk-A3_4
00051424  30 30 78 34 30 30 2E 6A  70 67                                00x400.jpg

00003184                                3A 68 74  74 70 73 3A 2F 2F 70 62  :https://pb
00003200  73 2E 74 77 69 6D 67 2E  63 6F 6D 2F 70 72 6F 66  s.twimg.com/prof
00003216  69 6C 65 5F 69 6D 61 67  65 73 2F 31 30 33 [REDACTED] ile_images/103**
00003232  [REDACTED] [REDACTED] 39 36 31 36 2F 39  *****9616/9
00003248  6E 6B 74 65 4E 4B 62 5F  62 69 67 67 65 72 2E 6A  nktenKb_bigger.j
00003264  70 67                                pg

2948278064  00 00 3A 68 74 74 70 73  3A 2F 2F 70 62 73 2E 74  :https://pbs.t
2948278080  77 69 6D 67 2E 63 6F 6D  2F 70 72 6F 66 69 6C 65  wing.com/profile
2948278096  5F 62 61 6E 6E 65 72 73  2F 31 30 33 31 [REDACTED] _banners/103****
2948278112  [REDACTED] [REDACTED] 32 35 34 35 2F 31 35 33  *****2545/153
2948278128  34 36 36 39 33 32 37 2F  36 30 30 78 32 30 30  4669327/600x200
```

Other artefacts found in the cache were related to hashflags, and hashtags which had not been used by the test users. It is important to note that trending hashtags or hashflags may be cached as well even when they have not been accessed by a user. Extracts of hashflags are shown below:

```
00001216                                3A 68 74 74 70 73  :https
00001232  3A 2F 2F 61 62 73 2E 74  77 69 6D 67 2E 63 6F 6D  ://abs.twimg.com
00001248  2F 68 61 73 68 66 6C 61  67 73 2F 31 39 37 35 5F  hashflags/1975_
00001264  61 6C 62 75 6D 72 65 6C  65 61 73 65 5F 32 30 31  albumrelease_201
00001280  38 2F 31 39 37 35 5F 61  6C 62 75 6D 72 65 6C 65  8/1975_albumrele
00001296  61 73 65 5F 32 30 31 38  2E 70 6E 67                                ase_2018.png

00005088  3A 68 74 74 70 73 3A 2F  2F 61 62 73 2E 74 77 69  :https://abs.twi
00005104  6D 67 2E 63 6F 6D 2F 68  61 73 68 66 6C 61 67 73  mg.com/hashflags
00005120  2F 45 75 72 6F 76 69 73  69 6F 6E 53 6F 6E 67 43  /EurovisionSongC
00005136  6F 6E 74 65 73 74 5F 32  30 31 39 2F 45 75 72 6F  ontest_2019/Euro
00005152  76 69 73 69 6F 6E 53 6F  6E 67 43 6F 6E 74 65 73  visionSongContes
00005168  74 5F 32 30 31 39 2E 70  6E 67                                t_2019.png
```

The cache artefacts can be used to determine user relationships however, there is a caveat; profile images and banners of Twitter accounts a user has not interacted with may also be cached e.g. profiles from other users, commercial entities with sponsored content on Twitter, or details from web storage data as seen in `webappstore.sqlite`. In this context, the presence of these artefacts in the cache does not necessarily equate to a visit; to determine or attribute user relationship using cache artefacts, other artefacts should be used as corroborating evidence.

4.2.3 Summary

The research objectives were discussed in **Section 3.4**, and the artefacts discussed in this section address the key problems identified in **Section 3.3**; with regards to recovering artefacts that could be used where communication content may not be immediately accessible. **Section 4.5** discusses how this experiment meets the objectives and addresses the research question.

4.3 Facebook

This section discusses the experiments and results from Facebook activity using Firefox. The results discussed in this section are from the manual recovery processes using a range of the traditional digital forensics tools discussed in **Section 3.6**.

4.3.1 Experiments

This section details the data generation and collection process. Experiments were conducted using each of the three user accounts created on the experimental VMs [**Section 3.9.2**]. These experiments consists of a number of activities carried out to simulate user activity; and the actions described below were repeated for each user on Firefox. As discussed in **Section 3.6.1**, Procmon was used to log system activity during the experiments, while Fiddler was used to capture HTTPS traffic.

Facebook's Help Centre [Facebook Help Centre, 2019a] provides guidance on how their platform can be used by new users [**Sections 3.7; 3.8.1**]. This provides a baseline that can be used to categorise user Facebook activities for this experiment. These categories are representative of one of the varying ways Facebook can be used.

Registration and Authentication

In order to connect with, and exchange messages with other users on Facebook, a registered account is required. To simulate the account registration process for Facebook, accounts were set up for the three Windows users created for this

4.3. FACEBOOK

experiment [Section 3.9.2]. Facebook mandates an age limit of ≥ 13 years for account creation [Facebook Help Centre, 2019a] ; and a valid email address, and/or a mobile phone number is required for verification. The Gmail accounts created for the users at the start of this research, and some mobile phone numbers were used for the account verification process.

As with Twitter, a registered Facebook user can personalise their account by adding profile photos; the privacy settings can also be amended to secure the account. This experiment will highlight artefacts created as a result of account registration on Facebook.

Searching and Viewing Content

It was also observed that similar to Twitter, individuals without Facebook accounts can access Facebook, and are able to view publicly available content, however, in order to participate in the social community, an account is required. Publicly available information refers to profiles, pages, and groups for TV channels, animated characters, TV shows, sports clubs, etc. that do not require authentication or membership to view.

To simulate searching and content viewing activities, the Facebook search bar was used to search for the users created during this experiment, and for public profiles and groups. Facebook profiles were visited and posts were viewed. This activity also involved clicking on notifications, settings etc., and viewing the content on those pages. This experiment will highlight the artefacts created as a result of searches and viewing content on Facebook; this is expected to cover the basic activities of registered and unregistered users in this category.

Sharing Content

Sharing content on Facebook involves typing and posting text, video, photos, links etc. on a user profile. A user can also like, share, or reply posts from other users. Facebook provides chat and messaging features which allow users to exchange content privately. Users can also communicate using hashtags to highlight topics of interest. According to Facebook, *“Hashtags turn topics and phrases into clickable links in your posts on your personal timeline or Page. This helps people find posts about topics they’re interested in”* [Facebook Help Centre, 2019b]; a user can search for a hashtag using the Facebook search bar.

Facebook posts can be edited or deleted after the fact. For this experiment, messages were exchanged between the experimental user accounts. Comments were made on

posts and content was shared. Due to the vast number of potential usage scenarios on Facebook, it was necessary to select, and simulate a subset of activities as described in Facebook Help Centre. This was done to establish a baseline for how Facebook could be used.

Friending

This is a way Facebook users form social relationships and connect to other users. The friending process involves sending a friend request to another Facebook user; an unregistered user may also become a friend if an invite is sent to their known email address or mobile phone number, and they register on Facebook. When a friend request is sent, both sender and recipient become friends when it is accepted by the recipient. Facebook's privacy settings present a way for users to control what their friends can see, or do on their profiles.

Facebook also has a 'follow' process, which allows users to connect with people or things they find interesting, for example, public figures, big brands, events, etc. For this experiment, the test users sent friend requests to each other. This was done to determine the artefacts created which may be used to infer user social relationships and associations on Facebook.

4.3.1.1 Availability and Recoverability of Artefacts

As done with the Twitter experiments [Section 4.2.1.1], it was necessary to pause the VM and copy the virtual disk to a working folder; and also to verify the availability, recoverability, and validity of previously recovered artefacts, by repeating a range of the outlined activities at the end of the research.

It was necessary, due to time constraints, to limit the number of potential use cases covered in the experiments. This is because the variability of user activity presents multiple scenarios that could occur. These factors were duly considered and deemed impractical thus, the activities outlined in **Section 4.3.1** were selected as a baseline.

4.3.2 Results

This section contains the results for Facebook activity using Firefox. These results can be regarded as case-specific artefacts inferring user activity. The results in this section were recovered using a range of the digital forensics tools discussed in **Section 3.6**.

4.3. FACEBOOK

Browser History

WinHex was used to extract the History profile folder from %APPDATA%\Mozilla\Firefox\Profiles. It is important to note that since these set of experiments were conducted using Firefox, there will be similarities in the location and category of artefacts recovered. As discussed in **Section 2.6.4**, it is possible to use attributes from URLs to contextualise user activity. URL artefacts were recovered from the history profile folder using DB Browser for SQLite [Table 3.2]. In addition to the SQLite database files, the directory also contains files in different formats such as .json, .jsonlz4, .xml, and .txt [Figure 4.1].

As discussed in **Section 4.2.2**, analysing data from the browser history was aimed at determining the type of information that could be recovered, and its availability in a live triage. The analysis of browser history data highlights the type of artefacts that are of interest to this research. To analyse the data related to a user's browser history, a range of the tools [Section 3.6] were used for specific data types and are briefly discussed below.

content-prefs.sqlite

This database contains user settings for websites [MDN, 2019]. The data contained in content-prefs.sqlite can be used as an indicator of, and to corroborate website visit data found in places.sqlite [Section 4.2.2]. An example of the data recovered in content-prefs.sqlite was information on the directory used for uploading content to Facebook [Figure 4.14]. This information recovered can be used to determine if any external devices were in use, and if they can be of forensic interest during a detailed forensic analysis.

time_stamp	id	groupID	settingID	value
20/02/2019 21:16	1	1	1	Z:\shared_folders_josh\pics
shared folder - mounted as network drive				

Figure 4.14: Shared network folder (mapped as a network drive) added for uploads and downloads

cookies.sqlite

The data of interest can be found in the host and associated timestamp columns. This information should be corroborated by other social media artefacts in order to determine when a site was visited.

createdTime	lastAccessedTime	expiryTime	baseDomain	name	value	host
20/02/2019 21:20	20/02/2019 21:20	01/01/1970 00:25	facebook.com	x-src	%2Fsettings%7Cpagelet_bluebar	.facebook.com
20/02/2019 20:58	20/02/2019 21:24	01/01/1970 00:26	facebook.com	datr	Xb9tXGuZWbqwlCLNjMu72_kl	.facebook.com
20/02/2019 21:11	20/02/2019 21:24	01/01/1970 00:25	facebook.com	locale	en_GB	.facebook.com
20/02/2019 20:58	20/02/2019 21:24	01/01/1970 00:26	facebook.com	sb	Xb9tXCXaFdc9KJtyim9edE0	.facebook.com
					1rFkh75eZ82HDt7Uz.AWXpULRUk syMcH47pQil2_CniQl.Bcbb9d.S7.F	
20/02/2019 20:58	20/02/2019 21:24	01/01/1970 00:25	facebook.com	fr	xt.0.0.8cbcSz.AWV6EBJe	.facebook.com
20/02/2019 20:59	20/02/2019 21:24	01/01/1970 00:25	facebook.com	wd	1024x654	.facebook.com

Figure 4.15: Facebook cookie data set for user in Firefox

favicons.sqlite

This database stores information about the icons for a user's Favourites/Bookmarks. The data of interest was found in the *page_url_hash* column in the *moz_pages_w_icons* table [Section 4.2.2]. This information can provide insight about user activity and can be used to corroborate other history related information inferring a user's activities on Facebook. Extracts of artefacts recovered from *moz_pages_w_icons* can be seen in Figure 4.16, and they corroborate the activities that took place during the experiments.

page_url
https://www.facebook.com/r.php?next=https%3A%2F%2Fwww.facebook.com%2F&locale=en_GB&display=page
https://accounts.google.com/signin/oauth?client_id=150...n49j6kmhkhjam4b00on1gb2n.apps.googleusercontent.com&as=AIOHfPMM-Agw8qLeTuzXxw&Email=ol...&destination=https://www.facebook.com&approval_state=...
https://www.facebook.com/confirmemail.php?next=https%3A%2F%2Fwww.facebook.com%2F&rd&__req...
https://www.facebook.com/home.php?ref=wizard
https://accounts.google.com/signin/oauth/identifier?client_id=150...49j6kmhkhjam4b00on1gb2n.apps.googleusercontent.com&as=AIOHfPMM-Agw8qLeTuzXxw&Email=ol...&destination=https%3A%2F%2Fwww.facebook.com&approval_state=...
https://www.facebook.com/..._im&oauthdpr=1&xrfsig=...
https://www.facebook.com/
https://www.facebook.com/?...
https://www.facebook.com/ol...7...&smuh=29172&lh=Ac8elqzjoixUMJf
https://accounts.google.com/signin/v2/challenge/pwd?client_id=150...j6kmhkhjam4b00on1gb2n.apps.googleusercontent.com&as=AIOHfPMM-Agw8qLeTuzXxw&Email=ol...&destination=https%3A%2F%2Fwww.facebook.com&approval_state=...
F9f8v8hGILG2uPUYcAKx4VLpzjp_im&oauthdpr=1&xrfsig=...
Als4RwUWx4zerAJ8YTxC9SUIIno4RAXRxMvSAxYgly4Pf3nRFHJxUhrnfWksRZurTCHT69Ht9kWiQrKRF-...
https://www.facebook.com/oauth2/redirect/?state=...
SjsASDaqSieVdoKiDdARTirYFmJ5eQKvHb9k5mLWnlBwqW8rY8V0tqwWq5Fi3u8lDJQ3pY2U9v5EZ5vWZrbo0NloaxbCUokwGVcozaNsXRz4A3UyFPxNpYVcRblR6d-2SLzvHM7KcJ4JfMzC_zSnYXw8_kMBZqzNI&code=...
mail=https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fplus.me+...&session_state=...
...&prompt=consent

Figure 4.16: Favicon artefacts showing Facebook account activities

formhistory.sqlite

This *moz_formhistory* table was discussed in Section 4.2.2. The username, email address, and a registered mobile phone number used for Facebook login were recovered. Extracts are seen in Figures 4.17 and 4.18. These results verify the data entered into form fields in Facebook during the experiments.

4.3. FACEBOOK

fieldname	value	timeFirstUsed	timeLastUsed	timesUsed
email	o[REDACTED]@gmail.com	05/11/2018 11:31	05/11/2018 11:31	1
firstname	Obed	05/11/2018 11:36	05/11/2018 11:36	1
lastname	[REDACTED]	05/11/2018 11:36	05/11/2018 11:36	1
reg_email__	o[REDACTED]@gmail.com	05/11/2018 11:36	05/11/2018 11:36	1
reg_email_confirmation__	o[REDACTED]@gmail.com	05/11/2018 11:36	05/11/2018 11:36	1

Figure 4.17: Extracts from *moz_formhistory* showing user name and associated email addresses

fieldname	value	datetime(firstUsed	datetime(lastUsed	timesUsed
email	7[REDACTED]2	20/02/2019 20:58	20/02/2019 20:58	1

Figure 4.18: Extracts from *moz_formhistory* showing the mobile phone number used for logging in to Facebook

permissions.sqlite

This database stores information about permissions granted to websites [Section 4.2.2]. This information of interest can be recovered from the origin and type columns in the *moz_perms* table; where Origin contains the URL, and type indicates the type of permission granted.

places.sqlite

As shown in Figure 4.4, this database contains fourteen tables and is used to store information like browsing history, input history, and bookmarks [Section 4.2.2]. For this research, the *moz_places* and *moz_historyvisits* tables are considered of interest as they contain visited URLs with dates and times of visit, and the number of times the web page was visited. The artefacts recovered as seen in extracts in Figure 4.19, shows the user (Josh) navigating through Facebook; searching for users Obed and Seth; searching for and visiting Facebook groups. The URLs also include the user's Facebook profile ID, some of which have been redacted for privacy purposes. These results are representative of, and confirm the activities that took place during the experiments.

visit_date	url	title	page title
20/02/2019 20:58	http://www.facebook.com/		(10) Facebook
20/02/2019 20:58	https://www.facebook.com/search/top/?q=obed%20and%20seth&epa=SEARCH_BOX	searching for Obed and Seth by name	(9) obed and seth - Facebook Search
20/02/2019 20:59	https://www.facebook.com/search/str/obed%20and%20seth/keywords_search?epa=SEARCH_BOX		(9) obed and seth - Facebook Search
20/02/2019 21:00	https://www.facebook.com/search/str/seth/keywords_search?epa=SEARCH_BOX		(9) seth - Facebook Search
20/02/2019 21:01	https://www.facebook.com/settings/apps/tabsedit/	tab navigation	(10) Privacy Settings and Tools
20/02/2019 21:01	https://www.facebook.com/settings?tab=applications		(10) Apps and websites
20/02/2019 21:01	https://www.facebook.com/settings?tab=privacy		(10) Privacy Settings and Tools
20/02/2019 21:02	https://www.facebook.com/settings?tab=privacy&section=canfriend&view		(10) Privacy Settings and Tools
20/02/2019 21:02	https://www.facebook.com/settings?tab=privacy&section=friendlist&view	searching for Obed and Seth by email and phone number	(10) Privacy Settings and Tools
20/02/2019 21:02	https://www.facebook.com/settings?tab=privacy&section=findemail&view		(10) Privacy Settings and Tools
20/02/2019 21:02	https://www.facebook.com/settings?tab=privacy&section=findphone&view		(10) Privacy Settings and Tools
20/02/2019 21:02	https://www.facebook.com/settings?tab=privacy&section=search&view	searching for groups	(10) Privacy Settings and Tools
20/02/2019 21:04	https://www.facebook.com/search/top/?q=074&epa=SEARCH_BOX		(10) 074 - Facebook Search
20/02/2019 21:04	https://www.facebook.com/search/str/silvia%20maria%20silva/keywords_search?epa=SEARCH_BOX		(10) silvia maria silva - Facebook Search
20/02/2019 21:04	https://www.facebook.com/search/people/?q=silvia%20maria%20silva&epa=SERP_TAB		(10) silvia maria silva - Facebook Search
20/02/2019 21:05	https://www.facebook.com/search/str/obed%20and%20seth/keywords_users?epa=SEARCH_BOX		(10) obed and seth - Facebook Search
20/02/2019 21:05	https://www.facebook.com/search/str/seth/keywords_users?epa=SEARCH_BOX		(10) seth - Facebook Search
20/02/2019 21:05	https://www.facebook.com/search/top/?q=people%20like%20you%20are%20the%20reason%20i%20drink&epa=SEARCH_BOX		(10) people like you are the reason i drink - Facebook Search
20/02/2019 21:06	https://www.facebook.com/people-like-you-are-the-reason-i-drink-281564511878484/?ref=br_rs		(10) People like you are the reason i drink - Home
20/02/2019 21:05	https://www.facebook.com/search/str/gumpycat/keywords_search?epa=SEARCH_BOX		(10) gumpycat - Facebook Search
20/02/2019 21:07	https://www.facebook.com/TheOfficialGrumpyCat/		(10) The Official Grumpy Cat - Home
20/02/2019 21:07	https://www.facebook.com/search/top/?q=eric%20and%20joshua&epa=SEARCH_BOX		(10) eric and joshua - Facebook Search
20/02/2019 21:08	https://www.facebook.com/OfficialCartman/		(10) Eric Cartman - Home
20/02/2019 21:11	https://www.facebook.com/		Facebook, log in or sign up
20/02/2019 21:12	https://www.facebook.com/profile.php?id=100000000000000&ref=bookmarks	profile view/visit	(10) Joshua
20/02/2019 21:13	https://www.facebook.com/100000000000000		(10) Joshua
20/02/2019 21:13	https://www.facebook.com/profile.php?id=100000000000000		(10) Joshua
20/02/2019 21:20	https://www.facebook.com/		Facebook, log in or sign up

Figure 4.19: Extracts showing Facebook history activity, sorted by visit date

The recovered artefacts were exported, and saved as a CSV file; this made it easier to filter, and manipulate the data when identifying properties that could be used for relationship attribution. This also made it easier to corroborate these artefacts with other data recovered from other SQLite databases.

webappsstore.sqlite

As discussed in **Section 4.2.2**, the data in the originKey, key and value columns are of interest to this research. However, in this instance, the data recovered from the *webappsstore2* table shows the website settings that enables the personalisation of content (e.g. adverts) on Facebook [Facebook, 2018a], and did not yield any data of interest.

sessionstore.jsonlz4

The *sessionstore.jsonlz4* file was extracted using WinHex [Section 4.2.2]. The artefacts from *sessionstore* show the user's searches on Facebook. From this, it is possible to identify attributes in the URL such as '*search/str/grumpy+cat/keywords_search?*'; or '*/TheOfficialGrumpyCat*', which can be used to infer user activity, describe the sequence of events that occurred, and determine user association. The artefacts recovered are consistent with user activity as recorded during experiments and as recovered from the *places.sqlite* database. Extracts from *sessionstore* are shown in Figure 4.20 (line breaks inserted for readability):

As seen in Figure 4.20, some valuable intelligence can be gathered from *sessionstore*. The artefacts in the *sessionstore* file tells a story, starting from when a user logged in to Facebook, and the activities that took place; to when the user logs out, and

4.3. FACEBOOK

```
{
  "version": ["sessionrestore", 1],
  "windows": [
    {
      "tabs": [
        {
          "entries": [
            {
              "url": "https://www.facebook.com/search/top/?q=people%20like%20you%20are%20the%20reason%20i%20drink&epa=SEARCH_BOX",
              "title": "(10) people like you are the reason i drink - Facebook Search",
              "url": "https://www.facebook.com/search/str/grumpy+cat/keywords_search?epa=SEARCH_BOX",
              "title": "(10) grumpy cat - Facebook Search",
              "resultPrincipalURI": null,
              "url": "https://www.facebook.com/TheOfficialGrumpyCat/",
              "title": "(10) The Official Grumpy Cat - Home",
              "subframe": true,
              "cacheKey": 0,
              "ID": 54,
              "docshellUUID": "{5dad172-0952-4ced-9969-f9f68c0db433}",
              "resultPrincipalURI": null,
              "scroll": "0,13680"
            },
            {
              "url": "https://www.facebook.com/search/top/?q=eric%20cartman&epa=SEARCH_BOX",
              "title": "(10) eric cartman - Facebook Search",
              "subframe": true,
              "cacheKey": 0,
              "ID": 67,
              "docshellUUID": "{5dad172-0952-4ced-9969-f9f68c0db433}",
              "resultPrincipalURI": null
            }
          ]
        }
      ]
    }
  ]
}
```

Figure 4.20: Extracts showing Facebook searches and page visit

terminates the browser session. This information can be used to corroborate history or cache artefacts recovered. In a time sensitive investigation, where an investigator needs to know if there has been any form of contact between a victim and a suspect, the artefacts in sessionstore can be used to support a timeline of activity.

logins.json

As discussed in **Section 4.2.2**, this file is used to store the encrypted versions of the username and password. Extracts from the `logins.json` file for the Facebook user is shown below (redacted):

```
{
  "nextId": 2,
  "logins": [
    {
      "id": 1,
      "hostname": "https://www.facebook.com",
      "httpRealm": null,
      "formSubmitURL": "https://www.facebook.com",
      "usernameField": "email",
      "passwordField": "pass",
      "encryptedUsername": "MDoE[REDACTED]7mb",
      "encryptedPassword": "MDoE[REDACTED]UcJ",
      "timeCreated": 1550696337049,
      "timeLastUsed": 1550696337049,
      "timePasswordChanged": 1550696337049,
      "timesUsed": 1,
      "disabledHosts": [],
      "version": 2
    }
  ]
}
```

As seen above, this file contains data of forensic interest i.e. the hostname where the login credentials are used; the created and last used time; date when the password was changed, and the number of times the credentials have been used. This data can be used to corroborate the other artefacts recovered; for example, data recovered from `sessionstore`, `formhistory.sqlite`, or `places.sqlite`. This data could be used,

in addition to context and case specific artefacts to connect a user to a Facebook account.

Browser Cache

The Firefox cache is in %LOCALAPPDATA%\Mozilla\Firefox\Profiles. The cache directory shares the same naming convention and an identical name with the History profile directory, and is used to temporarily store, images, scripts and other files from visited websites to enhance efficiency [Section 4.2.2].

To analyse the contents of the cache, the profile directory was extracted using WinHex and as part of the analysis, MZCacheView was used to examine the cache contents as described in Section 4.2.2. The artefacts of interest were those indicative of a user's direct interaction with other test users; for example, cached profile information (user IDs, URLs and images). The URLs recovered from the cache were used to corroborate the artefacts recovered from webappsstore and sessionstore. The following are extracts of the information recovered from the cache:

```
=====
[SNIP]
Filename       : filter[0]=app&filter[1]=page&filter[2]=group&filter[3]=friendli
URL            : https://www.facebook.com/ajax/typeahead/search/facebar/bootstrap/?filter[0]=
                app&filter[1]=page&filter[2]=group&filter[3]=friendlist&
                context=facebar&viewer=1000*****96733&
                token=v7&lazy=1&__user=100*****96733&__a=1}
Cache Name     : 1D4271E6FBD5DD59630577B29811BF79FE751FC8
Content Type   : application/x-javascript; charset=utf-8
Last Modified  : 13/10/2019 10:03:49
File Size     : 123
Fetch Count    : 1541418022
Last Fetched   : 13/10/2019 10:03:49
Server Response : HTTP/2.0 200 OK
Server Time    : 05/11/2018 11:40:22
Cache Control  : public,max-age=86400
URL Length     : 215
=====
```

```
=====
[SNIP]
Filename       : filter[0]=app&filter[1]=page&filter[2]=group&filter[3]=friendli
URL            : https://www.facebook.com/ajax/typeahead/search/facebar/bootstrap/?filter[0]=
                app&filter[1]=page&filter[2]=group&filter[3]=friendlist&
                context=facebar&viewer=1000*****51748&
                token=v7&lazy=1&__user=1000*****51748&__a=1
Cache Name     : 0D200A29D42C71411D30C801246A3F81305ACD37
Content Type   : application/x-javascript; charset=utf-8
Last Modified  : 20/02/2019 21:14:44
File Size     : 121
Fetch Count    : 3611820032
Last Fetched   : 20/02/2019 21:14:44
Server Response : HTTP/2.0 200 OK
Server Time    : 20/02/2019 21:14:47
```

4.4. GOOGLE DRIVE

```
Cache Control      : public,max-age=86400
URL Length        : 336
=====
```

It is necessary to note that other artefacts besides the URL and timestamps as shown in the extracts, can be recovered from the cache (e.g. images, audio, text files may be recovered), however, this research is focused on identifying potential sources of intelligence for a live triage investigation. Thus, the artefacts of interest were narrowed down to those pieces of evidence that do not require an extensive amount of time to recover and utilise.

The cache artefacts can be used to determine user relationships however, as with cookies and web storage data, there are caveats. Unrelated content for example, content from advertising may be cached on disk. In this context, the presence of these artefacts in the cache does not necessarily equate to a visit; to determine or attribute user relationship, other artefacts should be used as corroborating evidence.

4.3.3 Summary

The research objectives were discussed in **Section 3.4**, and the artefacts discussed in this section address the key problems identified in **Section 3.3**; with regards to recovering artefacts that could be used where communication content may not be immediately accessible. **Section 4.5** discusses how this experiment meets the objectives and addresses the research question.

4.4 Google Drive

This section discusses the experiments and results from Google Drive activity using Firefox. The results discussed in this section are from the manual recovery processes, using a range of the digital forensics tools discussed in **Section 3.6**.

4.4.1 Experiments

This section discusses the data generation and collection process. As with Twitter and Facebook, experiments were conducted using each of the three user accounts created for this research [**Section 3.9.2**]. The experiments for Google Drive consists of a range of activities that were carried out to simulate real life user activity; and the actions described below were repeated for each user on Firefox. As discussed in **Section 3.6.1**, Procmon and Fiddler were used to log system activity, and used to capture HTTPS traffic respectively.

As discussed in **Section 3.7**, for the purposes of this research, Google Drive is

regarded as a social media platform as it presents a platform for propagating communication through content sharing, and user collaboration. In **Section 4.2.1**, Gmail addresses were created; this was necessary because a Google account is required in order to use Google Drive.

To access Google Drive, a user can type in the URL `drive.google.com` in the browser address bar. G Suite Learning Center [2019] describes the ways in which Google Drive can be used; the categories of experiments performed are based on a subset of the use cases described. The actions taken during the experiments were recorded to provide a documented means of verifying artefacts of interest.

Create and Store

Users on Google Drive (hereafter referred to as Drive) can create a variety of files such as text documents, spreadsheets, slides, drawings, forms, and mind maps. Existing files (such as text files, PDF, Image files, and videos) stored on other devices can also be uploaded, stored, and viewed in Drive [G Suite Learning Center, 2019].

For this research, the first set of activities involved the creation of documents locally on the computer. These documents were then uploaded to Drive through a folder shared with the VM. This was followed by the creation of new documents, and folders in Drive by clicking the *New* button, and selecting the new document format.

These set of activities were performed to simulate user activity; this experiment will highlight the artefacts created as a result of document creation, and uploads which may be of interest during a forensic investigation.

Document Access and Update

Documents stored in Drive can be accessed from any web browser or digital device that has Drive installed. When changes are made to the documents stored in Drive, a sync activity is initiated, so when the document is accessed from a different device, all changes made would be visible [G Suite Learning Center, 2019].

The experiments conducted involved viewing and modifying the stored documents in Drive. The activities in this category also involved downloading files and folders, adding files to folders, and removing (deleting) files. These were done to simulate how a user may interact with the content in Drive.

Share and collaborate

Drive users share files and folders with each other; the access levels on shared content can be set by the content owner, to allow modification by users who have access

4.4. GOOGLE DRIVE

to the shared documents. Shared content have three access levels; can edit, can comment, and can view [G Suite Learning Center, 2019]. Files can be shared by clicking *Share* and then adding other users or by clicking *Share*, then clicking *Get shareable link* and forwarding the link to other users.

For this research, files and folders were shared between the three user accounts created for these experiments; modifications were made to shared documents, and shared documents were searched for using the search function in Drive.

4.4.1.1 Availability and Recoverability of Artefacts

As done with the previous experiments [Sections 4.2.1.1; 4.3.1.1] in this Chapter, it was necessary to pause the VM and copy the virtual disk to a working folder; and also to verify the availability, recoverability, and validity of previously recovered artefacts, by repeating a range of the outlined activities at the end of the research.

The variability of user activity presents multiple scenarios that could occur, for example, the creation and use of shared drives, using Drive through the browser alongside the Desktop application, and offline access to Drive. These factors were duly considered and deemed impractical thus, due to time constraints, the activities outlined in [Section 4.4.1] were selected as a baseline.

4.4.2 Results

This section contains the results for Drive activity using Firefox. These results can be regarded as case-specific artefacts inferring user activity. A range of the digital forensics tools discussed in Section 3.6 were used for the manual recovery and analysis of the results in this section.

Browser History

The location and contents of the Firefox History profile directory were discussed in Sections 4.2.2 and 4.3.2. As these set of experiments were also conducted using Firefox, there will be similarities in the location and category of artefacts recovered for all three social media platforms. The analysis of browser history data from Drive activity highlights the type of artefacts that are of interest to this research. The experimental results analysed are briefly discussed below.

content-prefs.sqlite

This database contains user settings and preferences for websites. For example, browser download or upload directory information [MDN, 2019]; as shown below, the user has a set preference for the use of the path `Z:\shared_folders_<user>`.

4.4. GOOGLE DRIVE

This is a shared network folder set up in VMware, but mapped as a network drive in Windows. This information could be used to identify external devices that may be of forensic interest in an investigation.

Example preference (uploads) for user Obed

time_stamp	id	groupID	settingID	value
02/12/2018 16:50	1	1	1	Z:\shared_folders_obed\documents

Example preference (uploads) for user Seth

time_stamp	id	groupID	settingID	value
02/12/2018 17:32	1	1	1	Z:\shared_folders_seth\my_docs

cookies.sqlite

The data of interest in can be found in the host, and associated timestamp columns of the *moz_cookies* table. This information can be used to understand when a site was visited and can be corroborated by, or used to corroborate other related artefacts. For these set of experiments, the following cookies were set for host *.docs.google.com* when documents were created or accessed:

createdTime	lastAccessedTime	value	host
02/12/2018 17:28	02/12/2018 17:29	documents=_Yius.[SNIP].S0Uc	.docs.google.com
02/12/2018 18:50	02/12/2018 18:50	documents=DAZdH.[SNIP].qGKT	.docs.google.com
02/12/2018 18:51	02/12/2018 18:51	documents=RCX2N.[SNIP]._UIk	.docs.google.com
02/12/2018 18:50	02/12/2018 18:52	documents=pZ-4m.[SNIP].E7k4	.docs.google.com

favicons.sqlite

The information in the *page_url_hash* column in the *moz_pages_w_icons* table [Section 4.2.2] can provide context, and be used to corroborate other artefacts. Extracts of artefacts recovered from *moz_pages_w_icons* are shown below. The data recovered from this table appears to corroborate the actions taken when Drive was accessed and files modified during the experiments.

page_url

<https://drive.google.com/drive/shared-with-me>
<https://drive.google.com/drive/my-drive>
<https://www.google.com/intl/en-GB/drive/>
[https://docs.google.com/document/u/0/d/1076G_\[SNIP\]_NGFABHJkqqDbdEe6wRM/edit](https://docs.google.com/document/u/0/d/1076G_[SNIP]_NGFABHJkqqDbdEe6wRM/edit)

page_url

<https://drive.google.com/drive/my-drive>
[https://docs.google.com/document/u/0/d/1UIrwywvGIId.\[SNIP\].4hQCNAG-172S0Y/edit](https://docs.google.com/document/u/0/d/1UIrwywvGIId.[SNIP].4hQCNAG-172S0Y/edit)
[https://docs.google.com/document/d/1UIrwywvGIId-\[SNIP\].9ik4hQCNAG-172S0Y/edit](https://docs.google.com/document/d/1UIrwywvGIId-[SNIP].9ik4hQCNAG-172S0Y/edit)
[https://docs.google.com/document/d/1FqsuG.\[SNIP\].JUG8GJm1hmdU1syr0moDCIro/edit](https://docs.google.com/document/d/1FqsuG.[SNIP].JUG8GJm1hmdU1syr0moDCIro/edit)
[https://docs.google.com/document/d/1F.\[SNIP\].ro/edit?usp=drive_web&ouid=11***62](https://docs.google.com/document/d/1F.[SNIP].ro/edit?usp=drive_web&ouid=11***62)

4.4. GOOGLE DRIVE

places.sqlite

This research considered the artefacts in the *moz_places* and *moz_historyvisits* tables as artefacts of interest, because they contain information that can be used to make inferences about user behaviour. With regards to Drive, the data contained in these tables can be used to infer document or folder creation, document or folder ID, tab titles indicating the document name etc. It is possible to use this information to determine whether a suspect has created, modified or shared documents on Drive; this is because a unique opaque ID is created for each file and file IDs are stable even if the file name changes [Google Developers, n.d].

To make data filtering and manipulation easier, the results were exported and saved as CSV [Section 4.2.2]. The following are extracts from the *moz_places* and *moz_historyvisits* tables for the test user accounts.

```
url          [page_title]
https://drive.google.com/drive/my-drive
[Page Title: My Drive - Google Drive]
```

```
https://docs.google.com/document/u/0/create?usp=drive_web&folder=0A****Uk9PVA
[Page Title: My Drive - Google Drive]
```

```
https://docs.google.com/document/u/0/d/1dk****5Y/edit
[Page Title: Untitled document - Google Docs]
```

```
https://docs.google.com/document/d/1dk****5Y/edit
[Page Title: favourite tv shows - Google Docs]
```

```
https://docs.google.com/document/d/1xd****XmzTI/edit
[Page Title: Oma's applecake - Google Docs]
```

```
https://docs.google.com/document/d/107****6wRM/edit
[Page Title: Treats - Google Docs]
```

```
https://docs.google.com/document/d/1UIr****S0Y/edit
[Page Title: funny quotes_all - Google Docs]
```

```
https://drive.google.com/drive/shared-with-me
[Page Title: Shared with me - Google Drive]
```

```
https://docs.google.com/document/d/1fo****af8w/edit
[Page Title: Oma's applecake - Google Docs]
```

sessionstore.jsonlz4

The decompressed **sessionstore** file from the Drive experiments was analysed in JSONViewer. The artefacts recovered appear to corroborate user activity (e.g. creating, or accessing documents; sharing files, and accessing shared files) as recorded during the experiments, and supports the data recovered from the **places.sqlite** database.

Due to the volume of data in the **sessionstore** file, snippets of "entries" were extracted and are shown in this section (with line breaks inserted for brevity). They infer that users Seth and Obed created documents; and that user Josh accessed a shared document; both supporting the history artefacts recovered from **places.sqlite**.

Josh accessed shared document [Oma's applecake]:

```
{"entries":[{"url":
"https://docs.google.com/document/d/1xd8bxWEEzcegQ****uXmzTI/edit",
"title":"Oma's applecake - Google Docs",
"referrer":"https://drive.google.com/drive/shared-with-me",
```

Obed created a document [Treats]:

```
{"entries":[{"url":
"https://docs.google.com/document/d/1076****e6wRM/edit",
"title":"Treats - Google Docs",
"originalURI":"https://docs.google.com/document/d/107****qqDbdEe6wRM/edit",
```

The artefacts recovered show access to a shared document Oma's applecake. This corroborates the artefacts recovered earlier from the *moz_places* and *moz_historyvisits* tables in **places.sqlite**. From the example above, it is possible to extract attributes in the URL such as the file ID: /d/1xd8bxWEE****DSKMuXmzTI; a search for artefacts matching that pattern can be conducted using RegEx. This information could be of interest in an investigation when describing the sequence of events that occurred e.g. a document was created, a shared document was accessed. Using other supporting evidence, these artefacts can be used to attribute actions to a user.

A simultaneous keyword search was initiated in WinHex. This was done to further identify locations where the content exchanged on Drive may be found. Extracts of file contents were identified in **unallocated space**, a deleted **places.sqlite.wal** file, and **pagefile.sys**; they were used to determine if the content of the files could be recovered. The results of the keyword search corroborates the actions taken during the experiments and are shown below:

4.4. GOOGLE DRIVE

Document created locally and uploaded:

5979132368	00 3F 00 5C 5C 76 6D 77	61 72 65 2D 68 6F 73 74	? \\vmware-host
5979132384	5C 53 68 61 72 65 64 20	46 6F 6C 64 65 72 73 00	\Shared Folders
5979132400	5A 3A 00 73 68 61 72 65	64 5F 66 6F 6C 64 65 72	Z: shared_folder
5979132416	73 5F 6A 6F 73 68 5C 64	6F 63 73 5C 4D 6F 72 72	s_josh\docs\Morr
5979132432	69 73 20 64 61 6E 63 69	6E 67 2E 64 6F 63 78 00	is dancing.docx
5979132448	1B 00 5A 00 3A 00 5C 00	73 00 68 00 61 00 72 00	Z : \ s h a r
5979132464	65 00 64 00 5F 00 66 00	6F 00 6C 00 64 00 65 00	e d _ f o l d e
5979132480	72 00 73 00 5F 00 6A 00	6F 00 73 00 68 00 5C 00	r s _ j o s h \
5979132496	64 00 6F 00 63 00 73 00	28 00 00 00 09 00 00 A0	d o c s (

6522422080	3D 22 4D 6F 72 72 69 73	20 64 61 6E 63 69 6E 67	= "Morris dancing
6522422096	2E 64 6F 63 78 22 20 64	61 74 61 2D 74 6F 6F 6C	.docx" data-tool
6522422112	74 69 70 2D 6F 6E 6C 79	2D 6F 6E 2D 6F 76 65 72	tip-only-on-over
6522422128	66 6C 6F 77 3D 22 74 72	75 65 22 3E 4D 6F 72 72	flow="true">Morr
6522422144	69 73 20 64 61 6E 63 69	6E 67 2E 64 6F 63 78 3C	is dancing.docx<
6522422160	2F 64 69 76 3E 3C 64 69	76 20 63 6C 61 73 73 3D	/div><div class=
6522422176	22 61 2D 48 5A 6E 6A 7A	64 2D 78 62 2D 4E 70 22	"a-HZnjzd-xb-Np"
6522422192	20 64 61 74 61 2D 74 6F	6F 6C 74 69 70 3D 22 59	data-tooltip="Y
6522422208	6F 75 20 75 70 6C 6F 61	64 65 64 20 74 6F 64 61	ou uploaded toda
6522422224	79 22 20 64 61 74 61 2D	74 6F 6F 6C 74 69 70 2D	y" data-tooltip-

Fragment of a document stored in Drive:

8928944320	08 01 06 25 08 05 84 15	00 68 74 74 70 73 3A 2F	% ,, https:/
8928944336	2F 64 6F 63 73 2E 67 6F	6F 67 6C 65 2E 63 6F 6D	/docs.google.com
8928944352	2F 64 6F 63 75 6D 65 6E	74 2F 64 2F 31 78 64 38	/document/d/1xd8
8928944368	XX XX XX XX XX XX XX XX	XX XX XX XX XX XX XX XX	b*****k
8928944384	45 4F 47 44 37 71 54 30	4D 5A 57 62 49 37 44 53	EOGD7qTOMZWbI7DS
8928944400	4B 4D 75 58 6D 7A 54 49	2F 65 64 69 74 3F 75 73	KMuXmzTI/edit?us
8928944416	70 3D 64 72 69 76 65 5F	77 65 62 26 6F 75 69 64	p=drive_web&ouid
8928944432	3D 31 31 35 39 XX XX XX	XX XX XX XX 38 39 34 39	=1159*****8949
8928944448	36 38 33 35 36 35 4F 6D	61 27 73 20 61 70 70 6C	6835650ma's appl
8928944464	65 63 61 6B 65 20 2D 20	47 6F 6F 67 6C 65 20 44	ecake - Google D
8928944480	6F 63 73 6D 6F 63 2E 65	6C 67 6F 6F 67 2E 73 63	ocsmoc.elgoog.sc
8928944496	6F 64 2E 64 00 05 7C 0E	78 C5 32 28 78 35 32 64	od.d xÅ2(x52d
8928944512	63 6C 2D 36 4B 41 59 37	2B 12 7C 1F 0D 72 4F 6D	cl-6KAY7+ rOm
8928944528	61 E2 80 99 73 20 41 70	66 65 6C 6B 75 63 68 65	aâ€™s Apfelkuche
8928944544	6E 20 28 47 72 61 6E 64	6D 61 E2 80 99 73 20 41	n (Grandmaâ€™s A
8928944560	70 70 6C 65 20 43 61 6B	65 29 20 20 54 4F 54 41	pple Cake) TOTA
8928944576	4C 20 54 49 4D 45 3A 20	50 72 65 70 3A 20 32 30	L TIME: Prep: 20
8928944592	20 6D 69 6E 2E 20 42 61	6B 65 3A 20 34 35 20 6D	min. Bake: 45 m
8928944608	69 6E 2E 20 2B 20 63 6F	6F 6C 69 6E 67 20 59 49	in. + cooling YI
8928944624	45 4C 44 3A 20 31 30 20	73 65 72 76 69 6E 67 73	ELD: 10 servings
8928944640	2E 20 49 6E 67 72 65 64	69 65 6E 74 73 20 35 20	. Ingredients 5
8928944656	6C 61 72 67 65 20 65 67	67 20 79 6F 6C 6B 73 20	large egg yolks

Fragment of document collaboration invitation:

```

5709974192 34 36 32 26 72 65 71 75 65 73 74 54 79 70 65 3D 462& requestType=
5709974208 69 6E 76 69 74 65 26 69 74 65 6D 49 64 73 3D 31 invite & itemIds=1
5709974224 79 6A 68 56 XX XX XX XX XX XX XX XX 4B 5F 73 32 yjhVXXXXXXXXXK_s2
5709974240 41 48 30 68 XX XX XX XX XX XX XX XX 51 42 49 37 AH0hXXXXXXXXXQBI7
5709974256 26 63 6F 6E 66 69 72 6D 65 64 3D 66 61 6C 73 65 &confirmed=false
5709974272 26 72 6F 6C 65 3D 33 30 26 73 65 6E 64 45 6D 61 &role=30& sendEma
5709974288 69 6C 3D 74 72 75 65 26 6E 6F 74 69 66 69 63 61 il=true &notifica
5709974304 74 69 6F 6E 49 6E 66 6F 3D 25 37 42 25 32 32 6D tionInfo=%7B%22 m
5709974320 65 73 73 61 67 65 54 79 70 65 25 32 32 25 33 41 essageType %22%3A
5709974336 25 32 32 69 6E 76 69 74 65 25 32 32 25 32 43 25 %22 invite %22%2C%
5709974352 32 32 72 65 63 69 70 69 65 6E 74 25 32 32 25 33 22 recipient %22%3
5709974368 41 25 32 32 6F 6B XX XX XX XX XX XX XX 25 34 30 A%22 ok***** %40
5709974384 67 6D 61 69 6C 2E 63 6F 6D 25 32 43 25 32 30 73 gmail.com %2C%20s
5709974400 6A 6F XX XX XX XX XX XX 25 34 30 67 6D 61 69 6C jo***** %40 gmail
5709974416 2E 63 6F 6D 25 32 32 25 32 43 25 32 32 6D 65 73 .com %22%2C%22 mes
5709974432 73 61 67 65 25 32 32 25 33 41 25 32 32 48 65 79 sage %22%3A%22 Hey
5709974448 25 32 30 67 75 79 73 25 32 43 25 32 30 70 6C 65 %20 guys %2C%20 ple
5709974464 61 73 65 25 32 30 63 6F 6E 74 72 69 62 75 74 65 ase %20 contribute
5709974480 25 32 30 74 6F 25 32 30 74 68 69 73 25 32 32 25 %20 to %20 this %22%
5709974496 32 43 25 32 32 72 6F 6C 65 25 32 32 25 33 41 33 2C%22 role %22%3A3

```

logins.json

As discussed in **Section 4.2.2**, this file is used to store user login information. Google Drive does not have (or require) separate login credentials, rather it makes use of the user's existing Google account credentials. The `logins.json` file can be used to determine, or corroborate last login time.

Browser Cache

The artefacts of interest were those indicative of a user's direct interaction with other users for example, cached profile information (including user IDs, URLs and images). The cached URLs were used to corroborate the artefacts other sources (e.g. `sessionstore`, `places.sqlite`). Artefacts of interest from the cache were those that corroborated the actions taken by the researcher during the experiments; for example, the extracts below show the cached content for the documents created during the experiments.

Document: Oma's applecake

=====

Filename : edit

URL : https://docs.google.com/document/d/1fo****DrDKoraf8w/edit

4.4. GOOGLE DRIVE

```
Cache Name      : 4E51D4543B7F6B3EEF04BA2B6099B7A1346F8416
Last Modified   : 13/10/2019 14:53:19
File Size       : 272
Fetch Count     : 452591616
Last Fetched    : 13/10/2019 14:53:19
URL Length      : 84
=====
=====
Filename        : edit
URL             : https://docs.google.com/document/d/1Fqs\*\*\*\*GJdUCIro/edit
Cache Name      : F4BE6E5A08E8663D36AD8E63E70DB920520BC743
Last Modified   : 13/10/2019 14:53:19
File Size       : 272
Fetch Count     : 452591616
Last Fetched    : 13/10/2019 14:53:19
URL Length      : 84
=====
Document: funny quotes
=====
Filename        : edit
URL             : https://docs.google.com/document/d/1UIr\*\*\*\*G-172S0Y/edit
Cache Name      : EE4D8106363DB464B19D0F57FA947551443A3C23
Last Modified   : 13/10/2019 14:53:15
File Size       : 272
Fetch Count     : 452591616
Last Fetched    : 13/10/2019 14:53:15
URL Length      : 84
=====
```

It is pertinent to note that other artefacts besides the header information are recoverable from the cache, however, this research is focused on identifying potential sources of intelligence for a live triage investigation. Thus, the artefacts of interest were narrowed down to those pieces of evidence that do not require an extensive amount of time and effort to retrieve and utilise. It is possible to use this information during a detailed analysis to identify and inspect specific cache entry files of forensic value, using the file name as highlighted above (Cache Name).

The cache results can be used to determine user activities and relationships however, there is a caveat; content from advertising or background processes from Google may be cached on disk. In this context, to determine or attribute user relationship, other OS and browser artefacts should be to strengthen the artefacts from Drive activity; especially in a live triage investigation where a 'mini picture' is being painted. A full digital forensic analysis can be used for validation where appropriate.

4.4.3 Summary

The research objectives were discussed in **Section 3.4**, and the artefacts discussed in this section address the key problems identified in **Section 3.3**; with regards to recovering artefacts that could be used where communication content may not be immediately accessible. **Section 4.5** discusses how this experiment meets the objectives and addresses the research question.

4.5 Relationship to Objectives and Research Question

The problem definition highlighted key areas that formed the research question [Sections 3.3; 3.4]. This section highlights how the artefacts recovered as a result of user activity on social media (Twitter, Facebook, and Google Drive) address the research question. The experimental results in this Chapter meet the following objectives:

- **[Objective 1] Characterise communications content:** the artefacts recovered highlight the user activities [Pages 73 - 75; 89 - 91; 99 - 100] initiated during the experiments, independent of the actual content being exchanged. They make a distinction between communication content and other artefacts, and demonstrate the possibility of recovering the content(s) of user communication, as shown in the tweets recovered from `sessionstore`, `unallocated space` and `pagefile.sys` [Pages 82 - 83]. Although the experiments were conducted with a limited dataset [Sections 3.5.1; 3.9.2], they set a baseline for the type of user activity that could occur on social media. The results demonstrate what constitutes communications content; and that other corroborating evidence may be used to build a picture, and ‘connect the dots’, in order to give weight to the case-specific artefacts.
- **[Objective 2] Categorise artefacts generated from the use of social media:** the artefacts recovered have been grouped into case-specific and context artefacts (e.g. user session information; web storage artefacts). They have been discussed in the context of their relevance, and how they can be used for relationship attribution. For example, a document was created during the experiment; the core browser artefacts show the document URL and name (title), and when the document was created; the cache shows information about when the document was cached. In the process of recovering and categorising these artefacts, it was necessary to identify features (attributes) that could be used to infer user action. The experimental results highlight the complementary nature of social media artefacts, i.e. how they can be corroborated, or used to corroborate other pieces of digital evidence.

4.6. RESULTS OVERVIEW

The categorisation of social media artefacts was achieved with a subset of possible usage scenarios however, the experimental methodology can be employed when investigating other potential user activities/behaviours that could occur on social media.

4.6 Results Overview

The recovery and analysis of the artefacts discussed in this Chapter highlight the artefacts that may be of interest in a live triage when investigating user social media activity; e.g. where they are located, and how they can be recovered. They highlight how the artefacts can be used to gather intelligence, and subsequently identify potential areas of focus during a detailed forensic analysis. This provides insight on known files, or areas of interest for social media investigations. Table 4.2 illustrates a summary of the results, based on the various categories of activities conducted during the experiments.

Table 4.2: Results overview - user activities and artefacts (Firefox)

		Categories of activities								
Social Media Platforms		Registration and Authentication	Searching and Viewing Content	Sharing Content	Following	Friending	Create and Store Documents	Document Access and Update	Collaboration	Availability and Recoverability
	Twitter	✓	✓	✓	✓					✓
	Facebook	✓	✓	×	×	✓				✓
	Google Drive	✓	✓	✓			✓	✓	✓	✓

- The ticks indicate the recovery of artefacts applicable to a category.
- No ticks (grey boxes) indicate that some activities are platform specific e.g.
 - “friending” is specific to Facebook so was not applicable to Google Drive or Twitter experiments
 - document collaboration is a Google Drive feature and as such was not applicable to Twitter and Facebook
- Cross boxes (x) indicate that no artefacts were recovered for that category from the experimental dataset e.g. communication content from Facebook

activity was not recovered from the experimental dataset.

During the analysis, it was necessary to highlight certain caveats to be aware of when investigating social media activity. For example, the artefacts created as a result of unrelated processes, or web storage of content the user has not interacted with. The analysis was focused on artefacts that provide quick intelligence, are easy to understand, and give an investigator a strategic advantage. The discussion highlights the need to contextualise user activity by corroborating each artefact of interest.

It was observed that the social media platforms behave differently with regard to the types of artefacts that could be recovered. This may be due to the limited experimental sample size, or due to the way each platform is designed to be used. However, the results highlighted the possibility of making inferences about user behaviour using a limited set of data, and also the possibility of determining user association(s).

4.7 Summary

This chapter discussed the ways in which social media platforms such as Twitter, Facebook and Google Drive can be used. It defined an experimental methodology that can be used to simulate user activity on different social media platforms. In the context of this research, this serves as a point of reference for mapping specific actions to the artefacts recovered during a social media related investigation; this enables the development of recovery and analysis techniques for specific artefacts types.

The chapter discussed the artefacts recovered from Firefox, created due to the subset of user activity performed on Twitter, Facebook and Google Drive. The results discussed were broadly grouped into browser history and cache artefacts. This was done to highlight where specific artefacts of interest are located. The results were discussed in the context of their relevance to a digital forensic investigation and were described as case-specific artefacts. This chapter demonstrates how relationships and actions may be attributed to a user, and highlights the importance of contextualising, and corroborating the different pieces of evidence recovered.

Chapter 5 discusses the experiments and results from Google Chrome.

5 Chrome: Social Media Artefacts

5.1 Introduction

This chapter discusses the methods used for the experiments conducted using Google Chrome. It discusses the individual experiments performed on Twitter, Facebook and Google Drive. It discusses the analysis strategy and the experimental results. The results presented in this chapter provide insight on the artefacts that are of interest in an investigation involving social media activity.

In this chapter, the following font conventions are used:

Table 5.1: Font conventions

Item	Font typespace
Paths and Filenames	<code>monospaced typewriter</code>
SQLite database table names	<i>slanted text</i>
Artefact and Hex extracts	typewriter
Quotes and definitions	<i>italics</i>

The rest of this chapter is structured as follows: **Section 5.2** discusses the experiments conducted on Twitter, the analysis of the data, and the results. **Section 5.3** discusses the Facebook experiments, analysis, and results. The Google Drive experiments are discussed in **Section 5.4**, followed by the analysis and results. **Section 5.5** discusses how these experiments address the research questions; followed by an overview of the results from Chrome in **Section 5.6**. **Section 5.7** contains the chapter summary.

5.2 Twitter

This section discusses the experiments and results from Twitter activity using Google Chrome. The results discussed in this section are from the manual recovery processes, using a range of the traditional digital forensics tools discussed in **Section 3.6**.

5.2.1 Experiments

This section details the data generation and collection process. Experiments were conducted using each of the three user accounts created on the experimental VMs [Section 3.9.2]. As discussed in **Section 3.6.1**, Procmon and Fiddler were used to used to log system activity and capture HTTPS traffic respectively. In a similar

fashion to the basic Twitter user activity described in **Section 4.2.1**, the same categories of activities make up the experiments on Google Chrome.

Registration and Authentication

The ability to connect with, and exchange messages with other users on Twitter is dependent on having a registered account [**Section 4.2.1**]. The user accounts created for these experiments were subsequently personalised i.e adding user profile photos, banners, and amending the account settings to make them private.

Searching and Viewing Content

As observed in **Chapter 4**, unregistered users are able to use search engines to find and visit publicly accessible Twitter accounts; they are also able to use the search function in Twitter to find other accounts, view trending topics and public content posted on Twitter.

To simulate the ‘search and view’ activity, the search function in Twitter was used to find public accounts, including the accounts of the other users created during this experiment. This activity also involved navigating through Twitter, and covers the activities of registered and unregistered users [**Section 4.2.1**].

Sharing Content

As with the experiments conducted in **Chapter 4**, Twitter will be used for content dissemination in the form of “Tweeting”, “Retweeting”, or hashtags; a subset of user activities were simulated; this was done to establish a baseline for artefacts that could be recovered [**Section 4.2.1**].

For this experiment, the test user accounts were used to post Tweets on own profile, post on another test user’s profile, reply Tweets, and share content through DMs.

Following

This activity enables a Twitter user to subscribe to another user’s Tweets; providing the subscriber with updates when a new Tweet is posted. Following is a two way process where users follow each other, however, it may not always be reciprocated as described in **Section 4.2.1**.

5.2.1.1 Availability and Recoverability of Artefacts

As with the experiments conducted in **Chapter 4**, the VM was paused and the virtual disk was copied to a working folder. It was necessary, due to the nature of social media platforms [**Section 3.2**], to verify the availability, recoverability,

5.2. TWITTER

and validity of previously recovered artefacts by repeating a range of the outlined activities at the end of the research.

Due to time constraints, the number of potential use cases covered in the experiments were limited. This is because the variability of user activity presents multiple scenarios that could occur. These factors were considered and it was impractical to anticipate and cover all possible use cases thus, the activities outlined in **Section 4.2.1** were selected as a baseline.

5.2.2 Results

This section discusses the results from Twitter activity using Chrome. These results can be regarded as case-specific artefacts that infer user activity. The artefacts were recovered using a range of the digital forensics tools discussed in **Section 3.6**.

Browser History

The recovery of digital evidence from the browser history was aimed at determining the type of information available, and how it can be recovered in a live triage. The analysis of browser history data highlights the type of artefacts that are of interest to this research. In order to analyse the generated data, a range of the tools discussed in **Section 3.6** were used for specific data types.

WinHex was used to extract the Chrome profile folder for the browser History. In Windows 7 onwards, this is located by default in %SYSTEMROOT%\Users\<username>\AppData\Local\Google\Chrome\UserData\Default. This folder contains the SQLite database files used by Chrome to store user activity; this includes but is not limited to navigation history, form-fill/auto complete, logins, downloads, cookies, and browser session information; an example is shown in Figure 5.1. Unlike Firefox where the History profile and cache directories are separate, this Default directory contains both the cache directory and history files.

Chrome creates individual profile folders for each Windows user account in the 'User Data' directory [Google, c]. Chrome also allows the creation of multiple profiles by navigating to `chrome://settings/manageProfile`, this allows a user to have separate profiles for different activities such as work, home, etc.; it also makes it possible for additional users such as children to make use of the existing chrome installation [Google, b].

In addition to the use of multiple profiles, Chrome has a sync feature where a user can sign into Chrome using a Google account, and is able to sync the browser

history, bookmarks etc. across multiple devices where the user is also signed in. The highlighted files in Figure 5.1 were considered of interest in this research, and the data they contain were analysed. The goal is to identify properties (features) that will be useful in inferring user action, and in relationship attribution during a live triage investigation. These are briefly discussed below.

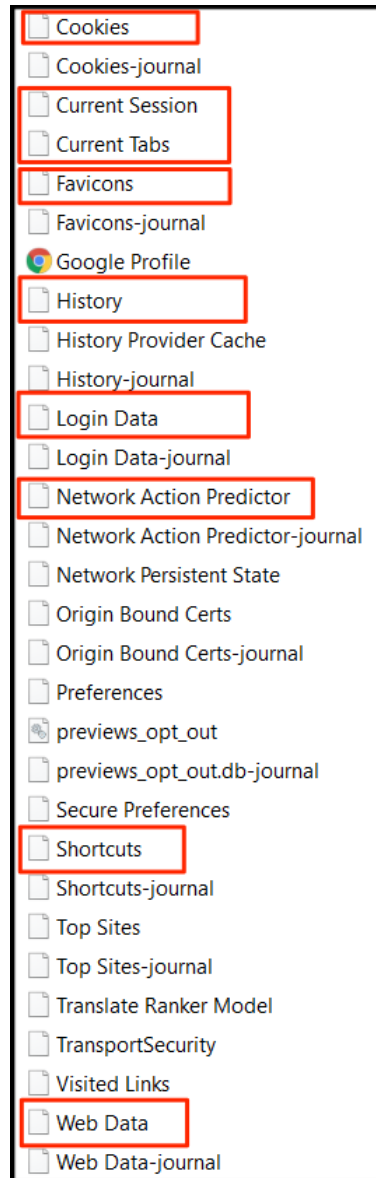


Figure 5.1: Chrome User Profile (History) contents

The files in the profile folder hold data of interest and forensic value; in Chrome, the name of the database (or file) is indicative of what is stored therein e.g. Cookies, History, Login Data. The files were analysed to guide the identification of artefacts of interest in a social media investigation, and to help with the implementation of their automated recovery in a live triage investigation. The analysed files are briefly

5.2. TWITTER

discussed below:

Cookies

This database contains two tables; *cookies*, and *meta*. It contains user settings for websites. Cookies are stored by Chrome, on behalf of websites visited by a user; they can be used to improve user experience, e.g. remembering user preferences on a website, and to identify the user on subsequent visits. The data of interest can be found in the *host_key*, *name*, and associated timestamp columns. Extracts of recovered cookies show cookies set by Twitter:

createdTime	lastAccessed	host_key	name	path	is_persistent
18/02/2019 21:56	18/02/2019 21:56	upload.twitter.com	lang	/	0
18/02/2019 21:41	18/02/2019 21:59	api.twitter.com	lang	/	0
18/02/2019 21:41	18/02/2019 21:59	.twitter.com	dnt	/	1
18/02/2019 21:59	18/02/2019 21:59	.twitter.com	personalization_id	/	1
18/02/2019 21:59	18/02/2019 21:59	.twitter.com	guest_id	/	1
18/02/2019 21:41	18/02/2019 21:59	syndication.twitter.com	lang	/	0
18/02/2019 21:40	18/02/2019 21:59	.twitter.com	_ga	/	1
18/02/2019 21:40	18/02/2019 21:59	.twitter.com	_gid	/	1
18/02/2019 21:59	18/02/2019 21:59	.twitter.com	_gat	/	1
18/02/2019 21:59	18/02/2019 21:59	.twitter.com	_twitter_sess	/	0
18/02/2019 21:40	18/02/2019 22:00	.twitter.com	ct0	/	1
18/02/2019 21:40	18/02/2019 22:00	.twitter.com	eu_cn	/	1
18/02/2019 21:41	18/02/2019 22:00	.twitter.com	kdt	/	1
18/02/2019 21:41	18/02/2019 22:00	.twitter.com	remember_checked_on	/	1
18/02/2019 21:41	18/02/2019 22:00	.twitter.com	csrf_same_site_set	/	1
18/02/2019 21:41	18/02/2019 22:00	.twitter.com	csrf_same_site	/	1

As discussed in **Chapter 4**, cookies are not necessarily an indication of a page visit as some cookies may have been created by adverts on a page, or other processes e.g. tracking cookies. During an investigation, other corroborating evidence should be used to establish the forensic value of a cookie.

Chrome History

The History database contains twelve tables; *downloads*, *downloads_slices*, *downloads_url_chains*, *keyword_search_terms*, *meta*, *segment_usage*, *segments*, *sqlite_sequence*, *typed_url_sync_metadata*, *urls*, *visit_source*, and *visits* as shown in Figure 5.2.

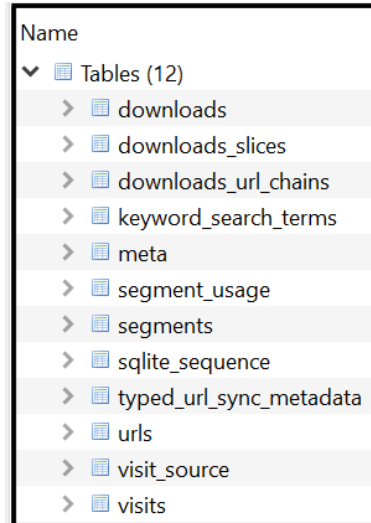


Figure 5.2: Tables in the Chrome History database

Valuable data can be recovered from these tables using targeted SQLite queries. For this research, the artefacts of interest in this database were URLs which present a timeline of user activity; e.g. Gmail account creation, Twitter account creation, verification of Twitter account, searching, communication, and navigating through Twitter. The activities inferred by the recovered artefacts were consistent with the actions recorded during the experiments. An extract from the History database is shown below (line breaks inserted):

visitTime	url	[title]
19/08/2018 10:48	https://www.google.com/gmail/about/#	[Gmail - Free Storage and Email from Google]
19/08/2018 11:03	https://mail.google.com/mail/?****bowib#inbox	[Inbox 1 - o*****@gmail.com - Gmail]
19/08/2018 11:05	https://twitter.com/	[Twitter]
19/08/2018 11:06	https://twitter.com/signup	[Twitter]
19/08/2018 11:13	https://twitter.com/i/flow/signup	[Twitter]
19/08/2018 11:15	https://mail.google.com/mail/?****bowib#inbox/[...]	[Confirm your Twitter account, Obed ***** - o*****@gmail.com - Gmail]

5.2. TWITTER

19/08/2018 11:15 https://twitter.com/account/confirm_user_email/103****
[Twitter]

19/08/2018 11:27 <https://twitter.com/search?q=goodfoodbbc&src=typd>
[BBC Good Food @bbcgoodfood | Twitter]

19/08/2018 11:34 <https://twitter.com/search?q=9gag&src=typd>
[9GAG @9GAG | Twitter]

19/08/2018 11:38 <https://twitter.com/search?q=9gag&src=typd>
[9GAG @9GAG | Twitter]

19/08/2018 11:38 <https://twitter.com/9GAG>
[9GAG @9GAG | Twitter]

19/08/2018 11:51 https://twitter.com/SYFY/status/103****
[SYFY on Twitter: "Despite the casualty rate, people still want to be pilots in the #StarWars galaxy."]

19/08/2018 11:58 https://twitter.com/SYFY/with_replies
[Media Tweets by SYFY @SYFY | Twitter]

19/08/2018 12:01 <https://twitter.com/i/moments>
[News - Twitter Moments]

19/08/2018 12:01 https://twitter.com/i/moments?category_id=1
[Sports - Twitter Moments]

19/08/2018 12:05 https://twitter.com/i/moments?category_id=3
[Fun - Twitter Moments]

[...excess data removed for brevity...]

19/08/2018 12:11 <https://twitter.com/logout>
[Twitter. It's what's happening.]

19/08/2018 12:11 https://twitter.com/?logged_out=1&lang=en
[Twitter. It's what's happening.]

The search terms typed into the search box in Twitter were found in the *keyword_search_terms* table; extracts below show the search term and the associated URL.

term	url
goodfoodbbc	https://twitter.com/search?q=goodfoodbbc&src=typd
9gag	https://twitter.com/search?q=9gag&src=typd
iamdeveloper	https://twitter.com/search?q=i*****r&src=typd

This information can be used to determine, and corroborate user search and possibly profile visit activities Twitter. These search terms match the URLs recovered from the urls and visits tables.

Login Data

This database stores user login credentials and contains three tables; *logins*, *meta*, and *stats*. The *logins* and *stats* tables contained data of forensic interest. In the *stats* table, the *origin_domain* column holds the URL while the *username_value* column holds the username for the website, in this instance the social media platform.

origin_domain	username_value	dismissal_count	updateTime
https://twitter.com/	sjos*****@gmail.com	1	18/02/2019 21:42

In the *logins* table, the data of interest recovered include the username, URL for the site where it is being used, number of times used, and created timestamp. It is important to note that although there were columns for password data, that information was encrypted.

createdDate	origin_url	username_value	signon_realm	times_used	date_synced
19/08/2018 11:10	https://twitter.com/	N**0bed	https://twitter.com/	1	0

Network Action Predictor

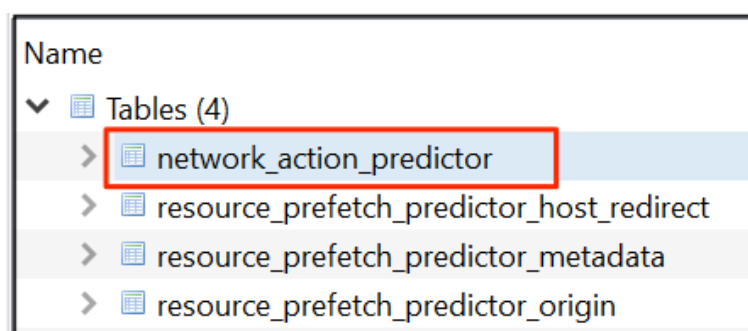


Figure 5.3: Network Action Predictor database

Chrome has a “prediction service” which is used to load links to pages the browser thinks the user might want [Google, b]; this data is stored in the *network_action_predictor* table, within the Network Action Predictor database (Figure 5.3). Suggestions are made whilst the user is typing; and two columns,

5.2. TWITTER

number_of_hits and number_of_misses, as their names imply, record hits or misses. It is important to note that some of the URLs found here are as a result of a page “prefetching” service. To determine actual visited URLs, the URL of interest should be corroborated by the URLs in the History database file:

user_text	url	number_of_hits	number_of_misses
twitter.c	http://twitter.com/search	0	1
twitter.c	http://twitter.com/onettdev	0	1
twitter.c	http://twitter.mobile/	0	1
twitter.c	http://twitter.com/colossaliscrazy	0	1
twitter.co	https://twitter.com/	1	0
twitter.co	http://twitter.co/	0	1
twitter.co	https://twitter.com/	1	0
twitter.co	http://twitter.com/lordjurrd	0	1
twitter.co	http://twitter.com/i/notifications	0	1
twitter.co	http://twitter.com/fortnitegame	0	1
twitter.co	https://twitter.com/	1	0
twitter.co	http://twitter.co/	0	1
twitter.co	http://twitter.com/search	0	1
twitter.co	http://twitter.com/onettdev	0	1
twitter.co	http://twitter.com/colossaliscrazy	0	1
twitter.com	https://twitter.com/	1	0

Shortcuts

This database file contains the *meta* and *omni_box_shortcuts* tables. Chrome uses the omnibox feature to auto complete text entries made by a user, or to auto-suggest possible entries to the user. The omnibox is the integration of the Google search bar functionality into the Chrome address bar. According to the Chromium project, the purpose of omnibox is “to merge both location and search fields while offering the user some highly relevant suggestions and/or early results” [The Chromium Projects, n.d]. Figure 5.4 shows an example of the omnibox functionality in Chrome.

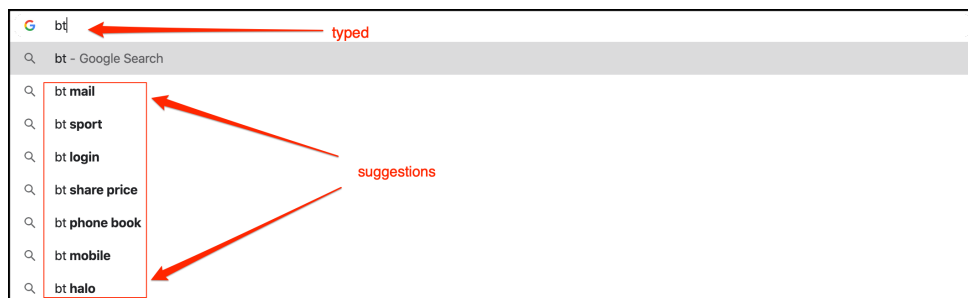


Figure 5.4: Example of Chrome’s omnibox functionality

For this research, the artefacts of interest in the *omni_box_shortcuts* table were

found in the text, urls, contents, description, last access time, number of hits, fill_into_edit, type, and keyword columns. An extract is shown in Table 5.2.

Table 5.2: Extract from the *omni_box_shortcuts* table

lastAccessed	text	fill_into_edit	url	contents	description	keyword	number_of_hits
19/08/2018 11:05	twitter.com	https://twitter.com	https://twitter.com/	twitter.com	Twitter		1

Web Data

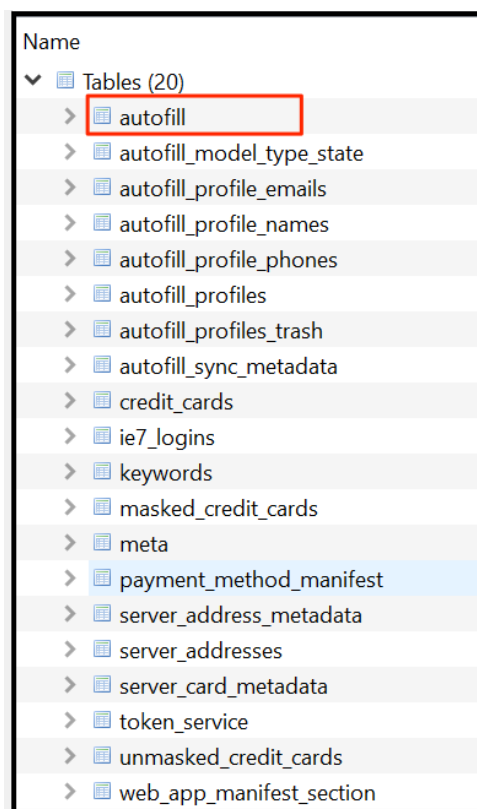


Figure 5.5: Web Data database

This database stores form data for websites and contains 20 tables as shown in Figure 5.5. The *autofill* table was of interest and the artefacts recovered include the username, value, created and last used timestamps. The data in the values column contained the user's Twitter username, and a phone number associated with the Twitter account. This table can be useful when recovering user formfill data, for example where a user is suspected of having multiple usernames for one or more social media platforms.

name	value	value_lower	createdDate	lastUsedDate
Username	o*****8	o*****8	19/08/2018 10:51	19/08/2018 10:51
phoneNumberId	7*****9	7*****9	19/08/2018 11:00	19/08/2018 11:02

5.2. TWITTER

Table 5.3: Extract from the *autofill* table

name	value	value_lower	createdDate	lastUsedDate
session[username_or_email]	sj***@gmail.com	sj***@gmail.com	18/02/2019 21:41	18/02/2019 21:41

Session Information

Chrome stores session information in four files:

- Current Session: this contains the data from the current session
- Current Tabs: this contains the tabs (and URL) information in the current session. e.g. Tab index etc.
- Last Session: this contains the similar information as the Current session, but the data held is about the previous session.
- Last Tabs: As with Current Tabs, but holds information about previous session

These files are used to restore previous sessions and tabs in the event of an unexpected termination of Chrome, or if the user closes the browser at the end of a session. At the time of writing, these files are stored in a proprietary binary format, although existing research [Bancel, Jean-Rémy, 2015; Caithness, 2012] shows they can be read by reverse engineering the source code in Chromium. These files can be identified by the file signature “SNSS”; already extracted as part of the History files, a Python tool Chromagnon by Bancel, Jean-Rémy [2015] was used to recover the data.

Extracts from the Current Session file:

```
UpdateTabNavigation - Tab: 2, Index: 2, Url: https://twitter.com/login
UpdateTabNavigation - Tab: 2, Index: 2, Url: https://twitter.com/login
SetWindowBounds3 - Window: 1, x: 10, y: 10, w: 1004, h: 708, State: 1
SetSelectedNavigationIndex - Tab: 2, Index: 2
[excess data removed for brevity]
SetSelectedNavigationIndex - Tab: 2, Index: 3
SetSelectedNavigationIndex - Tab: 2, Index: 4
UpdateTabNavigation - Tab: 2, Index: 4, Url: https://twitter.com/?logout=15**509
```

Extracts from Current Tab:

```
SelectedNavigationInTab - Tab: 4, Index: 4, Time: 13195000811559900
UpdateTabNavigation - Tab: 4, Index: 0, Url: chrome://newtab/
UpdateTabNavigation - Tab: 4, Index: 1, Url: https://twitter.com/
UpdateTabNavigation - Tab: 4, Index: 2, Url: https://twitter.com/login
UpdateTabNavigation - Tab: 4, Index: 3, Url: https://twitter.com/
UpdateTabNavigation - Tab: 4, Index: 4, Url: https://twitter.com/?logout=15**509
```

As seen in the data recovered from the session information files in Chrome, this file can be considered of value to digital forensic investigators. It shows the order in which the URLs were visited and also allows for the extraction of features from the stored URLs. There are also “Navigation” timestamps that can be used to correlate session activity with other artefacts recovered from Chrome. This data could be used, in addition to context and case specific artefacts to link a user to a particular Twitter account.

Browser Cache

The Chrome cache directory can be found in the same primary location (Default) as the History database files, but in a Cache directory: %SYSTEMROOT%\Users\<username>\AppData\Local\Google\Chrome\UserData\Default\Cache. When a user visits a website, Chrome temporarily stores images, scripts and other content from the visited website; this enhances efficiency and allows the browser to check if those resources are locally available (on disk), and up to date before downloading them again if that website is revisited.

The Chrome cache is usually comprised of five files - an index file and 4 data files (*data_0* - *data_3*); the index file contains addresses that point to a data file containing cached items. Separate data files may be stored for websites if the cached content is larger than 16KB [The Chromium Projects]. The naming convention for these additional files is a 6 digit hexadecimal number prefixed by ‘f_’. To analyse the contents of the cache, the cache directory was extracted using WinHex and as part of the analysis, NirSoft [2018a] ChromeCacheView was used to view the contents in the cache.

The artefacts of interest during the analysis were those indicative of a user’s direct interaction with other test users, for example, cached profile information such as profile banners, or profile photo URLs and images. These results appeared to corroborate the actions taken during the experiments e.g. visiting profiles, searching for users etc.

Profile information similar to what was seen in the Firefox cache [Section 4.2.2] were recovered in the Chrome cache; for example, profile photo URLs and Twitter profile ID. This data can be corroborated by, or used to corroborate other artefacts found on disk as a result of accessing Twitter via Chrome browser.

5.2. TWITTER

```
=====
Filename      : dFVqk-A3_normal.jpg
URL           : https://pbs.twimg.com/profile_images/103****712
               /dFVqk-A3_normal.jpg
Content Type   : image/jpeg
File Size     : 2,567
Last Accessed  : 18/02/2019 21:56:00
Server Time    : 18/02/2019 21:41:50
Server Last Modified: 19/08/2018 10:00:41
Server Name    : ECS lcy/1D59
Server Response : HTTP/1.1 200 OK
Cache Name     : data_2 [266240]
Cache Control  : max-age=604800, must-revalidate
URL Length    : 76
=====
=====
Filename      : 9nkteNKb_normal.jpg
URL           : https://pbs.twimg.com/profile_images/103****616
               /9nkteNKb_normal.jpg
Content Type   : image/jpeg
File Size     : 2,187
Last Accessed  : 18/02/2019 21:55:42
Server Time    : 18/02/2019 21:47:08
Server Last Modified: 19/08/2018 12:21:50
Server Name    : ECS lcy/1D64
Server Response : HTTP/1.1 200 OK
Cache Name     : data_2 [1073152]
Cache Control  : max-age=604800, must-revalidate
URL Length    : 76
=====
=====
Filename      : 7kGs1e0U_normal.jpg
URL           : https://pbs.twimg.com/profile_images/103****112
               /7kGs1e0U_normal.jpg
Content Type   : image/jpeg
File Size     : 2,263
Last Accessed  : 18/02/2019 21:55:38
Server Time    : 18/02/2019 21:55:40
Server Last Modified: 19/08/2018 16:41:09
Server Name    : ECS lcy/1D71
Server Response : HTTP/1.1 200 OK
Cache Name     : data_2 [8384512]
Cache Control  : max-age=604800, must-revalidate
URL Length    : 76
=====
```

```

=====
Filename      : 600x200.jpg
URL           : https://pbs.twimg.com/profile_banners/103****2545
               /1534669327/600x200
Content Type  : image/jpeg
File Size     : 26,380
Last Accessed : 18/02/2019 21:59:09
Server Time   : 18/02/2019 21:41:50
Server Last Modified: 19/08/2018 10:00:07
Server Name   : ECS lcy/1D56
Server Response : HTTP/1.1 200 OK
Cache Name    : f_000022
Cache Control  : max-age=604800, must-revalidate
URL Length    : 76
=====

```

As discussed in **Section 4.2.2**, content may be cached even when a user has not directly interacted with it; for example, sponsored content on Twitter as shown below.

```

=====
Filename      : advertiser_name=Pluralsight&cardname=unified_card
               &is_following.html
URL           : https://twitter.com/i/cards/tfw/v1/109****84?
               advertiser_name=Pluralsight&cardname=unified_card&is_following
               _advertiser=false&forward=true&impression_id=1477adb9f250cc5
               &edge=true&lang=en&card_height=271&scribe_context
               =%7B%22client%22%3A%22web%22%2C%22page%22%3A%22home%22
               %2C%22section%22%3A%22home%22%2C%22component%22%3A%22
               tweet%22%7D&bearer_token=AAA[SNIP]qaAxGfsyKCs0wRbw
Content Type  : text/html
File Size     : 1,370
Last Accessed : 18/02/2019 21:45:35
Server Time   : 18/02/2019 21:45:36
Server Name   : tsa_f
Server Response : HTTP/1.1 200 OK
Content Encoding : gzip
Cache Name    : data_2 [1015808]
URL Length    : 476
=====
=====
Filename      : 9
URL           : https://video.twimg.com/amplify_video/107****m3u8?tag=9
Content Type  : application/x-mpegURL

```


5.3. FACEBOOK

```
File Size      : 273
Last Accessed  : 18/02/2019 21:55:06
Server Time    : 18/02/2019 21:44:09
Server Last Modified: 29/12/2018 01:49:03
Server Name    : ECAcc 1ha/8C8E
Server Response : HTTP/1.1 200 OK
Content Encoding : gzip
Cache Name     : data_1 [79872]
Cache Control  : max-age=604800, must-revalidate
URL Length    : 88
=====
=====
Filename      : AmazonPrimeDay_2018.png
URL           : https://abs.twimg.com/hashflags/AmazonPrimeDay_2018
               /AmazonPrimeDay_2018.png
Content Type  : image/png
File Size    : 2,170
Last Accessed : 19/08/2018 12:55:52
Server Time   : 19/08/2018 12:55:59
Server Last Modified: 06/07/2018 05:13:00
Expire Time   : 19/08/2019 12:55:59
Server Name   : ECS lcy/1D3D
Server Response : HTTP/1.1 200 OK
Cache Name    : data_2 [6836224]
ETag         : "HygKqBGOD5jhbLo9P1N5WQ=="
URL Length    : 75
=====
```

5.2.3 Summary

The experiments and analysis conducted in this section look to address the key problems identified in **Section 3.3**; concerning the recovery of artefacts that could be used where communication content may not be immediately accessible. **Section 4.5** discusses how the Twitter experiments meet the objectives and addresses the research question [**Section 3.4**].

5.3 Facebook

This section discusses the experiments and results from Facebook activity using Google Chrome. The results discussed in this section are from the manual recovery processes using a range of the traditional digital forensics tools discussed in **Section 3.6**.

5.3.1 Experiments

This section details the data generation and collection process. Experiments were conducted using each of the three user accounts created on the experimental VMs [Section 3.9.2]. As discussed in Section 3.6.1, Procmon and Fiddler were also used to log system activity and capture HTTPS traffic respectively. The same categories of Facebook activities described in [Section 4.3], make up the experiments on Google Chrome.

Registration and Authentication

In order to connect with, and exchange messages with other Facebook users, a registered account is required. To simulate the account registration process for Facebook, accounts were set up for the Windows users created for this experiment [Section 3.9.2]. A valid email address and a mobile phone number were used to verify the Facebook accounts used for these experiments.

Searching and Viewing Content

Unregistered individuals are able to access Facebook through search engines, and can subsequently view publicly available content however, in order to engage with other users, users have to be logged in.

Facebook search bar was used to search for the users created during this experiment, and for public profiles and groups. This category of experiments mirrors the Facebook experiments conducted using Firefox [Section 4.3.1].

Sharing Content

Sharing content on Facebook involves posting text, video, photos, links etc. on a user profile. A user can like, share or reply posts from other users. Facebook provides chat and messaging features which allow users to exchange content privately. Users can also communicate using hashtags to highlight topics of interest. User can also communicate using hashtags highlight topics of interest [Facebook Help Centre, 2019b].

For this category, a subset of activities [Section 4.3.1] were simulated to establish a baseline for how Facebook could be used.

Friending

Friending is a way Facebook users form social relationships and associations with other users. It involves sending a friend request to another Facebook user; both sender and recipient become friends when the request is accepted by the recipient

5.3. FACEBOOK

[**Section 4.3.1**]. The privacy settings on a Facebook account are used to define what other users can view or do on a profile.

Just as Twitter, Facebook has a ‘follow’ process which allows users to connect with people or things they find interesting e.g. big brands, sports clubs, etc. This category of experiments mirror the ones conducted using Firefox.

5.3.1.1 Availability and Recoverability of Artefacts

As done with the Firefox experiments, it was necessary to pause the VM and copy the virtual disk to a working folder; limit the number of potential use cases covered in the experiments due to the potential number of usage scenarios; and verify the availability, recoverability, and validity of previously recovered artefacts, by repeating a range of the outlined activities at the end of the research [**Section 4.3.1.1**].

5.3.2 Results

This section contains the results for Facebook activity using Google Chrome. These results can be regarded as case-specific artefacts inferring user activity. The results in this section were recovered using a range of the digital forensics tools discussed in **Section 3.6**.

Browser History

Evidence recovery from Chrome was aimed at determining the type of information that could be recovered, and its availability in a live triage investigation. The analysis of browser history data highlights the type of artefacts that are of interest to this research. To analyse the data related to a user’s browser history, a range of the tools [**Section 3.6**] were used for specific data types and are briefly discussed below.

The location of Chrome history data was briefly discussed in **Section 5.2.2**. The user virtual disk (*.vmdk*) file was opened using WinHex and the Chrome profile directory was extracted. It is important to note that there will be similarities in the location and category of artefacts recovered from the VMs used for social media activities in Chrome. As discussed in **Section 3.10.1**, it is possible to use attributes from URLs to contextualise user activity.

The database files in the ‘Default’ profile directory were analysed using DB Browser for SQLite; the recovered artefacts were exported and saved as a CSV file. Exporting to CSV made it easier to filter and manipulate the data (sorting, grouping, searching), when identifying properties that could be used for relationship attribution. This also made it easier to corroborate these artefacts with those recovered from other SQLite databases. The analysed files are briefly discussed below.

Cookies

This database file contains two tables; *cookies* and *meta*. The data of interest can be found in the *host_key*, *name*, and associated *timestamp* columns. Certain types of cookies as shown in Table 5.4 are not necessarily as a result of a user’s visit to a website but may be set by embedded content on the visited page. It is important to use other pieces of evidence to support cookie artefacts when attributing actions to a user, or establishing relationships.

Table 5.4: Extract from the Cookies database file

host_key	name	value	path	is_persistent	creation_utc	last_access_utc
.www.google.com	PAIDCONTENT		/insight/consumersurveys	1	05/11/2018 11:21	05/11/2018 11:21
.google.com	AID		/ads	1	05/11/2018 11:23	05/11/2018 11:23

Chrome History

Recovering data from the History database file requires the use of targeted SQLite queries for specific types of data in any of the twelve tables it contains. For this research, the artefacts of interest in this database were those that provide a sequence/-timeline of user activity; for example, account verification, viewing and exchanging content, searching and navigation through Facebook. The artefacts recovered were consistent with the actions taken during the experiments, thus, it can be inferred that they are representative of user activity likely to occur on Facebook as shown in the extracts below:

<https://www.facebook.com/confirmemail.php?next=https%3A%2F%2Fwww.facebook.com>

[https://accounts.google.com/o/oauth2/\[...\]
&redirect_uri=https%3A%2F%2Fwww.facebook.com\[...\]&scope=openid+
email&login_hint=sm***%40gmail.com](https://accounts.google.com/o/oauth2/[...]&redirect_uri=https%3A%2F%2Fwww.facebook.com[...]&scope=openid+email&login_hint=sm***%40gmail.com)

The extracts above indicate a newly created Facebook account awaiting confirmation, and it can be inferred that the user has clicked on the link on the Facebook account registration page ([confirmemail.php?next=https%3A%2F%2Fwww.facebook.com](https://www.facebook.com/confirmemail.php?next=https%3A%2F%2Fwww.facebook.com)), and has been redirected to the registered email account ([https://accounts.google.com/o/oauth2/\[...\]](https://accounts.google.com/o/oauth2/[...]&redirect_uri=https%3A%2F%2Fwww.facebook.com[...]&scope=openid+email&login_hint=sm***%40gmail.com)) and then back to Facebook (`\&response_type=code&redirect_uri=https://www.facebook.com`). This is useful in a scenario where an entity (as defined in **Section 2.8**) is suspected of using a newly created, and perhaps alternate account(s) for disagreeable or unlawful activities on Facebook. This can be used to tie the entity to a particular Facebook account; this can be achieved by comparing the email address “*sm***@gmail.com*” seen in the URL with other formfill and login artefacts.

5.3. FACEBOOK

Every Facebook profile has an identifier; this could be a numeric or alphanumeric identifier e.g. *1000*****1748*, *seth.m***.123*. These identifiers can be used to infer user action on Facebook; for example where entity_a intends to communicate with entity_b; entity_a searches for entity_b on Facebook, and then proceeds to visit entity_b's profile. Although the artefacts recovered do not show the content of communication, they 'tell a story' and can be used to determine what has occurred. Extracts from the History database (shown below) can be used to determine who entity_a has searched for; whether or not they visited entity_b's page afterwards. Using the examples below, it shows it is possible to search for another Facebook user by name, email address, phone number or profile ID; in the same manner, it is possible to use these properties to attribute actions and relationship to an entity.

```
url          [title]
https://www.facebook.com/profile.php?id=100*****748
[Joshua S*****]

https://www.facebook.com/find-friends/browser/
[Facebook]

https://www.facebook.com/search/top/?q=sm****%40gmail.com&epa=SEARCH_BOX
[sm*****@gmail.com, Æi Facebook Search]

https://www.facebook.com/search/top/?q=ok****%40gmail.com&epa=SEARCH_BOX
[ok****@gmail.com, Æi Facebook Search]

https://www.facebook.com/search/top/?q=eric%20cartman&epa=SEARCH_BOX
[eric cartman, Æi Facebook Search]

https://www.facebook.com/profile.php?id=107****835
[eric cartman, Æi Facebook Search]

https://www.facebook.com/OfficialCartman/
[Eric Cartman - Home]

https://www.facebook.com/pg/OfficialCartman/about/?ref=page_internal
[Eric Cartman - About]

https://www.facebook.com/pg/OfficialCartman/photos/?ref=page_internal
[Eric Cartman - Photos]

https://www.facebook.com/pg/OfficialCartman/posts/?ref=page_internal
[Eric Cartman - Posts]
```

[https://www.facebook.com/login/device-based/regular/logout/?button_name=logout
&button_location=settings](https://www.facebook.com/login/device-based/regular/logout/?button_name=logout&button_location=settings)
[Facebook, Äi log in or sign up]

https://www.facebook.com/search/top/?q=07*****0&epa=SEARCH_BOX
[07*****0, Äi Facebook Search]

https://www.facebook.com/search/top/?q=seth.micah.***&epa=SEARCH_BOX
[2 seth.m****.***, Äi Facebook Search]

https://www.facebook.com/seth.m****.***
[3 Seth ****]

Favicons

This database file contains four tables; *favicons_bitmap*, *favicons*, *icon_mapping*, and *meta*. It is used to store and map icons to a user's bookmarks, visited webpages etc. The data stored in Favicons can provide context and be used to corroborate other history related artefacts inferring a user's activities on Facebook; for example, setting Bookmarks may indicate frequent access.

Login Data

This database file is used to store user login credentials, and contains three tables; *logins*, *meta*, and *stats*. The *logins* and *stats* tables contained data of forensic interest. In the *stats* table, the *origin_domain* column holds the URL while the *username_value* column holds the username for the visited website. The *logins* table contained data of interest in the *origin_url*, *username_element*, and *username_value* columns as shown in Table 5.5.

Table 5.5: Extract from the *logins* table

origin_url	username_element	username_value	signon_realm
https://www.facebook.com/	https://www.facebook.com/login/device-based/regular/login/	sm****@gmail.com	

Network Action Predictor

This database is used to store information related to Chrome's page pre-loading/network prediction functionality as discussed in **Section 5.2.2**. Example predictions from Facebook experiments include:

user_text	url	number_of_hits	number_of_misses
www.facebook.c	https://www.google.co.uk/search?q=www.facebook.com****	0	1
www.facebook.com	http://www.facebook.com/	1	0

5.3. FACEBOOK

```
www.facebook.com https://www.google.co.uk/search?q=www.facebook.com&**** 0 1
www.facebook.com http://www.facebook.com/device 0 1
www.facebook.com http://www.facebook.com/recover.php 0 1
```

Shortcuts

As discussed in the Twitter results section in this Chapter [Page 118], the artefacts of interest in the *omni_box_shortcuts* table were found in the text, urls, contents, description, last access time, number of hits, fill_into_edit, type, and keyword columns as shown below.

```
lastAccessed text fill_into_edit url contents description keyword number_of_hits
05/11/2018 11:47 www.facebook.com www.facebook.com http://www.facebook.com/
www.facebook.com 1
```

Web Data

This database is similar in function to Firefox's *formhistory.sqlite*; and stores form data for websites. The artefacts in this database could be used to attribute or determine account ownership [Page 119]. For example, if the same username e.g. an email address or mobile phone number is used across multiple platforms, it may be linked to the 'known' owner of that email address. Artefacts extracted show the known usernames for Facebook, stored by Chrome.

name	value	createdDate	lastUsedDate
reg_email_confirmation__	sjos****@gmail.com	05/11/2018 10:30	05/11/2018 10:30
email	sjos****@gmail.com	05/11/2018 11:22	25/11/2018 10:43

Session Information

Chrome stores session information in these four files: Current Session, Current Tabs, Last Session, and Last Tabs [Page 120]. Extracts recovered from the Session information files for Facebook experiments are shown below.

Current Session file:

```
Transition: 0
Title: Facebook
State: https://www.facebook.com/home.php?ref=wizard
https://www.facebook.com/gettingstarted/
https://www.facebook.com/home.php?ref=wizard
https://www.facebook.com/find-friends/index.php
https://upload.facebook.com/ajax/timeline/profile_pic_upload
https://www.facebook.com/search/top/
https://www.facebook.com/search/web/direct_search.php
```

Last Tabs file:

UpdateTabNavigation - Tab: 281282675, Index: 5,
 URL: <https://www.facebook.com/profile.php?id=100...748>,
 Title: Joshua S*****

UpdateTabNavigation - Tab: 281282675, Index: 6,
 URL: https://www.facebook.com/profile.php?id=100***748&sk=friends&source_ref=pb_friends_tl,
 Title: None,
 State: None

UpdateTabNavigation - Tab: 281282675, Index: 7,
 URL: <https://www.facebook.com/find-friends/browser/>,
 Title: None,
 State: None

UpdateTabNavigation - Tab: 281282675, Index: 8,
 URL: https://www.facebook.com/search/top/?q=sm***%40gmail.com&epa=SEARCH_BOX,
 Title: None,
 State: None

UpdateTabNavigation - Tab: 281282675, Index: 9,
 URL: https://www.facebook.com/search/top/?q=ok***%40gmail.com&epa=SEARCH_BOX,
 Title: None,
 State: None

UpdateTabNavigation - Tab: 281282675, Index: 10,
 URL: https://www.facebook.com/search/top/?q=eric%20cartman&epa=SEARCH_BOX,
 Title: eric cartman Facebook Search,
 State: None

The artefacts recovered validate the data recovered from the History database file. The Current/Last Session file keeps a record of activities that occur within a specific Chrome session, in this instance, Facebook visits. The Current/Last Tabs file keeps a record of the same information but also indexes the tabs to show the sequence activities; as seen in the indexing in the extracts above. This data also corroborates what was recovered as part of the browser history. From the data available in the Session data files, it is possible to extract features from that could be used for attribution.

Browser Cache

Profile information similar to what was seen in the Firefox cache [Section 4] was recovered from the Chrome cache; the artefacts of interest during the analysis were those indicative of a user's direct interaction with other test users, e.g. profile photos

5.4. GOOGLE DRIVE

with their associated URLs and filenames, shared, viewed or downloaded content, etc. In order for an investigator to support the notion of a profile visit, the cache artefacts can be corroborated by, or used to corroborate other artefacts found on disk.

It was observed that Facebook profile photo URLs in the cache had a filename prefixed with `_nc_cat`, for example, `_nc_cat=107&_nc_ohc=PacM****jM&_nc_ht=scontent-lhr8-1.jpg`; the accompanying URL appears to follow this format: `https://scontent-lhr8-1.xx.fbcdn.net/v/t1.0-1/p160x160/5243****7232_n.jpg`. These properties may be used to narrow the search for, and recover cache artefacts using RegEx. As with other browser artefacts, it is important to note that the profile images from other Facebook users who were not part of these experiments were inadvertently cached by Chrome; such data has been excluded from published results.

5.3.3 Summary

The research objectives were discussed in **Section 3.4**, and the artefacts discussed in this section address the key problems identified in **Section 3.3**; with regards to recovering artefacts that could be used in a live triage investigation where the contents of communication may not be immediately accessible. **Section 4.5** discusses how this experiment meets the objectives and addresses the research question.

5.4 Google Drive

This section discusses the experiments and results from Google Drive activity using Chrome. The results discussed in this section are from the manual recovery processes, using a range of the traditional digital forensics tools discussed in **Section 3.6**.

5.4.1 Experiments

This section discusses the data generation and collection process. As with Twitter and Facebook, experiments were conducted using each of the three user accounts created for this research [**Section 3.9.2**]. These experiments for Google Drive mirror the actions described for Firefox in [**Section 4.4.1**]. As discussed in **Section 3.6.1**, Procmon and Fiddler were used to log system activity, and used to capture HTTPS traffic respectively.

Create and Store

Users on Google Drive (hereafter referred to as Drive) can create a variety of files in various formats, and files stored on other devices can be uploaded and stored in Drive [**Section 4.4.1**].

The first set of activities involved the creation of documents locally and uploading them to Drive through a folder shared with the VM. This was followed by the creation of new documents and folders in Drive using the *New* button and selecting the document format.

These activities were performed to simulate user activity; and will highlight the artefacts created as a result of Drive activity, which may be of interest during a forensic investigation.

Document Access and Update

Documents stored in Drive can be accessed from any web browser or digital device that has Drive installed. When changes are made to stored documents, they are synced, making them available when the document is accessed by another user with access, or from a different device [Section 4.4.1].

The experiments conducted involved viewing and modifying the stored documents in Drive. The activities in this category also involved downloading files and folders, adding files to folders, and removing (deleting) files. These were done to simulate how a user may interact with the content in Drive.

Share and collaborate

Drive users share files and folders with each other; the access levels on shared content can be set by the content owner, permitting other users with access to comment on, or modify the shared documents. There are three levels of access: can edit, can comment, and can view [Page 99].

For this experiment, files and folders were shared between the three user accounts. Modifications were made to shared documents, and the shared documents were searched for using the search bar in Drive.

5.4.1.1 Availability and Recoverability of Artefacts

As done with the previous experiments [Sections 5.2.1.1; 5.3.1.1] in this Chapter, it was necessary to pause the VM and copy the virtual disk to a working folder; and also to verify the availability, recoverability, and validity of previously recovered artefacts, by repeating a range of the outlined activities at the end of the research.

The nature of user social media makes the provision for multiple usage scenarios that could occur with regards to how a user would interact with others using social media. For example, the creation and use of shared drives, using Drive through the browser alongside the Desktop application, and offline access to Drive. The activities

5.4. GOOGLE DRIVE

outlined in [Section 5.4.1] were selected as a baseline due to the time constraints that would be imposed on the research if all possible scenarios were to be covered in the research experiment.

5.4.2 Results

This section contains the results for Drive activity using Chrome. These results can be regarded as case-specific artefacts inferring user activity. A range of the digital forensics tools discussed in Section 3.6 were used for the manual recovery and analysis of the results in this section.

Browser History

The location and contents of the Chrome History profile directory were discussed in Sections 5.2.2 and 5.3.2. As these set of experiments were also conducted using Chrome, there will be similarities in the location and category of artefacts recovered for all three social media platforms discussed in this Chapter. The analysis of browser history data from Drive activity highlights the type of artefacts that are of interest to this research. The experimental results analysed are briefly discussed below.

Cookies

For these set of experiments, the following cookies were set for `.docs.google.com` when documents were created or accessed:

```
host_key  name value path
.docs.google.com S /document/u/0/d/1Xbse0PyM8Ms****4rkmbnbn0e49tsmxxaAo
.docs.google.com S /spreadsheets/u/0/d/1NHaas-DGH****nL0kgod2Z7uBTTEuHXIT7HV
.docs.google.com S /presentation/u/0/d/1Ku8ALsA9dS****BVMT3azZhhWfExQJ6xMKNKY
.docs.google.com S /presentation/d/1Ku8ALsA9dS****BVMT3azZhhWfExQJ6xMKNKY
.docs.google.com S /comments/u/11595****565/d/AAHRpnXui-****D5W6HUXgp-ByW6g
```

This information can be used to understand when Drive was visited, and can be corroborated by or used to corroborate other related artefacts.

Chrome History

The History database file contains data of forensic interest and value. Recovering data from this database requires the use of targeted SQLite queries for specific artefacts. To reiterate, the focus at this stage of the research is on artefacts that show a sequence of user activity such as creating and accessing documents on Google Drive. From the extracts below, user activity can be inferred using the URL and page title information.

5.4. GOOGLE DRIVE

url [title]

https://drive.google.com/?usp=chrome_app

[Google Drive: Free Cloud Storage for Personal Use]

<https://www.google.com/intl/en-GB/drive/>

[Google Drive: Free Cloud Storage for Personal Use]

https://drive.google.com/?utm_source=en-GB&authuser=0

[Google Drive]

<https://drive.google.com/drive/my-drive>

[Google Drive: Sign-in]

https://docs.google.com/document/create?usp=drive_web&ouid=****&folder=0A****

[Untitled document - Google Docs]

https://docs.google.com/document/u/0/create?usp=drive_web&folder=0A****

[Untitled document - Google Docs]

https://docs.google.com/spreadsheets/u/0/?ftv=1&usp=drive_web_tmpl&folder=0A****

[Google Sheets]

https://docs.google.com/presentation/u/0/create?usp=drive_web&folder=0A****

[Untitled presentation - Google Slides]

https://docs.google.com/presentation/d/1K****NKY/edit?usp=drive_web&ouid=****

[Untitled presentation - Google Slides]

<https://drive.google.com/drive/my-drive>

[Google Drive: Sign-in]

<https://drive.google.com/drive/shared-with-me>

[Shared with me - Google Drive]

<https://drive.google.com/drive/computers>

[Computers - Google Drive]

https://drive.google.com/drive/folders/1qz****cEmSMqV

[TV show extracts - Google Drive]

https://drive.google.com/drive/folders/1T4Q****o2pboccZ

[photography - Google Drive]

5.4. GOOGLE DRIVE

Login Data

This database stores user login credentials. The *logins* table contained data of interest in the `origin_url`, `username_element`, and `username_value`, `signon_realm` and `possible_username_pairs` columns.

```
origin_url username_element username_value
https://accounts.google.com/SignUp https://accounts.google.com/ sm***
```

```
signon_realm
https://accounts.google.com/
```

```
possible_username_pairs
"M** _lastName_ Seth firstName"
```

The data recovered from the *logins* table shows the possibility of recovering the username, URL for where it is being used; the registered first and last names, as seen in the `possible_username_pairs` columns. The encrypted password used with the username is also stored in this table.

Web Data

The artefacts in the database can be used to attribute, or determine account ownership [Page 119]. For example, if the same username e.g. an email address or mobile phone number is used across multiple platforms which are accessed frequently, it may be possible, using other corroborating evidence, to associate the ‘known’ owner of that phone number or email address to the accessed account(s).

Extracts from the *autofill* table in the Web Data database file show the known registered phoned number, stored by Chrome for user account verification during the experiments as seen below and in Figure 5.6.

```
name          value      value_lower
phoneNumberId 7*****2 7*****2
```

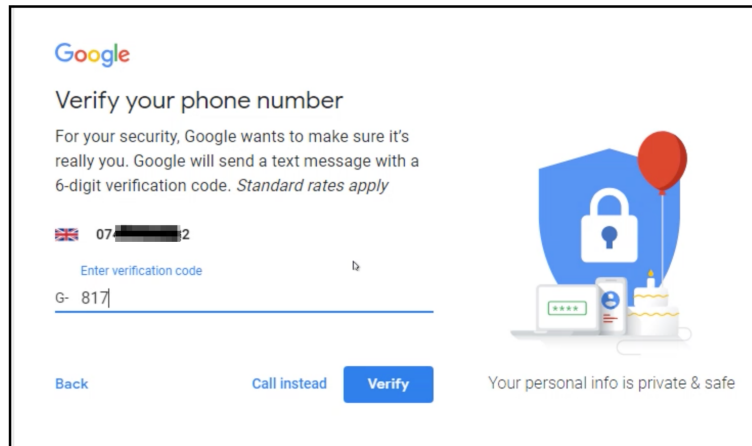


Figure 5.6: Google account verification with a mobile number

Session Information

As discussed in **Section 5.2.2**, Chrome stores session information in four files. These files were extracted from the Default profile folder using WinHex and Chromagnon [Bancel, Jean-R  my, 2015] was used to recover the data.

Extracts of artefacts recovered are shown below (line breaks inserted):

UpdateTabNavigation - Tab: 7, Index: 0,

Url: https://docs.google.com/document/d/1Xbse****kmbn0e49tsmxxaAo/edit

SelectedNavigationInTab - Tab: 9, Index: 1, Time: 13195085267684244

UpdateTabNavigation - Tab: 9, Index: 0,

Url: https://docs.google.com/spreadsheets/u/0/?ftv=1&folder=0AC****PVA

UpdateTabNavigation - Tab: 9, Index: 1,

Url: https://docs.google.com/spreadsheets/d/1NHaas-DGH****T7HVs/edit#gid=****

SelectedNavigationInTab - Tab: 11, Index: 0, Time: 13195085468745629

UpdateTabNavigation - Tab: 11, Index: 0,

Url: https://docs.google.com/presentation/d/1Ku****MNKY/edit#slide=id.g4f****

SelectedNavigationInTab - Tab: 13, Index: 0, Time: 13195085534115061

UpdateTabNavigation - Tab: 13, Index: 0,

Url: https://docs.google.com/presentation/d/1Ku****QJ6xMNKY/edit#slide=id.p

SelectedNavigationInTab - Tab: 15, Index: 0, Time: 13195085562563160

As seen in the extracts above, some intelligence can be gathered from the session information files by inferring the activities that took place. This information can be used to corroborate other artefacts recovered. In a time sensitive investigation where an investigator needs to know if there has been any form of contact between entity_a and entity_b, the artefacts in these session files can be used to make a

5.4. GOOGLE DRIVE

determination.

Artefacts recovered show the user accessing documents on Google drive (e.g. `docs.google.com/spreadsheets`). From this, it is possible to extract attributes in the URL such as `'docs.google.com/presentation/d/*'` using RegEx; subsequently, this information may be used to infer user activity such as creating, viewing, or editing a document.

Browser Cache

The location of the Chrome cache directory was discussed in **Section 5.2.2**. There are other artefacts that could be recovered from the cache, for example video, audio, images, text files etc., however, this research is focused on identifying potential sources of intelligence for a live triage investigation. Thus, the artefacts of interest was narrowed down to those pieces of evidence that do not require an extensive amount of time to recover, interpret, and utilise.

Cache artefacts can be used to determine user relationships however, content from background processes may be cached on disk. In this context, the presence of these artefacts in the cache does not necessarily equate to a visit; to make a distinction between an unrelated process and an actual visit, other artefacts should be used to provide context and as corroborating evidence.

Extracts found in the Chrome cache are shown below, and indicate the creation of a file, and a folder by the user.

```
=====
Filename       : usp=drive_web&oid=115****65&folder=0AC****html
URL            : https://docs.google.com/document/create?usp=drive_web&
                oid=115****565&folder=0AC****9PVA&authuser=0
File Size      : 0
Content Type   : text/html
Last Accessed  : 19/02/2019 21:21:18
Server Time    : 19/02/2019 21:21:20
Expire Time    : 19/02/2019 21:21:20
Server Name    : GSE
Server Response : HTTP/1.1 302 Moved Temporarily
Content Encoding : gzip
URL Length     : 118
=====
```

```

=====
Filename      : usp=drive_web&folder=0AC****9PVA&authuser=0.html
URL           : https://docs.google.com/document/u/0/create?usp=drive_web&
               folder=0AC****9PVA&authuser=0
File Size     : 0
Content Type  : text/html
Last Accessed : 19/02/2019 21:21:18
Server Time   : 19/02/2019 21:21:20
Expire Time   : 19/02/2019 21:21:20
Server Name   : GSE
Server Response : HTTP/1.1 302 Moved Temporarily
Content Encoding : gzip
URL Length    : 95
=====

```

Setting permissions on the contents: user Josh tries to open a shared folder through a link sent by email, access is denied as permission was granted for a document, not the parent folder [see **page 103** for extract where the email was sent].

```

=====
Filename      : openDrive=false&reason=112&syncType=0&errorRecovery=false&fiel
URL           : https://clients6.google.com/drive/v2internal/files/0AC****VA
               /permissions?openDrive=false&reason=112&syncType=0&error
               Recovery=false&fields=kind%2Citemkind%2Ctype%2Cid%2Cname
               emailAddress%2CwithLink%2CphotoLink%2Crole%2CadditionalRoles
               %2Cdeleted%2Cview%2CnextPageToken&maxResults=100&
               supportsTeamDrives=true&key=AIza****3_l0aBk4
Content Type  : application/json
File Size     : 182
Last Accessed : 19/02/2019 21:40:26
Server Time   : 19/02/2019 21:40:30
Expire Time   : 19/02/2019 21:40:30
Server Name   : GSE
Server Response : HTTP/1.1 200
Content Encoding : gzip
Cache Name    : data_1 [83968]
ETag          : "KEjgAPY1My092hX6wg1WtfsBoI8/X9W6S0mRS7u6mWECu-Cqz\_EF5mQ"
URL Length    : 364
=====

```

Looking at the extract above, the file containing this data is stored within one of the cache data files (**data_1**). This cache file was extracted using ChromeCacheView and explored further. The file contained JSON format data which can be used to attribute ownership of a Google Drive account to a user as shown below.

5.4. GOOGLE DRIVE

Cache file: openDrive=false&reason=112&syncType=0&errorRecovery=false&fiel

```
{
  "kind": "drive#permissionList",
  "items": [
    {
      "kind": "drive#permission",
      "id": "09363****9819",
      "name": "Josh S****",
      "emailAddress": "sjosh...@gmail.com",
      "role": "owner",
      "type": "user",
      "deleted": false
    }
  ]
}
```

A number of cache files with similar names (openDrive=*) were identified and analysed. An extract from one of the files examined is shown below.

Cache file: openDrive=true&reason=301&syncType=0&errorRecovery=false&field~1

```
{
  "kind": "drive#about",
  "user": {
    "kind": "drive#user",
    "displayName": "Josh S****",
    "isAuthenticatedUser": true,
    "id": "1159****565",
    "permissionId": "09363****9819",
    "emailAddress": "sjosh****@gmail.com"
  },
  "quotaBytesTotal": "16106127360",
  "quotaBytesUsed": "14243",
  "quotaBytesUsedAggregate": "21025545",
  "quotaBytesUsedInTrash": "0",
  "quotaType": "LIMITED",
  "quotaBytesByService": [
    {
      "serviceName": "DRIVE",
      "bytesUsed": "0"
    },
    {
      "serviceName": "PHOTOS",
      "bytesUsed": "0"
    }
  ],
}
```

```
{
  "serviceName": "GMAIL",
  "bytesUsed": "21011302"
}
],}
```

5.4.3 Summary

The experiments and analysis conducted in this section aim to address the key problems identified in **Section 3.3**. This is with reference to the recovery of artefacts that could be used in a live triage investigation, where communication content may not be immediately accessible. **Section 5.5** discusses how the Facebook experiments meet the objectives and addresses the research question [**Section 3.4**].

5.5 Relationship to Objectives and Research Question

The problem definition highlighted key areas that formed the research question [Sections 3.3; 3.4]. This section highlights how the artefacts recovered as a result of user activity on social media (Twitter, Facebook, and Google Drive) address the research question. The experimental results in this Chapter meet the following objectives:

- **[Objective 1] Characterise communications content:** the artefacts recovered highlight the user activities [Pages 111; 125 - 126; 132 - 133] initiated during the experiments, independent of the actual content being exchanged. They make a distinction between communication content and other artefacts, and demonstrate the possibility of recovering the content(s) of user communication.

Although the experiments were conducted with a limited dataset [Sections 3.5.1; 3.9.2], they set a baseline for the type of user activity that could occur on social media. The results demonstrate what constitutes communications content; and that other corroborating evidence may be used to infer and theorise a timeline of user activity.

- **[Objective 2] Categorise artefacts generated from the use of social media:** the artefacts recovered have been grouped into case-specific and context artefacts. They have been discussed in the context of their relevance, and how they can be used for relationship attribution. For example, a document was created during the experiment; the core browser artefacts show the document URL and name (title), and when the document was created; the cache shows information about when the document was cached.

5.6. RESULTS OVERVIEW

In the process of recovering and categorising these artefacts, it was necessary to identify attributes that could be used to infer user action. The experimental results highlight the complementary nature of social media artefacts, i.e. how they can be corroborated, or used to corroborate other pieces of digital evidence. The categorisation of social media artefacts was achieved with a subset of possible usage scenarios, however, the experimental methodology can be employed when investigating other potential user activities/behaviours that could occur on social media.

5.6 Results Overview

The recovery and analysis of the artefacts discussed in this Chapter highlight the artefacts that may be of interest in a live triage when investigating user social media activity; e.g. where they are located, and how they can be recovered. They highlight how the artefacts can be used for intelligence gathering, and to provide insight on known files or areas of interest for social media investigations. Table 5.6 illustrates a summary of the results, based on the various categories of activities conducted during the experiments.

Table 5.6: Results overview - user activities and artefacts (Chrome)

		Categories of activities									
Social Media Platforms		Registration and Authentication	Searching and Viewing Content	Sharing Content	Following	Friending	Create and Store Documents	Document Access and Update	Collaboration	Availability and Recoverability	Recovering Communication Content
	Twitter	✓	✓	✓	✓					✓	✓
	Facebook	✓	✓	×	×	✓				✓	×
	Google Drive	✓	✓	✓			✓	✓	✓	✓	✓

- The ticks indicate the recovery of artefacts applicable to a category.
- No ticks (grey boxes) indicate that some activities are platform specific e.g.
 - “friending” is specific to Facebook so was not applicable to Google Drive or Twitter experiments
 - document collaboration is a Google Drive feature and as such was not

applicable to Twitter and Facebook

- Cross boxes (x) indicate that no artefacts were recovered for that category from the experimental dataset e.g. communication content from Facebook activity was not recovered from the experimental dataset.

During the analysis, it was necessary to highlight certain caveats to be aware of when investigating social media activity. For example, the artefacts created as a result of unrelated processes, or web storage of content the user has not interacted with. The analysis was focused on artefacts that provide quick intelligence; are easy to retrieve and understand; and give an investigator a strategic advantage. The discussion highlights the need to contextualise user activity by corroborating each artefact of interest.

It was observed that the social media platforms behave differently with regard to the types of artefacts that could be recovered. This may be due to the limited experimental sample size, or due to the way each platform is designed to be used. However, the results highlighted the possibility of making inferences about user behaviour using a limited set of data, and also the possibility of determining user association(s).

5.7 Summary

This chapter discussed the ways in which social media platforms such as Twitter, Facebook and Google Drive can be used. It defined an experimental methodology that can be used to simulate user activity on different social media platforms. In the context of this research, this serves as a point of reference for mapping specific actions to the artefacts recovered during a social media related investigation; this enables the development of recovery and analysis techniques for specific artefacts types.

The chapter discussed the artefacts recovered from Google Chrome, created due to the subset of user activity performed on Twitter, Facebook and Google Drive. The results discussed were broadly grouped into browser history and cache artefacts. This was done to highlight where specific artefacts of interest are located. The results were discussed in the context of their relevance to a digital forensic investigation and were described as case-specific artefacts. This chapter demonstrates how relationships and actions may be attributed to a user, and highlights the importance of contextualising, and corroborating the different pieces of evidence recovered.

The framework conceptualisation is discussed in **Chapter 6**.

6 SMURF: Conceptualisation and Testing

6.1 Introduction

This chapter discusses the development of a Social Media User Relationship Framework (SMURF). In the context of this research, SMURF is a collection of components purposed for digital forensic live triage tasks involving social media activity. It is aimed at helping digital investigators and analysts build live triage tools quickly and effectively.

SMURF automates the recovery of artefacts that can be used for making inferences about user activity, and the attribution of actions to a user. Although created from a limited experimental dataset, as mentioned in **Chapters 4 and 5** the dataset is representative of a subset of defined categories of user social media activity.

In this chapter, the following font conventions are used:

Table 6.1: Font conventions

Item	Font typespace
Paths and Filenames	<code>monospaced typewriter</code>
Quotes and definitions	<i>italics</i>

The rest of this chapter is structured as follows: **Section 6.2** discusses the requirements for a framework. **Section 6.3** discusses the framework development process. **Section 6.4** presents an overview and description of the framework. The Proof of Concept (PoC) implementation is discussed in **Section 6.5**, and **Section 6.6** contains the chapter summary.

6.2 Framework Problem Definition

The experimental data analysed in **Chapters 4 and 5** was from a subset of user activities, used as a baseline for the ways in which social media may be used. The analysis of the results provided insight on the types of artefacts that may be recovered after social media usage, depending on the type of activity involved. These results were analysed in a post-mortem scenario where the virtual disk was in a powered

6.2. FRAMEWORK PROBLEM DEFINITION

off/inactive stage (i.e. simulating a physical disk image), and the artefacts recovered were used to make inferences about user activity, associations and relationships.

The analysis demonstrated that some artefacts may or may not be readily accessible in a live triage, depending on the context of user activity, as described in experimental methodology sections in **Chapters 4 and 5**. This impacts on the availability, and ability to recover artefacts such as the contents of user communication during a live triage investigation. In **Chapters 4 and 5**, pausing and making a copy of the virtual disk was devised as a means of comparing the artefacts recovered in a post-mortem scenario with what may be recoverable during a live triage investigation.

Digital forensics practitioners often come across time sensitive investigations; for example, cases involving missing persons, abductions, or where there is an imminent threat to life [**Section 2.6.3**]. This often means that the need for quick actionable intelligence takes priority over a detailed forensic analysis in a post-mortem environment; and the investigators would need to acquire the required data from a live environment. The prompt recovery of social media artefacts that can be used to make inferences about user behaviour, provides actionable intelligence at the early stages of an investigation, and can be used to scope/define an investigation.

In order to recover social media artefacts, a method of generalising potential usage scenarios is required. This will facilitate the identification and categorisation of the types of artefacts that may be recoverable; to achieve this, a generalised framework is required. To provide context to the need for a generalised framework, live triage was referenced multiple times during the course of this research, and was described as a *technique that can be used by investigators to retrieve quick actionable intelligence at a crime scene* [**Section 2.6.3**]. A generalised framework for this research would be designed to:

1. be used in a live triage investigation; providing multiple levels of functionality, and a reusable design that can be applied in any context.
2. facilitate the recovery, categorisation, and interpretation of social media artefacts, to enable the attribution of actions to a user.
3. provide investigators with information that would define the initial stages of the investigation, and generate new leads prior to a detailed forensic analysis of the target device(s).

6.3 Developing SMURF

6.3.1 Characteristics of the Framework

Whilst developing the framework, it was necessary to consider the characteristics that would define its functionality; this includes considering the availability of artefacts from user activity, as discussed in the framework problem definition [Section 6.2]. For example, during the analysis, an artefact inferring that a Tweet was sent was identified [Figure 4.7], and recovered as part of the session activity data; from **unallocated space** using the contents of the Tweet in a keyword search. In a real live scenario where the contents of user communication may be unknown, the recovery of this type of artefact may pose a challenge. This makes it necessary to devise a weighting system [Figure 6.1] that can be used to categorise the artefacts that may be recovered in a live triage investigation.

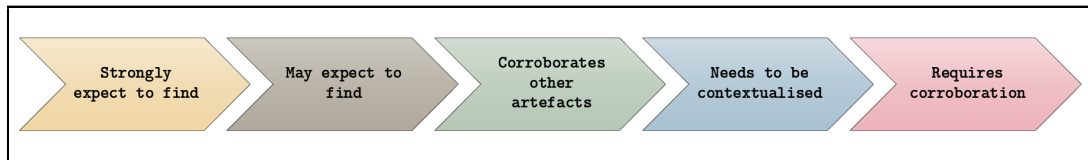


Figure 6.1: An illustration of the weighting scale

In the context of a live triage investigation, the following definitions of the weighting scale are proposed:

- **Strongly expect to find** refers to a range of artefacts that should generally be available to the investigator, for example, username, email address, user ID.
- **May expect to find** refers to a range of artefacts that may often be available but not always, for example, user login/logout activity, the profile or user IDs of other users.
- **Corroborates other artefacts** refers to artefacts that validate previously recovered artefacts, for example, search parameters in a URL.
- **Needs to be contextualised** refers to any artefact that could be interpreted to mean different things; and thus, needs to be put in context of the previously recovered artefacts to verify its relevance to the case, for example, the user's email address, email addresses of other users, file name or ID.
- **Requires corroboration** refers to any artefact that cannot be validated on its own, and therefore needs to be supported by existing artefacts. For example, communication content i.e. text extracts, images, video etc.

It is important to note that there may be overlaps when weighting the availability,

and relevance of artefacts recovered during a live triage investigation. For instance, an investigator **may expect** to find the username of another user, but would need to **corroborate** that information, and put it in **context** to determine if it occurred as a result of direct user interaction, or background processes on the social media platform.

In cases where there may be an imminent threat to life, the investigation is usually fast paced, and would more likely require live triage [Montasari, 2016]. As a result, some artefacts that would normally be recovered in a post-mortem digital forensic analysis, for example, data carving out of slack or unallocated space; analysing Volume Shadow Copies (VSC) [Section 3.10.1]; are out of scope for this research. This highlights the need for the formalisation of the evidence recovery process in a live triage, into a generalisable model that permits the identification of the communication, and type of activity that occurred on social media, as stated in the framework problem definition [Section 6.2].

Linking entities through the evidence recovered from social media investigations may be key to identifying co-conspirators in a criminal enterprise, as shown in UNITED STATES OF AMERICA v. ASHRAF AL SAFOO [Page 40]. In the context of this research, it is important that the framework developed can be used to recover artefacts that can be contextualised and corroborated, when determining a user’s social media associations and activities. The results analysed from the experimental users Josh, Obed, and Seth show their individual usage of social media; but in order to determine association or establish a relationship, their activities need to be reliably linked to each other [Arshad et al., 2019].

To show that communication occurred between two entities, it is necessary to show a relationship exists. For example, in Section 4.2.2, artefacts inferring a visit to another user’s profile were recovered [Fig. 4.19]. In order to make use of this information and to determine its value during a live triage investigation, it would need to be contextualised and corroborated by other related artefacts such as; profile names, profile IDs, profile searches, or data recovered from the cache. In addition, a relationship or association with the entity whose profile was viewed would need to be established through corroboration with other artefacts related to social media activity e.g. artefacts inferring consecutive visits and possible exchange of messages.

The recovery of data from live environments is usually common practice in incident response and malware investigations [Section 3.2.2]; however, it appears not to be

6.3. DEVELOPING SMURF

as common in digital forensic investigations. According to Casey [2013], “*there are immediate benefits to obtaining initial insights into computer contents*”, in order to determine whether the contents of the device has any probative value. Casey also, while addressing the risks associated with live triage in digital forensics, opines that when conducted in a forensically sound manner, a live triage showcases the “duality” of digital forensics as both a scientific discipline and an investigative tool. In this context, consideration was given to the need for an investigative tool that is generic enough to be used in any context and offers the flexibility for the development of forensic tools.

6.3.2 Purpose of Framework

In order to build the framework, it was necessary to address a number of concerns. This involved identifying existing approaches that can be used to solve the problem of social media artefact recovery, and user attribution in a live triage. It also involved identifying the benefits of these approaches, any risks involved in using them, and mitigations of these risks. Whilst there has been research in both the field of social media forensics and live triage, there has been no “marrying” of both subject areas to implement a technique for social media investigations in a live triage. This research identified the concerns from both subject areas, which were approached from ethical and legal perspectives, and are discussed in this section.

6.3.2.1 Individual Concerns

There were privacy concerns with regards to the personal data about other entities being analysed as part of a social media investigation. This was demonstrated and discussed in **Chapters 4 and 5**, where some of the artefacts recovered contained information about unrelated entities. It is important to note the risk that the personal data of entities not involved in the investigation may be captured during a live triage. Thus, the investigator should ensure that processes are put in place so that such data, where required may be excluded from the evidence submitted in court, and securely destroyed when the investigation is concluded.

6.3.2.2 Evidential Concerns

ACPO principle 2 states that “*In circumstances where a person finds it necessary to access original data, that person must be competent to do so and be able to give evidence explaining the relevance and the implications of their actions*”. According to Casey [2013], reasonable limits must be set when conducting triage on live systems as the investigator’s actions may alter the state of the digital data, thus compromising the evidence making it unreliable. This means that the investigator must have

the relevant forensic skills and awareness required in order to conduct a live triage; ensure that an audit trail is maintained to account for any potential changes, and to maintain evidential integrity.

Although live triage presents the opportunity of gaining access to, and acquiring data that may otherwise not be immediately available if a full forensic analysis was conducted [Casey et al., 2009; Montasari, 2016], it is important to note that the live triage process is not intended to be used as a substitute, or as a replacement for a full forensic investigation. It may however be used as a means of generating investigative leads, through the identification of data of probative value, which may lead to the discovery of inculpatory or exculpatory evidence.

6.3.2.3 Non-dependency of the Framework

The need for an investigative tool needs to be balanced with generalisability [Section 6.3.1]. This would enable the extension and maturity of the framework. The expectation for this framework is geared towards an approach that is not reliant on a particular operating system, social media platform, web browser, or on the use of a specific digital forensic tool set. The neutrality of the framework would enable an investigator to implement any suitable, forensically sound technique in a live triage investigation.

6.3.2.4 Defining Relationships

Using the data acquired during a live triage investigation, it is possible to quickly make inferences, and attribute actions or relationships to an entity. To establish that an offender is liable for a criminal offence, it is important to show beyond all reasonable doubt, through the evidence gathered, that the suspect's conduct was unlawful (*actus reus*) as a result of an act or the failure to act; it is equally important to demonstrate that there was an intent to commit the criminal act (*mens rea*) [Monaghan, 2016].

Live triage has been described as a way of previewing the contents of a digital device [Section 6.3.1]. When two or more entities are communicating on social media, it may be inferred that they know or are acquainted with each other. To establish *mens rea*, and demonstrate that there was an intent to commit a criminal act either on or through the use of social media, the artefacts recovered must be contextualised and corroborated. This involves validating different pieces of evidence against other related artefacts to contextualise potential user activity, and how an event occurred as demonstrated in **Chapters 4 and 5**. Attributing actions and relationships to users can also be used as a means of preventing future crimes from

6.3. DEVELOPING SMURF

occurring, through the dismantling of criminal enterprises [U.S.D.C., NORTHERN DISTRICT OF ILLINOIS, 2018]; and preventing further harm [Casey et al., 2009] or evidence tampering.

6.3.3 Framework Development

As discussed in **Section 6.3.1**, the framework needs to be able to recover social media artefacts during a live triage. Due to the limitations with the recoverability of social media artefacts for example, communication content may not be readily accessible; it was necessary to identify and define the artefacts that may be of interest and may be recovered in a triage investigation. These artefacts were identified during the manual analysis of the experimental data in **Chapters 4 and 5**.

The artefacts of interest were browser history data inferring; visited URLs, browser session activity, authentication credentials, cached data, information on created files, shared files and collaboration. In addition to these web browser related artefacts, system artefacts were identified [**Section 3.10.1**] as context and corroborating artefacts which support and provide a background to how social media activity may have occurred. The system artefacts of interest include Windows registry keys; these provide information on the browser installed on a target system, the default browser and frequency of use as this will likely be used for accessing social media.

Windows event logs and prefetch files were also considered of interest to social media investigations, as they support the evidence recovered from the registry, and corroborate activities that occur within a specified timeline. For example, providing support for browser session activity based on the system login and logoff times recorded in the event logs. Prefetch files also corroborate information recovered from the registry, and provide insight on any external devices and files accessed by an application, and the frequency of access.

Due to the fast paced nature of a live triage investigation, and the need for data that is not as extensive as the output of a detailed post-mortem analysis; it is necessary to reiterate that not all system artefacts were considered of interest to a live triage. However, other system artefacts may be used during a detailed analysis to further corroborate the output of the framework.

6.3.3.1 Framework Components

This stage of the development process involved identifying major components of the framework. This was an ongoing process during the manual recovery and analysis of the artefacts discussed in **Chapters 4 and 5**. As the artefacts were analysed,

a consideration was made as to whether or not they would provide immediate, easy to understand information during a live triage. This process involved the categorisation artefacts of interest in social media investigations into context or case-specific artefacts, i.e. core OS generated data or data created as a result of user browser activity. These were subsequently broken down into components for specific artefact types.

There was a need for a component that recovers the account information for the computer users; this includes but is not limited to how many user accounts are on the computer; user Security Identifier (SID) for the computer - this is a unique number used to identify a user account on Windows; access details i.e. logon/logoff, and currently logged in users. This was done to record whose user profile, in the event of multiple user accounts, the data was recovered from. This was followed by the need for a component that recovers and corroborates installed applications; for example, browsers. It identifies and returns what browsers are installed, when they were installed and frequency of use.

A component that would help an investigator quickly make a determination on how, when and what social media platforms have been used by the user was also required. This component would enable the recovery of the data of interest as identified in **Chapters 4 and 5**, and described in **Section 6.3.1**. This component also deconstructs the URL artefacts into individual attributes based on path segments (e.g. `/my_user_profile`) following the general URL syntax as specified in Berners-Lee et al. [1994]:

URL syntax:
`<scheme>:<scheme-specific-part>`

Following the above syntax, an example HTTP URL format is:

`http://<host>/<path>?<searchpart>`

Where `<host>` is the fully qualified domain name of a social media platform e.g. `twitter.com`, `facebook.com`, or `drive.google.com`. In the context of this research, the information after the protocol part of the URL i.e. from the host portion to the end of the URL is important, as it can be used to make inferences about user behaviour on social media. This includes establishing whether a profile was viewed, a resource was searched for etc. This syntax is a standard form of HTTP URL presentation and interpretation, and as such is not browser dependent.

6.3. DEVELOPING SMURF

In compliance with ACPO principle 2, it was determined that a component would be required to document the interactions between the deployment device and the target device. For example, capturing information about the device used in the deployment of the framework. A reporting component was also required as a means of presenting the output in an easy to assimilate format.

6.3.3.2 Development Environment: Set up

At the end of the theoretical stage of identifying what comprises the framework, it was necessary to define how it would be implemented. This resulted in the development and testing of a PoC. To do this, it was necessary to identify a programming language for the PoC. Python was chosen because it is widely used within the digital forensics community for the development of tools [Bryce and Miller, 2017; Hosmer, 2016; Miller and Bryce, 2019]. The simplicity of its syntax meant fewer lines of code could be used to achieve the intended results.

Selecting Python involved identifying and making a list of the libraries (and modules) that would be required for the framework components, and identifying ways of catching and handling exceptions for the different data types to be recovered. It also involved reviewing existing live triage techniques where open source tools have been employed. There are a variety of Graphical User Interface (GUI), and Command Line Interface (CLI) tools written for incident response and digital forensics investigations. As part of the tool review process, these tools were downloaded and tested on a Windows 10 physical machine. This led to the identification of some third party tools as useful in the recovery of artefacts in a live environment.

To aid the scripting process, it was necessary to select an Integrated Development Environment (IDE) and a version control tool. For this, PyCharm [JetBrains, n.d.] was installed, and the required plugins were enabled. Python modules were installed as required during the development process, and GitHub was used for version control via both the desktop and web interfaces.

6.3.4 Building the Framework PoC

Previous chapters highlighted that the focus of a live triage is to obtain intelligence from a crime scene in a timely manner, and that such information may be used by investigators in the assessment of the severity of a crime. It may also be used as a means of obtaining evidence (volatile and non-volatile) that can be used to support further forensic analysis. With this in mind, the data of interest were identified and discussed in **Chapters 4 and 5**. The components of SMURF as identified in **Section 6.3.3.1** are illustrated in Figures 6.2(a) and 6.2(b); showing how each

component is constructed, and the data flow into the core of the framework. This two-part diagram can be described as a “*practitioner-friendly*” graphical representation of SMURF (larger representations are also available in Appendix E). This section discusses how the script(s) for each component was written and the artefacts it is expected to recover. The testing was an ongoing process and was done as the code was being written for individual components. For the script extracts shown in the remainder of this section, several lines have been removed for brevity.

6.3.4.1 device_identifier

This component returns the identifying information for the device through which the framework (SMURF) is deployed. It searches for information about devices that have been or are currently plugged in to the computer and returns the Vendor ID (VID), Product ID (PID), device name, and timestamp. To achieve this, a script was written to parse the `setupapi.dev.log` file.

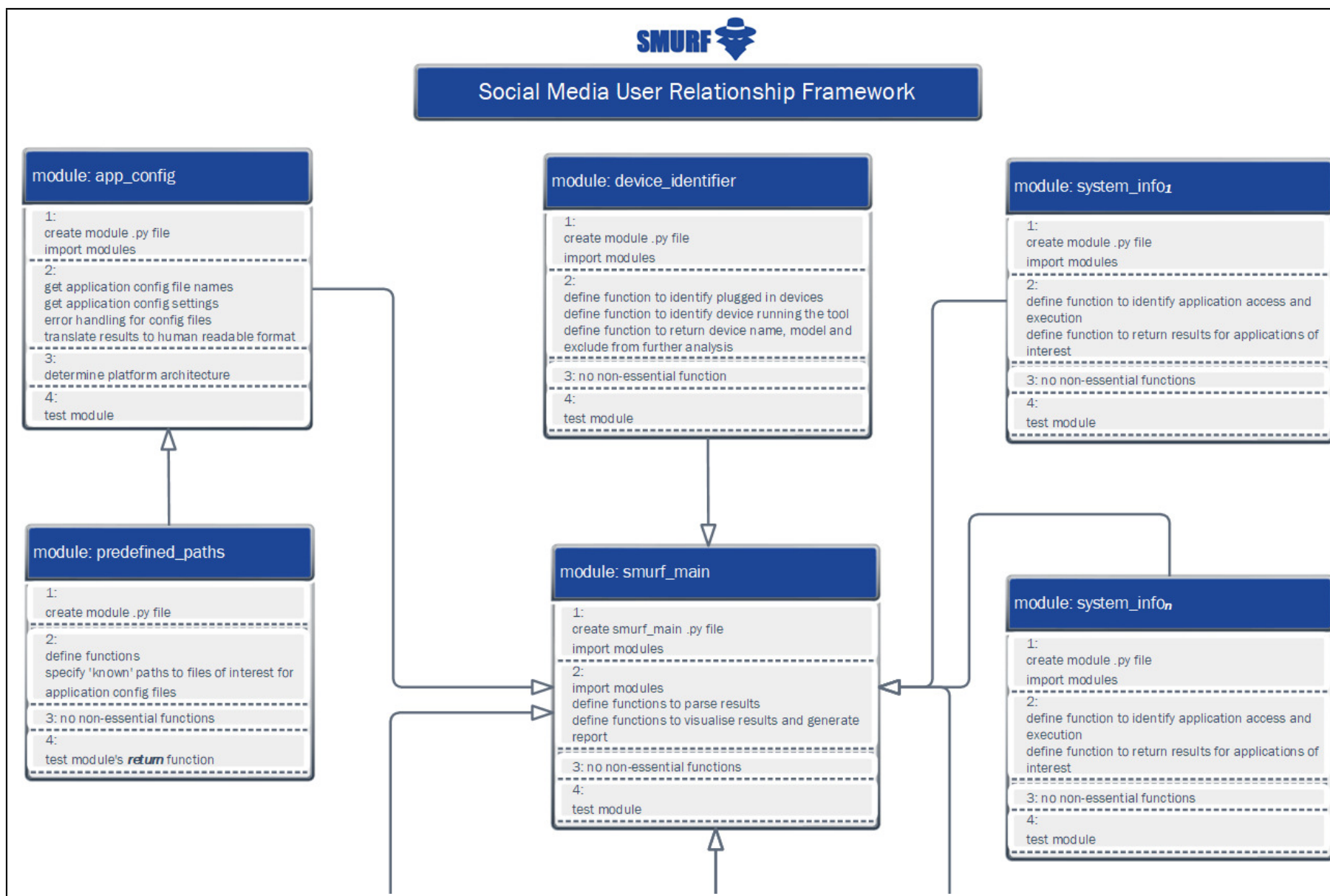
Extract from `device_id.py`:

```

1  # Path to log file = "%WINROOT%\Windows\INF"
2  win_dir = os.path.abspath(os.sep)
3  path_to_logfile = pathlib.Path(win_dir).joinpath('Windows', 'INF')
4
5  # Read data from setupapi.dev.log to get 'Device Install' Events
6  # for USB Devices
7
8  def get_devices():
9      # Check that the setupapi.dev.log file exists on this machine
10     if pathlib.Path(path_to_logfile).joinpath('setupapi.dev.log').is_file():
11
12         # Read lines in the setupapi file and search for device
13         # identifiers
14         try:
15             device_list = []
16
17             with open(os.path.join(path_to_logfile,
18                                   'setupapi.dev.log'), 'r') as log_file:
19
20                 # Get the OS info from the first 6 lines of the log file
21                 # (in Win 10).
22                 # This may change in the future - if it does, amend the
23                 # number of lines to read in.
24
25                 for i in range(6):
26                     os_info = str(log_file.__next__().strip(""))
27                     device_list.append(os_info)

```

This was devised as a method of reporting and logging the introduction of an external device into the target system during a live triage. The information recovered regarding the device running the tool is also printed in the final report.



6.2(a): A 'practitioner friendly' illustration of SMURF [Part 1]

6.3.4.2 predefined_paths

This component holds the predefined paths of interest from the registry. This serves as a global settings file for the registry parsers. It specifies the keys of interest for each of the registry hives; SAM, SECURITY, SYSTEM, SOFTWARE, NTUSER.dat, and UsrClass.dat.

The specified keys and their values contain information that can be useful in a live triage as it helps the investigator gain context about user activities for example, applications installed, recently accessed files, mounted devices etc.

Extract from predefined_hives.py:

```

1 def current_user_keys():
2     # keys of interest in the current user hive
3     # HKCU\Software\Microsoft\Windows\CurrentVersion
4     # HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer
5     # HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32
6     # HKCU\Software\Microsoft\Windows\Shell\BagMRU
7     # HKCU\Software\Clients\
8
9     comdlg_set = \
10         r"Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\
11         CIDSSizeMRU", \
12         r"Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\
13         OpenSavePidlMRU", \
14         r"Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\
15         LastVisitedPidlMRU", \
16         r"Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs", \
17         r"Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU", \
18         r"Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2", \
19         r"Software\Microsoft\Windows\CurrentVersion\Explorer\Taskband", \
20         r"Software\Microsoft\Windows\CurrentVersion\Run", \
21         r"Software\Microsoft\Windows\CurrentVersion\RunOnce", \
22         r"Software\Microsoft\Windows\Shell\Bags\1\Desktop", \
23         r"Software\Clients\StartMenuInternet", \
24         r"Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist",
25
26     return comdlg_set

```

6.3.4.3 app_config

This component is comprised of parsers for the registry hives; SAM, SECURITY, SYSTEM, SOFTWARE, NTUSER.dat, and UsrClass.dat. It was observed whilst testing the registry parsers that Windows places a lock on the SAM and SECURITY hives when the computer is powered on, making these hives inaccessible. To address

6.3. DEVELOPING SMURF

this, the Python subprocess module was used to dump the hives for subsequent parsing using the reg save command (i.e. `reg.exe save hklm\sam`). The HARDWARE hive was also dumped; this was done to enable the capture of system hardware information as the data in the HARDWARE hive is lost when the computer is powered off.

To access the registry hives, the winreg module was required. It provides access to the Windows registry API, and using the registry operations discussed in **Section 3.10.1** i.e. OpenKey, CloseKey, etc. In a live triage, besides the requirement for intelligence gathering, it is important to ensure that if alteration to the state of the device is unavoidable, it should be minimal and well documented. Based on this, it was necessary to ensure that the winreg Python module accessed the Windows registry in a read-only state.

The extract below is an example for the SOFTWARE hive. It shows read access to enumerate the subkeys in the predefined paths and return their respective subkeys and values. The key:value pairs are then written to a CSV file; this is to make it easier for filtering and manipulation of the data.

Extract from `parse_software.py`:

```
1  # predefined registry keys. These can be updated in 'predefined_hives.py'
2  prog_run = predefined_hives.software_keys()
3
4
5  def local_software():
6      for path in prog_run:
7          try:
8              i = 0
9              while True:
10                 get_keys = OpenKeyEx(HKEY_LOCAL_MACHINE, path, 0, KEY_READ
11                                     | reg_view_flag)
12                 name, value, data_type = EnumValue(get_keys, i)
13                 name = reg_get_name(name)
14                 data_type = reg_data_type(data_type)
15
16                 file_exists = pathlib.Path(sw_csv_file).is_file()
17
18                 with open(sw_csv_file, 'a+') as skv:
19                     fields = ["path", "name", "value", "data_type"]
20                     csvwriter = csv.DictWriter(skv, delimiter=',',
21                                                lineterminator='\n',
22                                                fieldnames=fields)
23
24
```

```

25         # create - if the file doesn't exist, write a header
26         if not file_exists:
27             csvwriter.writeheader()
28
29             skv.write("{0}, {1}, {2}, {3} ".format(path, name, value,
30                                                         data_type) + '\n')
31
32         i += 1

```

This component is designed to provide at quick glance, information regarding application configuration settings, recently accessed documents, and user specific settings that can be used to support other evidence found during the live triage. This information can also be used to identify areas to review in the event of a full digital forensic analysis. This module successfully returned the key:value pairs of interest with regards to system and application configuration in the registry.

6.3.4.4 system_info

This component is designed for system generated i.e. context artefacts. It covers the recovery of artefacts from Windows **Prefetch** and **Event Logs**. This provides information on access times, applications installed, the last time an application (e.g. a browser) was accessed.

As discussed in **Section 3.10.1**, these artefacts provide context to user activity. In a live triage investigation, these artefacts may provide information indicating the use of encryption in which case the “pull the plug” approach may not be a viable option. Information from context artefacts can be used to guide an investigation, and can be considered as part of the overall process of the digital forensics investigation. In the course of scripting this component, a number of third-party tools currently used for digital forensics and incident response were identified and tested. This was done to demonstrate that available CLI tools, or plugins where appropriate, may be used within SMURF to achieve the same results. The following tools were used in this component to recover data from **Prefetch** and the **EventLogs**:

- **PECmd** [Zimmerman, 2019], a CLI executable tool that processes prefetch files and reports evidence of execution. The output data can be exported to several formats (CSV, JSON, HTML) which can be specified at the command line.
- **python-evtx** [Ballenthin, 2019] is a Python parser for post-XP Windows Event Log files. The `evtx_dump.py` script in `python-evtx` was used to process the “Application, HardwareEvents, Security and System” event logs

6.3. DEVELOPING SMURF

and the output files were saved as an XML file to enable further parsing and filtering.

- SysInternals tools were used to recover additional system related information.

The subprocess module in Python was used to execute these third-party tools as shown in the following snippets(excess data removed for brevity).

Extract from `parse_prefetch.py`:

```
1 pref_out_file = "output_files/"
2
3 # set %winroot% as the OS partition may not be C: by default.
4 # join %winroot% to the prefetch path
5
6 win_root = os.environ['WINDIR']
7 prefetch_dir = os.path.join(win_root, 'Prefetch\\')
8
9
10 def parse_pf():
11     # if str(path).endswith(".pf"):
12     logging.info("Parsing {}".format(prefetch_dir))
13
14     subprocess.call([r"PECmd\PECmd.exe", "-d", prefetch_dir, "--json",
15                     pref_out_file], shell=True)
```

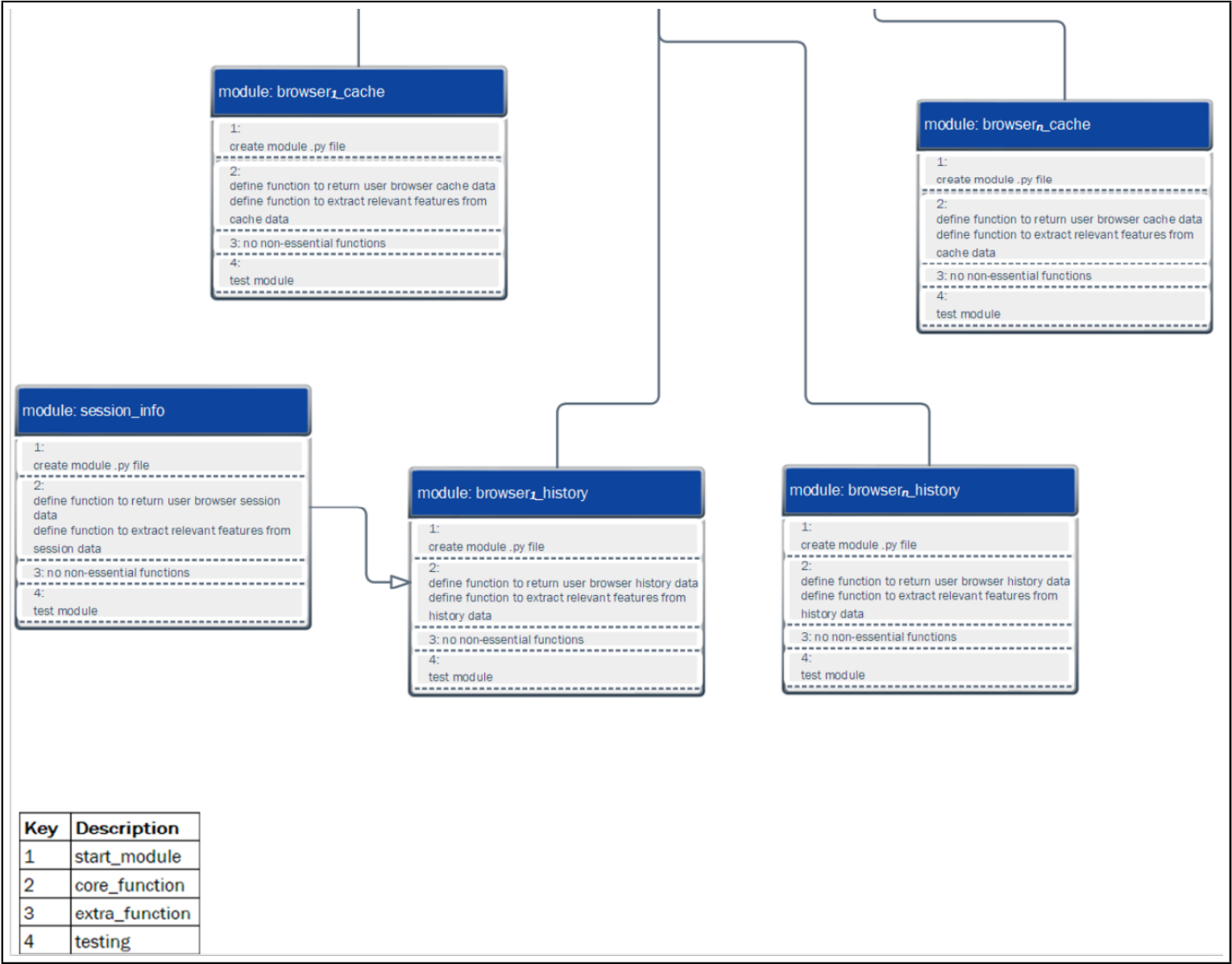
Extract from `parse_evtx.py`:

```
1 # set %winroot% as the OS partition may not be C: by default.
2 # join %winroot% to the evtx logs path
3
4 win_root = pathlib.Path(os.environ['WINDIR']).drive
5 evt_dir = pathlib.Path(win_root).joinpath('\\Windows', 'System32', 'winevt',
6                                           'Logs')
7
8
9 def parse_logs():
10     for each in evt_dir.iterdir():
11         i = str(each)
12
13         e_logs = ["Application.evtx", "HardwareEvents.evtx", "Security.evtx",
14                 "Software.evtx"]
15
16         for log in e_logs:
17             if i.endswith(log):
18                 evt_file = i
19
```

```

20         subprocess.run([r"python", r"third_party\bin\python_evtx\
21                         scripts\evtx_dump.py", evtx_file, ">>",
22                         evtx_out_file], shell=True)
23
24     subprocess.run([r"third_party\bin\sysinternals\psloglist64", "-accepteula",
25                     ">", r"output_files\loglist.txt"], shell=True)
26
27     subprocess.run([r"third_party\bin\sysinternals\pslist64", "-accepteula",
28                     ">", r"output_files\processlist.txt"], shell=True)
29
30     [# excess data removed for brevity]
31
32     # Parse XML
33
34     # Parse the python_evtx output file
35
36     tree = etr.iterparse(dest, remove_pis=True, huge_tree=True, \
37                         remove_blank_text=True, resolve_entities=True, \
38                         recover=True)
39
40     [...]
41
42     for event, elem in tree:
43         for child in elem:
44             results = child.tag, child.text
45
46             with open(evt_results, 'a', encoding="utf-8") as ft:
47                 # ft.write(child.tag, child.text)
48                 ft.write("{0}{1}{2}{3}{4}".format(child.tag, ': ',
49                                                    child.text, ' ', '\n'))

```



6.2(b): A ‘practitioner friendly’ illustration of SMURF [Part 2]

Figure 6.2: A two-part illustration of SMURF components and data flow

6.3.4.5 session_info

This component was based on the information session data files contain about user behaviour, as shown in **Chapters 4 and 5**. For the purposes of this research, data from the session information files from Firefox and Google Chrome was collected.

1. **sessionstore.jsonlz4** is used by Firefox to store session information [Section 4.2.2]. This file was converted to .json format using a Python script derived from Blumenbach [2015]’s mozlz4a python script.

```

1  # sessionstore
2  def sessionstore_info():
3      try:
4          import lz4.block as lz4
5      except ImportError:
6          import lz4
7
8      for each_dir in profile_dir.iterdir():
9          for i in each_dir.iterdir():
10             i = str(i)
11
12             if 'sessionstore' in i and 'lz4' in i:
13                 file_name = i
14
15                 get_data = open(file_name, "rb")
16                 read_bytes = get_data.read()
17                 get_sig = read_bytes[:8] # verify header
18                 if bytearray(get_sig).startswith(b"mozLz40"):
19                     json_data = json.loads(lz4.decompress(
20                         read_bytes[8:]).decode("utf-8"))
21
22                     with open(json_data_sstore, 'a+') as jd:
23                         printable = set(string.printable)
24                         jsd = filter(lambda x: x in printable, json_data)
25                         jd.write(str(json_data))
26
27                     for win in json_data.get("windows"):
28                         for tab in win.get("tabs"):
29                             i = int(tab.get("index")) - 1
30                             urls = tab.get("entries")[i].get("url")
31
32 [...]
33
34                     last_accessed = win.get("tabs")[i]
35                         .get("lastAccessed")
36                     tab_last_accessed = dt.datetime
37                         .utcfromtimestamp(last_accessed//1000.0)
38
39 [...]

```

2. **Current/Last Session; Current/Last Tabs files:** these files are used by Chrome to store information about the user’s window and tab activity

6.3. DEVELOPING SMURF

during a browsing session. The python subprocess module was used to run `chromagnonSession.py`; a part of the Chromagnon tool set which was used to parse the session data files from Chrome. The `Current Session` and `Current Tabs` files in Chrome are locked, and thus inaccessible when `chrome.exe` is running. It was necessary to programmatically terminate the process in order to recover the data in the files. The data recovered was saved in text file for further processing using RegEx to manipulate the data and for feature identification.

```
1  # The path to profile folder is '\\AppData\\Local\\Google\\Chrome\\
2  # User Data\\Default'
3  # pathlib.Path.home() joins the user's home directory path
4  # to the profile path
5  chrome_dir = \
6  pathlib.Path.home().joinpath('AppData', 'Local', 'Google', 'Chrome',
7                               'User Data', 'Default')
8
9  user_profile_dir = pathlib.Path(chrome_dir).glob('*')
10
11
12  def chrome_data():
13      # Check if chrome.exe is running and terminate process
14      # before parsing
15      # Chrome places a lock on the browser files when it is running
16      for proc in psutil.process_iter():
17          try:
18              # check if chrome.exe is running
19              if 'chrome.exe' in proc.name():
20                  proc.kill()
21          except (psutil.NoSuchProcess, psutil.AccessDenied,
22                  psutil.ZombieProcess):
23              continue
24
25      time.sleep(5)
26
27      # Parse current session
28      c_file = str(chrome_dir) + r"\Current Session"
29      logging.info("Parsing: {}".format(c_file))
30      subprocess.run(["python", r"chromagnonSession.py", c_file, ">",
31                     current_session], shell=True)
32
33      # Parse current tabs
34      ct_file = str(chrome_dir) + r"\Current Tabs"
35      logging.info("Parsing: {}".format(ct_file))
36      subprocess.run(["python", r"chromagnonTab.py", ct_file, ">",
37                     current_tabs], shell=True)
```

6.3.4.6 browser_history

The browser history component is designed to be extensible; As seen in Figure 6.2(b), there is room for the recovery of data from multiple browsers using SMURF. This also allows for session information from other browser to be fed into the framework to support browser history artefacts.

For the purposes of the PoC, the artefacts from Firefox and Google Chrome were analysed [Chapters 4 and 5] to help identify the types of artefacts that may be recoverable, and useful for attribution.

1. **Firefox:** The *sqlite* module in Python was used to query the database files. The recovered data was saved to CSV files for further processing. RegEx was used to filter the data of interest e.g. URL visits related to Twitter, Facebook, and Google Drive, as shown in the extracts below (excess data removed, and line breaks for brevity).

```

1  home_dir = \
2  pathlib.Path.home().joinpath('AppData', 'Roaming', 'Mozilla', 'Firefox',
3                                'Profiles')
4  profile_dir = pathlib.Path(home_dir)
5
6
7  def user_places():
8      for each_dir in profile_dir.iterdir():
9          for i in each_dir.iterdir():
10             i = str(i)
11             if 'sqlite' in i:
12                 file_name = i
13
14                 # establish a connection to the database
15                 conn = sq3.connect(file_name)
16                 cursor = conn.cursor()
17
18                 # query for places.sqlite
19                 sql = "SELECT datetime(visit_date/1000000, 'unixepoch')"
20                       "as visitDate, url, title "
21                       "FROM moz_places, moz_historyvisits "
22                       "WHERE moz_places.id = moz_historyvisits.place_id "
23                       "ORDER BY visit_date "
24
25  [...]
26
27  with open(places_csv) as rd:
28      readCSV = csv.reader(rd, delimiter=',')
29      for row in readCSV:
30          for item in row:
31              ml = ['twitter', 'facebook', 'drive']
32              for j in ml:
33                  if j in item:

```


6.3. DEVELOPING SMURF

```
34         patt = \
35             r'((?<=\/)[^:\/\s]*[?*\\w\\w]+)'
36         regexp = re.compile(\
37             patt, re.IGNORECASE)
38         matches = regexp.findall(item)
39
40         with open("parsed_url.txt", \
41                 'a+') as pu:
42             pu.write("{0}, {1}".format(\
43                 row[0], ', '.join(\
44                     matches)) + '\n')
```

2. **Chrome:** chromagnonHistory was used to parse the Chrome History file. This was done to demonstrate the extensibility and flexibility of the framework [Section 6.3.2.3].

```
1     # Parse history
2     h_file = str(chrome_dir) + r"\History"
3     logging.info("Parsing: {}".format(h_file))
4     subprocess.run(["python", r"chromagnonHistory.py", h_file, ">",
5                     chrome_hist_file], shell=True)
6
7     with open(chrome_hist_file, 'r') as ch:
8         for line in ch:
9             get_each_val = line.split(None, 3)
10            cut_date_val = [i.split('.', 1)[0]
11                            for i in get_each_val[:-2]]
12
13            ml = ['twitter', 'facebook', 'drive']
14            for j in ml:
15                if j in line:
16                    patt = r'((?<=\/)[^:\/\s]*[?*\\w\\w]+)'
17                    regexp = re.compile(patt, re.IGNORECASE)
18                    matches = regexp.findall(line)
19
20                    with open(chrome_parsed_url, 'a+') as pur:
21                        pur.write("{0}, {1}".format(\
22                            " ".join(cut_date_val),
23                            ', '.join(matches)) + '\n')
```

6.3.4.7 browser_cache

The browser cache component was designed to recover data that can be used to corroborate other browser activity data. As observed during the manual analysis of browser cache data, content may be cached independent of user interaction/activity. It is important to note that the volume of data returned by the cache can be overwhelming; thus this research has focused specifically on URL related cache information. This is because the recovery and verification of images, video or audio

content in the cache during a live triage, would require additional time to process such data, and thus defeats the purpose of conducting a live triage.

Cache data however, can be used during a detailed forensic analysis to support other evidence found on the device. The command line version of *Hindsight* by Obsidian Forensics [Benson, 2019] was used to parse the cache as shown in the extract below.

```

1  [...]
2  # Parse cache
3  chrome_cache_file = "output/" + \
4                      str(dt.datetime.now().strftime("%Y%m%d-%H%M%S")) + \
5                      '-chrome_cache_data'
6
7  chrome_dir = pathlib.Path.home().joinpath(\
8      'AppData', 'Local', 'Google', 'Chrome', 'User Data', 'Default')
9
10 user_profile_dir = pathlib.Path(chrome_dir).glob('*')
11
12
13 chrome_d = str(chrome_dir)
14 chrome_cache = str(chrome_dir) + r"\Cache"
15
16 subprocess.run(["python", r"hindsight\hindsight.py", "-i", chrome_d, "-o", \
17                chrome_cache_file, "-f", "jsonl"], shell=True)

```

6.3.5 Testing the core function of the framework

The main component of the framework (*smurf_main*) calls other components and manipulates the data recovered from them. It also generates a HTML report template that displays a “bitesize” version of the entire results.

Testing the core function of SMURF was done in stages as the each framework component was written. This was done to identify and resolve any dependency conflicts that may occur, and to ensure that the exceptions are handled correctly.

It is important to note that when the python modules required for the framework were identified, several components required the same modules (e.g. *pathlib*, *re*, *sys*, *csv*); thus, a global module/settings import file, *common_imports.py*, was used to provide standardisation across the components of the framework PoC.

6.3. DEVELOPING SMURF

The following is an extract from `smurf.py`:

```
1  # SMURF Main
2
3
4  def smurf():
5      # read content of output files from framework components
6      # write data of interest to SMURF Report.html
7
8      # Content to be published
9      title = "Triage Report"
10     devices = "SMURF deployed from: "
11
12     [...]
13
14     # Rename existing template and load the new report template
15     sys.stdout.write("Creating report template... \n")
16     suffix = "report/" + str(dt.datetime.now().strftime("%Y%m%d-%H%M%S")) \
17             + "-smurf_report.html"
18     html_template = "report/report_template.html"
19     if os.path.exists(html_template):
20         os.rename("report/smurf_report.html", suffix)
21         shutil.copy2(html_template, "report/smurf_report.html")
22
23     [...]
24
25     file_loader = FileSystemLoader("smurf_report")
26     env = Environment(loader=file_loader, autoescape=select_autoescape(
27         enabled_extensions=('html', 'xml', 'txt', 'csv', 'json', 'jsonl',),))
28     env.datetime_format = '%a, %d %b %Y %H:%M:%S'
29     template = env.get_template("smurf_report.html")
30     template.globals['now'] = dt.datetime.now
31
32     """
33     Render a template and write it to file.
34     """
35     sys.stdout.write("Generating report...")
36     with open(r"smurf_report/smurf_report.html", "w") as f:
37         f.write(template.render(start_time=start_time,
38                                title=title, devices=devices,
39                                tool_device=tool_device,
40
41     [...]
42
43     time.sleep(5)
44     sys.stdout.write('\n' "Exiting...")
```

6.3.6 Reporting the results

The reporting process involved the use of Jinja templating in Python to generate a HTML report with collapsible panels that enable the investigator to view sections of the framework output at a time.

According to The Pallets Projects [2019], Jinja is a “*modern and designer-friendly templating language for Python, modelled after Django’s templates. It is widely used, secure, and fast*”. It provides features such as sandboxed execution of the template, automatic HTML escaping system, and a configurable syntax. With Jinja templating, it was easy to generate a report by declaring variables and expressions in the template. These variables and expressions were substituted with the values from `smurf.py` when the report template was rendered.

Below is a minimal extract from the SMURF report template:

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4    <title>Triage Report</title>
5  </head>
6  <body>
7
8  <div class="container">
9    <h1><img src=smurfimg.png alt="SMURF icon" class=image width=120 height=50
10     /><strong>Live Triage Report</strong></h1>
11
12  [...]
13
14  <h4><strong>{{start_time}}</strong></h4>
15  <div class="panel-group" id="accordion">
16    <div class="panel panel-default">
17      <div class="panel-heading">
18        <h4 class="panel-title">
19          <a data-toggle="collapse" data-parent="#accordion"
20            href="#collapse1">Device and System</a>
21        </h4>
22      </div>
23      <div id="collapse1" class="panel-collapse collapse in">
24        <div class="panel-body">
25          <h5><strong>{{devices}}</strong></h5>
26          <p><span style="font-size: xx-small; "><ul>
27              {{tool_device}}
28            </ul></span></p>
29          <h5><strong>{{s_info}}</strong></h5>
30          <p><span style="font-size: xx-small; "><ul>
31              {% for n in user_os_info %}
32                <li>{{n}}</li>
33              {% endfor %}
34            </ul></span></p>
35        </div>
36      </div>
37  [...]
38
39  </div>
40 </div>
41

```

6.4. SMURF: HOW IT WORKS

```
42 [...]
43
44     <h4><strong> Finished: </strong></h4>
45 </div>
46
47 </body>
48 </html>
```

6.4 SMURF: How it works

SMURF was developed as a result of the need for a generalisable method for the recovery of social media artefacts during a live triage; with the aim of attributing actions and relationships to a user in the absence of communications content. There were several thought processes and activities involved in the planning and development of the framework. It is designed to be investigative tool with potential for extensibility and maturity.

The design process ensures that SMURF is **modular**, using a core component with several individual components. SMURF through its individual components, can be used to recover the predefined system artefacts, and the social media artefacts of interest from the browser, which can be used to relationship attribution. Its modular design enables the addition of new components; and the extension or exclusion of existing components from the core functionality of the framework, depending on the requirements of the end user community i.e. Law Enforcement/Investigators/Digital Forensics Practitioners/Researchers.

Figure 6.3 is an illustration of how SMURF operates; the components used to achieve each stage were discussed in **Section 6.3.4**.

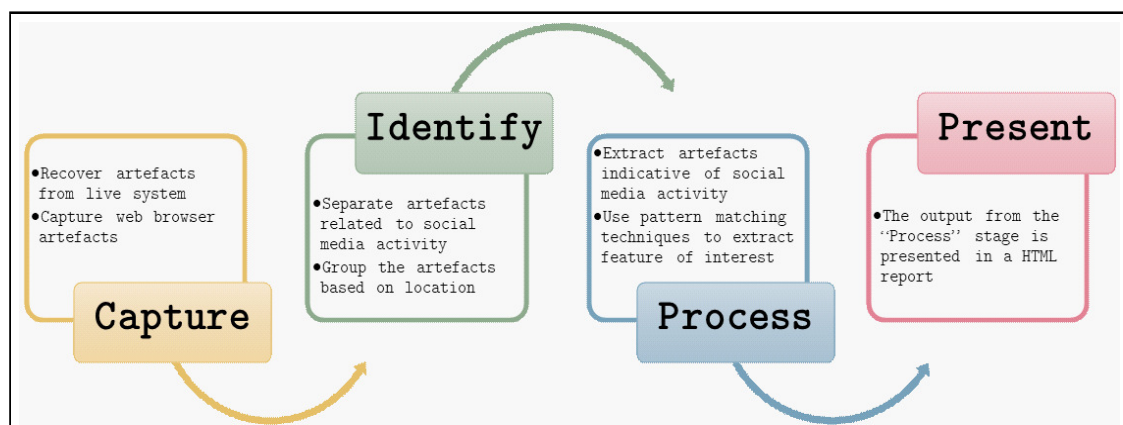


Figure 6.3: An illustration of how SMURF works

- **Capture:**

This stage involves recovering data from a live environment; and initiates interaction between the live triage device and the target device. It was deemed necessary to programmatically record the triage device details, including the date and time of connection to the target device, before proceeding with the data collection. The data collection at this stage focused on system generated and user generated artefacts (e.g. artefacts from social media activity).

- **Identify:**

This stage involves segregating the data captured at the preceding stage. The artefacts are grouped based on type (e.g. URL, JSON) and the location (cache, history). For example, if the data contains the social media platform name or an associated name, and it is a URL (**type**), put it in the “visited places” group for further processing.

- **Process:**

The artefacts recovered and grouped at the “Identify” stage are processed to determine the type of activity that could have created them (e.g. example, user account registration, viewing and sharing content etc). Pattern matching techniques are used to extract features of interest. These artefacts are parsed, and exported to text or CSV files to facilitate the reporting stage.

- **Present:**

The data from preceding stages feed into the “Present” stage. The output (CSV, text) files from the “Process” stage are parsed, and returned in a text based HTML report as shown in the report template extract. Following the theme of a live triage, this report provides the investigator with a quick view of the data recovered, in a usable manner. The present stage involves presenting the parsed data from SMURF in a HTML report that easy to view and digest.

SMURF is designed to be **maintainable** through the adaptability of its components, in the event of a change to an operating system, web browser or social media platform functionality. The intention is for it to be **generic** and function dependent, not artefact dependent; enabling further development of the framework, to broaden its implementation across a wider range of investigations. Being a triage tool, its **ease of use** was an important factor thus, the decision was made to design a CLI PoC, which can be executed by typing in the program name in the OS command line prompt.

The nature of live triage investigations makes potential alteration to a digital device unavoidable. This is because oftentimes, the live triage tool is deployed via a

6.5. SMURF POC IMPLEMENTATION

portable device and interacts with a live system. To enable SMURF meet the forensically sound requirements, it has been designed so that artefact recovery is mostly achieved through read-only processes with logging enabled to record potential interactions.

6.5 SMURF PoC Implementation

6.5.1 Justification for Implementation

It has been discussed in previous chapters that conventional digital forensic tools are usually not suitable for a live triage investigation, as they are time and resource intensive when it comes to the processing of artefacts. Roussev and Quates [2012] suggests that triage tools need to avoid deep examination such as file carving, and indexing; in favour of alternative approaches such as focusing on lightweight methods that do not impact on performance. Bearing this in mind, this chapter looks to demonstrate the practicality of high level approaches to the recovery of digital evidence during a live triage.

At the end of the theoretical stage of identifying what comprises the framework, it was necessary to define how it would be tested and applied in a real world scenario. This resulted in the development of a PoC as a means of demonstrating the viability of a framework for live triage investigations; tasked with the automated recovery of social media artefacts that can be used to determine user associations and activities.

To test the applicability of the framework in a real world scenario, it was important to understand and define categories of artefacts the framework is expected to recover, based on the user activities defined in **Chapters 4 and 5**. The results from **Chapters 4 and 5** have been used to establish that it is possible to recover artefacts from social media activity. This was necessary in order to have a basis for comparison between the artefacts recovered by the framework and artefacts recovered manually.

The PoC has been written in Python specifically for this research, with the functionality of each framework component defined in **Section 6.3.4**. However, it is important to state that the purpose of developing the PoC is to show that the framework is generalisable; produces the expected results, and can be implemented in real world scenarios. Thus, there is no formal requirement in this research for benchmarks, performance checks, or code optimisation because the focus is on the viability of the framework.

As part of the testing process for the PoC, it was necessary to determine an efficient

way to apply the framework in a live environment using real world data. This involved identifying a suitable method for deploying the framework in a live environment with a larger dataset. The chosen method of application is discussed further in **Section 6.5.2**.

6.5.2 Method of deployment

Several options were considered with regards to the deployment of the PoC; for example, `PyInstaller`, `cx_Freeze`, `py2exe`. Being a collection of components with third party tools/utilities, the option of compiling into one executable was not a viable option.

The utilities reviewed during this research can be used to bundle a python script and all its dependencies into an executable file; either as a one file bundled executable, or a folder bundle with the executable and other required files. `PyInstaller`, `cx_Freeze`, and `py2exe` were installed on the development machine and tested. They were not considered a practical solution at this time. This is because the use of third party tools and scripts within the framework was aimed at demonstrating the flexibility, and extensibility of the framework; thus compiling into a single tool was deemed impractical.

In addition, it was important to ensure:

- good ethical conduct, and compliance with the legal use of the third party tools (i.e. adherence to potential licence ramifications or legal implications);
- respect for the authors/developers, and their licences by using the tool in the way author consent implies.

WinPython is a free and portable Python-distribution for Windows, and it does not require any OS installation; it can be customised to add additional packages, and can be redistributed as a zip file [WinPython, 2019]. The WinPython zip file was downloaded, extracted to a USB and tested on a Windows 10 computer. WinPython was subsequently chosen as a method of testing the PoC on a live system.

6.5.3 Framework Testing

Testing the framework through the PoC presents a means to identify, understand, and document potential changes that could occur as a result of the actions an investigator may take during a live triage. Through the framework, it is possible to identify the components that provide useful information quickly, with no extensive forensic analysis required; and also ascertain if the artefacts of interest are where they are expected to be as indicated by the results analysed in **Chapters 4 and 5**.

6.5. SMURF POC IMPLEMENTATION

Although the research started with Windows 7, where the experimental dataset was generated, this operating system is now out of support. The limited dataset generated during the experiments made it necessary to identify a suitable, larger dataset for testing; the framework testing was performed on a Windows 10 computer with real life data generated by the researcher. Testing the framework with real world data slightly increases the available dataset, and presents a means of testing the recoverability of social media artefacts from a dataset comprised of a range of user Internet activity. It strengthens the research through the introduction of a variety of artefacts that can be used to test the framework’s ability to facilitate the identification and extraction of features that are specific to social media activity.

This dataset can be used to ascertain the reliability of the results recovered by the framework, when compared with the data recovered through manual identification. It also simulates real life implementation as the dataset has been generated over a number of years; with varying activities that were not extensively covered during the experiments. The use of real life data limits the ability to publish or share the data due to the personal information it contains.

Framing the PoC testing involved defining the categories of artefacts expected from the user activities defined in **Chapters 4 and 5**, it also includes the use of a weighting system [**Section 6.3.1; Figure 6.1**] to ascertain the availability and recoverability of different artefact types. The framework components are executed consecutively; the artefacts of interest are extracted, and the output is saved to text or CSV file for further testing. Then the final results were presented in the HTML report generated by `smurf_main.py`.

The expected artefacts and their individual weighting are outlined in the remainder of this section, with a results summary presented at the end of this section. The following symbols are used as identifiers for the results in Tables 6.2 - 6.8.

- Ticks (✓) indicate the recovery of artefacts applicable to a weighting category.
- Grey boxes (■) indicate that some activities are not applicable to a category for example,
 - artefacts that lean towards ‘**strongly expect to find**’ would not fall under ‘**may expect to find**’ at the same time.
- Lozenges (◆) indicate categories/artefacts that would be nice to have but are not a priority.
- Club (♣) indicates that no artefacts (e.g. communication content) were recovered for that activity/category from the experimental dataset.

6.5.3.1 Registration/Authentication

As discussed in **Section 4.2.1**, it is possible for a user to access some social media platforms; search and view content, but would not be able to interact with other users. Table 6.2 shows the user artefacts that can be recovered as a result of account registration or authentication activities initiated on social media.

Table 6.2: Group of artefacts related to user registration or authentication

Registration/Authentication		Weighting				
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised	Requires corroboration
User artefact	Email address	✓	◆	◆	✓	✓
	Email addresses of other users				✓	
	Username	✓	◆	◆	◆	◆
	User ID	◆	✓	◆	◆	◆
	Username of other users		✓	◆	◆	
	User ID other users		✓		◆	
	Profile name	✓	◆	◆	◆	◆
	Profile ID	◆	✓	◆	◆	✓
	Profile name of other users		✓		✓	
	Profile ID of other users		✓		✓	
	File name		✓		◆	✓
	File ID		✓		◆	✓
	Create		◆		◆	◆
	Edit		◆		◆	◆
	Search parameters	◆	✓	◆	◆	◆
	Query parameters	◆	✓	◆	◆	◆
	Login	✓	◆		✓	◆
	Logout	✓	◆		✓	◆
	Communication content	♣	✓		◆	✓

The results indicate that there is a strong possibility of recovering artefacts such as the email address that was used for account registration and subsequently for authentication. In addition to obtaining the email addresses or username(s), URLs indicating login/logout activity can also be recovered, and used to infer user account authentication activity.

6.5. SMURF POC IMPLEMENTATION

There are other artefacts that may be used to support account activity. These include finding the usernames, profile names, and user/profile IDs for other users. This may occur during the registration process where Twitter suggests people the user may like to follow. The artefacts shown are as identified during the manual analysis [Chapters 4 and 5]. The weighting matrix has been used to group the artefacts as recovered from all three social media platforms (Twitter, Facebook, and Google Drive) during the PoC testing.

In the context of user activity, ‘**strongly expect to find**’ provides stronger evidence to validate the assumption about possible account ownership. The user artefacts in this category were recovered using the `session_info`, `browser_history` and `browser_cache` framework components. The output was saved to individual CSV and text files; and parsed by the framework core component `smurf_main.py`. The artefacts weighted as **strongly expect to find** were recovered. Followed by search and query parameters in the **may expect to find** weighting category.

Additional artefacts recovered in relation to user authentication activity were the usernames and profile IDs of other users whose profiles had been viewed prior to the the login activity.

6.5.3.2 Search and View Content

As mentioned in Chapters 4 and 5, it is possible for users who are not logged in or registered to search for and view content. Table 6.3 shows the user artefacts that could be used to infer ‘search and view’ activities. This applies to both registered and unregistered users; the main difference being the availability of artefacts indicating that the user has an account on the social media platform. These activities were searched for using the framework to determine the reliability of the weightings applied to the artefacts recovered during the manual analysis of the experimental results.

Using the weighting matrix, the PoC was used to recover artefacts such as user/profile names and IDs in the ‘**strongly expect to find**’ category. Other artefacts that can be used to build context were in the ‘**may expect to find**’ category. For example, where another user profile has been searched for by email address; the search parameters (**may expect to find**) can be used to contextualise the profile ID recovered (**strongly expect to find**), and thus the profile visit/view.

6.5. SMURF POC IMPLEMENTATION

Table 6.3: Group of artefacts related to user search and view activity

Search and View content					
		Weighting			
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised
User artefact	Email address	✓	✓	✓	✓
	Email addresses of other users	✓	✓	✓	✓
	Username	✓	✓	✓	✓
	User ID	✓	✓	✓	✓
	Username of other users	✓	✓	✓	✓
	User ID other users	✓	✓	✓	✓
	Profile name	✓	✓	✓	✓
	Profile ID	✓	✓	✓	✓
	Profile name of other users	✓	✓	✓	✓
	Profile ID of other users	✓	✓	✓	✓
	File name	✓	✓	✓	✓
	File ID	✓	✓	✓	✓
	Create	✓	✓	✓	✓
	Edit	✓	✓	✓	✓
	Search parameters	✓	✓	✓	✓
	Query parameters	✓	✓	✓	✓
	Login	✓	✓	✓	✓
	Logout	✓	✓	✓	✓
	Communication content	✓	✓	✓	✓

The user artefacts in this category were recovered using the `session_info`, `browser_history` and `browser_cache` framework components. RegEx was used to parse the output files to extract artefacts inferring a range of search and view activities. The artefacts weighted as **strongly expect to find** were recovered as expected. In the **may expect to find** category, search and query parameters, usernames, profile IDs, and file IDs were recovered; showing that the data recovered during the manual analysis can be recovered by the PoC implementation.

6.5. SMURF POC IMPLEMENTATION

6.5.3.3 Share Content

Communication content in the context of this research can be described as *messages exchanged between two entities, which may contain audio, video, image files or text*. As demonstrated by the manual analysis, recovering the content of communication may be obtainable in some scenarios.

Table 6.4: Group of artefacts related to user content sharing

Share content		Weighting				
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised	Requires corroboration
User artefact	Email address		✓	◆	◆	
	Email addresses of other users		✓		✓	✓
	Username	✓			✓	✓
	User ID	✓			✓	✓
	Username of other users	✓			✓	✓
	User ID other users	✓			✓	✓
	Profile name	✓			✓	✓
	Profile ID	✓			✓	✓
	Profile name of other users	✓			✓	✓
	Profile ID of other users	✓			✓	✓
	File name		✓	✓	✓	✓
	File ID		✓	✓	✓	✓
	Create	◆			◆	◆
	Edit		✓		✓	✓
	Search parameters		✓	✓	✓	◆
	Query parameters		✓	✓	✓	◆
	Login	✓		✓		◆
	Logout	✓		✓		◆
	Communication content	◆	✓	✓	✓	✓

Table 6.4 shows the user artefacts that can be recovered content being shared on social media. Evidence of these activities was searched for using the framework to determine the reliability of the weightings applied to the artefacts recovered during the manual analysis of the experimental results.

The user artefacts in this category were recovered using the `session_info`, `browser_history` and `browser_cache` framework components. The artefacts of interest which infer that content was shared on social media, consists of features which may be used to identify users for example, user/profile names and IDs, URLs parameters such as file names etc. These can be used in conjunction with page titles and login/logout URLs to infer that a user was logged in, and possibly shared the file. The weighting of these artefacts shows that contextualisation is required in order to link content to an entity.

The artefacts weighted as **strongly expect to find** were recovered as expected. In the **may expect to find** weighting category, search and query parameters were recovered. In this instance, communication content was not recovered; as the goal of a live triage is the prompt provision of reliable intelligence, for the intended audience (e.g. investigators), in an exploitable format. It is important to note that non-registered users cannot share but are able to view publicly available content so **context** is necessary in order to draw inferences.

6.5.3.4 Association/Relationship

Table 6.5 shows the artefacts that can be used to infer a user's association on social media, and by extension, relationships. When trying to ascertain relationships between entities on social media, it is important to contextualise the evidence inferring contact between two entities.

As discussed in **Chapters 4 and 5**, it is possible for associations to be formed without a direct relationship. For example, when an entity is connected with a brand name social media profile.

The user artefacts in this category were recovered using the browser framework components. The artefacts of interest consists of features which may be used to identify users e.g. usernames, profile IDs etc. When analysed in context, the activities surrounding the creation of these artefacts can be used to ascertain the connection between two or more entities.

6.5. SMURF POC IMPLEMENTATION

Table 6.5: Group of artefacts inferring user associations/relationships

Association/Relationship		Weighting				
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised	Requires corroboration
User artefact	Email address	✓		◆		
	Email addresses of other users		✓		✓	✓
	Username	✓		◆	◆	
	User ID	✓		◆	◆	
	Username of other users	✓		✓	✓	✓
	User ID other users	✓		✓	✓	✓
	Profile name	✓		✓	✓	✓
	Profile ID	✓		✓	✓	✓
	Profile name of other users	✓		✓	✓	✓
	Profile ID of other users	✓		✓	✓	✓
	File name		✓	✓	✓	✓
	File ID		✓	✓	✓	✓
	Create		♣		◆	◆
	Edit	♣	◆	◆	◆	✓
	Search parameters	◆	✓	✓	✓	◆
	Query parameters	◆	✓	✓	✓	◆
	Login		✓	◆		
	Logout		✓	◆		
	Communication content	♣	✓	✓	✓	✓

The artefacts weighted as **strongly expect to find** were recovered as expected. In the **may expect to find** weighting category, search and query parameters were recovered, indicating that particular user profiles were searched for, then added as social connections. In this instance, the framework did not recover communication content related to creating social relationships.

6.5.3.5 Document Creation

The results from the manual analysis show that URL path segments may contain properties that can be used to ascertain whether a user created a document.

6.5. SMURF POC IMPLEMENTATION

Table 6.6: Group of artefacts related to user document creation

Document creation					
		Weighting			
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised
User artefact	Email address	✓	■	◆	✓
	Email addresses of other users	■	✓	◆	◆
	Username	✓	■	◆	◆
	User ID	✓	■	◆	◆
	Username of other users	■	✓	■	◆
	User ID other users	■	◆	■	◆
	Profile name	✓	■	■	◆
	Profile ID	✓	✓	■	✓
	Profile name of other users	♣	♣	■	■
	Profile ID of other users	■	✓	■	◆
	File name	✓	■	◆	◆
	File ID	✓	■	◆	◆
	Create	✓	■	◆	◆
	Edit	✓	■	◆	◆
	Search parameters	◆	✓	✓	◆
	Query parameters	◆	✓	✓	◆
	Login	■	✓	■	■
	Logout	■	✓	■	■
	Communication content	◆	✓	✓	✓

The file names/IDs recovered may be used to identify a file, or files of interest. Table 6.6 shows some of the artefacts of interest and their weighting categories.

6.5.3.6 Document Update

Artefacts created as a result of a document update could be useful during an investigation. When a document is updated by a user, it is possible that fragments of the document may be written to disk. In a live triage, due to the requirement for quick, exploitable information; artefacts indicating edits/updates can be useful in determining the artefacts that may require further analysis; for example, unlawful images, or documents used in the dissemination of unlawful content.

6.5. SMURF POC IMPLEMENTATION

Table 6.7: Group of artefacts related to user document update

Document update		Weighting				
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised	Requires corroboration
User artefact	Email address		✓	◆		
	Email addresses of other users		✓		✓	✓
	Username	✓		◆	✓	✓
	User ID	✓		◆	✓	✓
	Username of other users		✓	◆	✓	✓
	User ID other users		✓	◆	✓	✓
	Profile name	✓			✓	✓
	Profile ID	✓			✓	✓
	Profile name of other users		✓		✓	✓
	Profile ID of other users		✓		✓	✓
	File name	✓		✓	✓	✓
	File ID	✓		✓	✓	✓
	Create	◆		◆	◆	
	Edit	✓		◆	✓	✓
	Search parameters		✓	✓	✓	
	Query parameters		✓	✓	✓	
	Login	✓			◆	
	Logout	✓			◆	
	Communication content	♣	✓	✓	✓	✓

The artefacts recovered during the framework testing corroborates the artefact types recovered in **Chapters 4 and 5**. It is important to note that registry artefacts may be used to contextualise, and corroborate updated/edited documents; these were recovered using the `app_config` component, and provide information on recently accessed files, which may be used to corroborate the file names recovered. Table 6.7 illustrates the weighted artefact categories that can be recovered during a live triage.

6.5.3.7 Document Collaboration

Artefacts created as a result of a document collaboration can be useful during an investigation; for example, where the entities involved are members of a criminal enterprise. Artefacts in this context may be recovered from the browser or system

6.5. SMURF POC IMPLEMENTATION

activity; these artefacts can be useful in determining whether further investigation is warranted.

It is also a means of establishing that a relationship exists between the individuals, and the documents may require further analysis; for example, documents used in the dissemination of unlawful content. Table 6.8 shows the weighted artefact categories that can be recovered during a live triage.

Table 6.8: Group of artefacts related to user document collaboration

Document collaboration					
		Weighting			
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised
User artefact	Email address	✓	✓	✓	✓
	Email addresses of other users	✓	✓	✓	✓
	Username	✓	✓	✓	✓
	User ID	✓	✓	✓	✓
	Username of other users	✓	✓	✓	✓
	User ID other users	✓	✓	✓	✓
	Profile name	✓	✓	✓	✓
	Profile ID	✓	✓	✓	✓
	Profile name of other users	✓	✓	✓	✓
	Profile ID of other users	✓	✓	✓	✓
	File name	✓	✓	✓	✓
	File ID	✓	✓	✓	✓
	Create	✓	✓	✓	✓
	Edit	✓	✓	✓	✓
	Search parameters	✓	✓	✓	✓
	Query parameters	✓	✓	✓	✓
	Login	✓	✓	✓	✓
	Logout	✓	✓	✓	✓
	Communication content	✓	✓	✓	✓

The artefacts recovered through the framework corroborates the artefact types recovered in **Chapters 4 and 5**. It is important to note that the registry may also provide artefacts that can be used to contextualise and corroborate updated/edited documents. The data recovered can be used to identify a user; infer user activity, and attribute relationships, i.e. a connection between two entities such as a profile

6.6. SUMMARY

page visit, or between an entity and an action such as creating or accessing a shared document.

When tested with real live data, it was observed that historical data which had accumulated over the years was recovered, this includes usernames and passwords; hostnames where the credentials were used; file and folder IDs etc. The PoC has demonstrated that the framework can be used in a live triage for the recovery of social media artefacts. When applied to real life data, which can be said to be more representative of user activity, it provides additional information that can be used to corroborate the previously recovered evidence. Other artefacts of interest recovered include; browser data synced across devices by the researcher. In a live triage investigation, this data may be useful in understanding how, and what external devices are being used.

The artefacts recovered are also subject to the same caveats applied during the manual analysis [Chapters 4 and 5]. It is important to contextualise digital evidence in order to exclude data that is not of interest to an investigation. Examples include some data from the browser cache, cookies etc., which are not always representative of user activity. The framework at this stage requires a review of the artefacts by an investigator, in order to verify and contextualise user activity. However, it is important to note that this framework leaves room for further development.

6.5.3.8 Generalisability of SMURF

The framework was created using a limited dataset which was not fully representative of real world usage of social media. The data collected was from a subset of user social media activity and thus, may not present an accurate representation of the range of artefacts that may be recoverable. A larger dataset was required to test the generalisability of the framework so, a Windows 10 computer with real world data was used. The artefacts recovered from this dataset can be said to be representative of real life user activity to a degree, and may be recovered in a live triage.

6.6 Summary

This chapter highlights that the focus of a live triage is on the ability of an investigator to obtain actionable intelligence, during time critical investigations, through the recovery of meaningful evidence. It provides insight on the importance of contextualisation of digital evidence, in order to establish user intent when investigating a crime committed with, or through social media.

It highlights the concerns raised when conducting a live triage investigation, and provides a rationale for the need for a viable framework (SMURF). It discusses the build process for the framework, which includes identifying artefacts of interest and the individual components for the framework testing. It discussed building a PoC which is intended to demonstrate the generalisability of the framework; justifying the selection of Python as the language of choice due to its widespread use within the digital forensics community.

The next chapter discusses the research approach, and evaluates the research conducted to develop a framework for the investigation of social media activity.

7 Discussion

7.1 Introduction

The aim of this research was to evaluate whether artefacts other than the content of user communication on social media can be used for the attribution of actions or relationships to a user. This research was also focused on the development of a framework that can be used for the recovery of these social media artefacts in a live triage investigation. This is based on the premise that the content of user communication may not be immediately accessible, and there is a need for quick actionable intelligence at the crime scene.

The remainder of this chapter is structured as follows: **Section 7.2** discusses the approach taken to address the key research problems. **Section 7.3** evaluates the research objectives, which are broken down into individual sections; **Section 7.3.1** evaluates the methods employed in the characterisation of communication content. **Section 7.3.2** evaluates the approach taken to group the artefacts created from the use of social media. The method used in the development of the PoC framework is evaluated in **Section 7.3.3**. **Section 7.3.4** evaluates the methods used for assessing the reliability of social media artefacts, and **Section 7.3.5** evaluates the methods used to determine the relationship between communicating entities. In **Section 7.4**, the research question is evaluated, followed by the chapter conclusion in **Section 7.5**.

7.2 Addressing key research problems

In order to determine the feasibility of a framework for user attribution in a live triage, it was necessary to understand the triage process and the existing standards. The goal was to understand how such a framework would work, given the delicate nature of live investigations; and trying to strike a balance between the need for the recovery of artefacts, user privacy, and the requirement not to compromise a target device [ACPO, 2012]. Manual analysis of social media artefacts was used to establish a baseline of the type of artefacts that may be recoverable.

To understand the research, it was necessary to deconstruct the research question, and map its keywords to the research objectives. This was done to show how these keywords relate to, and help address the key research problems identified in

Section 3.2; these collectively reinforce the research aim, and an illustration can be seen in Table 7.1.

Table 7.1: Relationship between the research objectives, research question, and key research problems addressed

Research objectives	Relationship to research question	Key problems (KPs)
Characterise communications content	In the absence of the contents of user communication	KPs 1 and 5
Categorise artefacts generated from the use of social media	can social media artefacts be used to	KPs 2 and 4
Develop a framework for the recovery and analysis of social media artefacts		KPs 2, 3, 4, and 5
Assess the reliability of social media artefacts	reliably attribute actions to individual users?	KPs 4 and 6
Determine the relationships between communicating entities	Can they be used to link other entities involved in the communication?	KPs 4 and 5

7.3 Research objectives

7.3.1 Characterise communications content

This objective required the identification of what constitutes communication content. To do this, it was necessary to generate data that can be used to characterise communication content; a method was devised to use an *indicator:action:artefact* system, where the *indicator* is the URL in the address bar; *action* is the activity e.g. content exchanged, and *artefact* is the data written to disk. This was achieved in **Chapters 4 and 5** where experiments were conducted on three social media platforms; Twitter, Facebook and Google Drive.

The experiments were designed to simulate real life user activity on these platforms; the range of actions performed during experiments involved the creation of user accounts, connecting to other entities within the network; and for Google Drive, creating, editing, and sharing documents [**Chapters 4 and 5**]. The range of activities carried out during the experiments were limited, and so were not considered wholly representative of typical user social media activity. This lack of representative data poses a challenge to both the digital forensics research community and practitioners because, the variations in the type of communication that may occur between entities on social media, results in the lack of realistic data to cover all possible scenarios that may occur during communication.

For practitioners especially, they are bound by the time constraints of an investigation, and may not have the time to devote to researching other potential user activities. Although other possible usage scenarios were considered during this research, it was impractical to attempt to perform experiments that will cover all possible ways social media can be used. Doing this would have extended the scope of the research,

7.3. RESEARCH OBJECTIVES

and impacted on time and resources. One of the considerations made during the experiments was devising a way of identifying what constitutes *communications content*. This was done to form a first level relationship between an action and the artefact it creates. The data generated was used to determine if the *indicator:artefact* pairs were consistent, and that they fit the definition of communication [Section 2.4] within the context of this research.

Through the analysis of the experimental data in **Chapters 4 and 5**, there was an opportunity to ascertain whether communication content within the bounds of this research was written to disk. This aims to clarify for the purposes of a live triage investigation, the artefacts that would most likely be of interest when trying to infer user action. It was observed that although some content was written to disk and could be recovered, the recovery process may involve file carving and keyword searches which were not considered ideal for this research.

This is because the fast paced nature of a live triage requires a less resource and time intensive approach [Roussev and Quates, 2012]. Avoiding exhaustive examinations presents both an opportunity and a challenge to practitioners during a live triage. It means that investigators have access to the data they need in a short period of time however, if the content of communication includes information that may be pertinent to the case, its availability would inevitably be delayed until a detailed forensic analysis is completed.

The analysis highlighted the importance of contextualising user activity. As evidenced in **Section 4.2.2** and **Section 5.2.2**, there is a possibility that data, independent of a user's direct interaction on social media may be cached. For the digital forensics research community, this raises ethical and privacy questions with regards to publishing full research results. For practitioners, this also raises privacy concerns that an investigator should be aware of, and put measures in place to exclude such data where it has no evidential value. This behaviour was observed on Windows 7, the OS of choice for the research experiments; and Windows 10, which was used for results validation and framework PoC testing; there was no evidence during the research, to determine if this behaviour is obtainable on other operating systems.

This research highlighted the importance of contextualising digital artefacts, when making inferences at a high level about user activity. This was achieved by corroborating the URL artefacts recovered with other evidence found on disk. While it has been demonstrated that the actual information exchanged may not be readily

accessible on disk, it sets expectations for the types of digital evidence that may be recoverable in a live triage; this also provides guidance on artefacts that may be found during further forensic analysis, to support the inferences made during the live triage.

7.3.2 Categorise artefacts generated from the use of social media

This objective required the categorisation of the social media artefacts recovered through the experiments conducted in **Sections 4.2.1, 4.3.1, 4.4.1, 5.2.1, 5.3.1, and 5.4.1**. The recovered artefacts were broadly classified as case-specific and context artefacts [**Section 3.10.1**]. Case-specific artefacts were recognised as those artefacts directly related to a case, as evidenced by the analysis of web browser data in **Chapters 4 and 5**. This group of artefacts are directly related to user activity on social media; and can be used to make inferences about the sequence of events that occurred. As discussed in **Section 7.3.1**, when attributing actions to a user, it is important to investigate the likelihood that some artefacts recovered may not be directly connected to a user communicating with another user.

Context artefacts on the other hand, were recognised as artefacts related to system activities often created as a result of user actions such as the system power on, login, application install, file access, etc. [**Section 3.10.1**]. These artefacts are used to corroborate the case-specific artefacts, by providing supporting information that shows the user initiated an action that may have led to the creation of social media artefacts in the web browser.

It was observed that a subset of case-specific artefacts were as a result of `localStorage`, or page caching used by the browser and the social media platform to improve user experience [**Section 4.2.2**]. It was also observed that these subset of case-specific artefacts are often not backed by URL information in the browser history; thus the approach taken was to recover the visited URL information from the cache, carry out a comparison to determine if the URL corroborates other artefacts from the browser cache or `localStorage`; and then contextualise user activity.

The affected subset of artefacts were analysed, and contextual information was used to determine the relationship to user activity [**Figure 4.5**]; the data connected to the user activities were recovered, and the data on unrelated third parties were excluded from the published research results. For the digital forensics research community this behaviour observed with the browser presents a challenge, considering the potential

7.3. RESEARCH OBJECTIVES

volume of data that could be recovered, researchers may be required to manually review and compare the data recovered from the cache/localStorage with URL information, to determine its probative value; abide by ethical guidelines and, ensure compliance with privacy laws.

This research highlighted that the nature of web browsers and social media platforms pose a challenge to digital investigations, as such updates/upgrades may impact on the format and location of artefacts. However, due to the focus on using visited URL artefacts to support or corroborate other artefacts, it may be possible to recover useful information as the URL syntax would remain the same (`<scheme>: <scheme-specific-part>`) [Berners-Lee, 1994; Berners-Lee et al., 1994]. This presents a level of standardisation, and stability for researchers and practitioners; as URL artefacts can still be of value irrespective of the underlying changes and can provide a means of determining user activity.

Several web browsers were considered at the start of this research, however, due to project time constraints, a decision was made to limit the test sample to two web browsers; Firefox and Google Chrome as they were the two browsers with a larger user base [Section 4.2.1]. Therefore, there is insufficient evidence to show that the analysed results, and behaviour observed, are common amongst other browsers. In spite of the limited sample size, the results from **Chapters 4 and 5** have demonstrated that it is possible to group social media artefacts based on user activity, and the identification of subcategories within the case-specific artefacts.

7.3.3 Develop a framework for the recovery of social media artefacts

This objective relied on the outcome of **Objective 2**: “Categorise artefacts generated from the use of social media”. The research was iterative in nature, thus the results from the manual analysis [Chapters 4 and 5] were required in order to have a baseline for how the framework would be designed. This was achieved in **Section 6.3** where components based on the case-specific and context artefact categories were identified. The aim is for a framework that can be used in a live triage environment. The requirement for a live triage depends on the nature of the criminal investigation [Sections 6.2; 6.3.3]; thus, when the need for actionable intelligence outweighs the requirement to “pull the plug” and take the computer back to the lab, a live triage may be the only reasonable option. To this end, the framework was purposed to be an investigative tool flexible enough to permit modification by practitioners to suit the nature of a live investigation.

This research reviewed a number of the existing live triage models [Section 2.6; 6.3.1] and found that although it is common practice in incident response investigations, it appears not be common practice in digital forensics investigations. Thus, there was the potential to develop flexible modular framework that can be used in the discipline of digital forensics. This research recognised the need to strike a balance between evidence recovery from a live environment and maintaining the integrity of the digital device.

Figure 7.1 is a graphical representation of the framework development process, which formed a basis for building a PoC to test its feasibility. The process consists of five stages: planning, theoretical framework, PoC build, framework testing, and implementation.

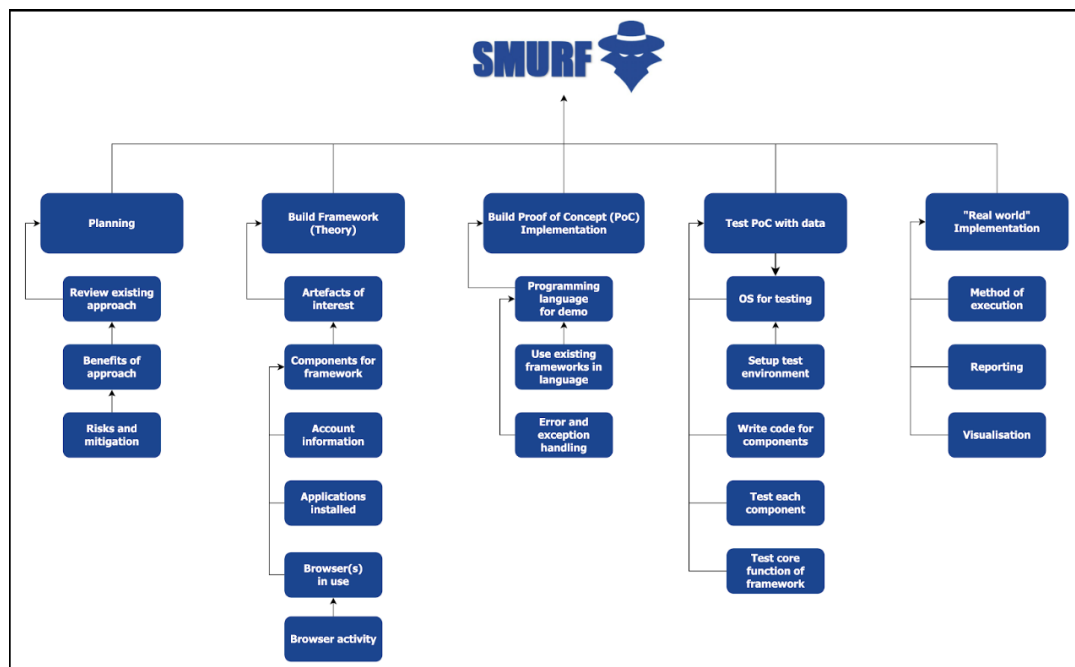


Figure 7.1: Graphical representation of the framework method

This research highlighted the focus of a live triage and the need for quick actionable intelligence in time sensitive investigations. The development and testing of a PoC was aimed at meeting this need, and not to test the performance of the code [Chapter 6.5]. The results from the PoC were compared to the results from the manual analysis [Chapters 4 and 5] to determine if the same categories of artefacts are recoverable [Section 6.5.3]. It was observed that communication content was not always available and may need to be recovered through a detailed forensic analysis. Part of the PoC development required a means of deploying the it for testing. It was determined that deployment via a portable device was an ideal

7.3. RESEARCH OBJECTIVES

solution [Section 6.5.2].

Figures 6.2(a) and 6.2(b) show how the components feed into the core of the framework. The flexibility of the framework is such that a practitioner is able to selectively use tools or plugins that are able to parse the artefact categories in the components, and get the results in a format that is “*quick to view, understand and use*”, such as a HTML report [Section 6.3.6]; viewable **not** on the target system, but on the investigator’s forensic device (e.g. laptop). The data generated during the experiments although limited in size and scope [Section 3.7], was deemed to be representative of a subset of real live activity thus, it is expected that such data may be found and recovered during an investigation. For researchers and practitioners, this highlights the need for realistic data for social media investigations however, this need might not be met in full due to privacy and ethical issues. Where real life data is used in research, measures need to be put in place to determine what data, if any is published.

During the framework testing, real life data from the researcher was made available. This was done to verify the ability to recover the artefacts identified from the limited experimental dataset [Section 6.5.3]; and also extended the scope of recovery. The framework was tested on Windows 10, and it was observed that due to the restrictions Windows places on some system files such as the SAM and Security hives, executing the PoC required administrator privilege in order to save a copy of the files for further analysis [Section 6.3.3 (app_config)]. Finally this objective as discussed in Section 6.4, produced a framework that is “*MMEG*”; modular, maintainable, easy to use, and generic.

It is pertinent to note that the research, and the framework development did not employ any Machine Learning (ML) or Artificial Intelligence (AI) techniques because, the aim was to make the framework implementation easy, and “*practitioner-friendly*”. In addition, the variability of artefact type as shown in the PoC, makes it generic enough so it can be adapted for use in a range of investigations such as used for social media, generic web browser forensic investigations, or user FTP activities.

7.3.4 Assess the reliability of social media artefacts

This objective required a criteria for assessing the reliability of the recovered web app artefacts. In a live triage, due to the nature of the investigation which involves interacting with the target device, it is important to show that the evidence being recovered has not been compromised [Section 6.4]. Although the evidence from

7.3. RESEARCH OBJECTIVES

a live triage is not intended for use in court, it is still necessary to ensure that any impact to the target device being investigated is clearly documented and justified. This is useful for researchers and practitioners within the digital forensics community as it ensures compliance with ACPO Guidelines and maintains the standards of admissibility of evidence.

Assessing reliability in the context of this research involved ensuring that the artefact can be used to infer user activity. This was achieved through a number of means; the first being the use of a multi tool validation technique during the manual analysis stages [**Chapters 4 and 5**], where a range of tools were used to recover and analyse the data. A comparison was made between the artefacts recovered, and the information recorded during the experiments [**Chapters 4 and 5; Section 3.6.1**]. This was necessary because a requirement for this research is the ability to infer user behaviour using data other than the content of user communication. As the comparison was done with a limited dataset, the behaviour of larger and varying datasets may differ.

To ensure the reliability of social media artefacts, the framework components were designed to use read-only processes when recovering data [**Section 6.4**]. In addition, a consideration was made to determine how any interaction that would alter the state of the target device would be identified, documented and justified. This was achieved through the use of Procmon [Russinovich, 2018] to log system activity. In a real world application, deploying the framework from a USB will undoubtedly interact with, and write data to the target device; an understanding of these possible changes as highlighted in this research would help an investigator document and justify the actions taken.

It is possible to use these documented changes to create a timeline of events that may be reviewed and possibly excluded during a detailed forensic analysis. For researchers and practitioners, this means that these changes can be classified and used as a baseline for process checks and validation during future live triage investigations. This research has shown that social media artefacts to a certain degree, can be reliable once the investigator is aware of the existence of unrelated artefacts [**Section 7.3.2**] and can contextualise, and exclude them where applicable.

7.4. RESEARCH QUESTION

7.3.5 Determine the relationships between communicating entities

This objective required the determination of the relationship between entities. This was achieved and demonstrated through the analysis of the experimental results in **Chapters 4 and 5**. Determining the relationships between entities involved demonstrating that there is a connection between two entities, using the features (attributes, properties) in a URL that indicate an entity initiated contact with another entity. Initiating contact as demonstrated in this research [**Chapters 4 and 5**] could be achieved through a range of activities such as searching for, and then adding as a contact (friend, following).

It may be necessary to establish that a relationship is more than just social acquaintances [U.S.D.C., NORTHERN DISTRICT OF ILLINOIS, 2018]; but can show active participation in unlawful activities. Due to the volume of data potentially available from the browser due to social media activity, manual (human) intervention would be required to filter the data and to make inferences about user relationships. The framework is not making the inferences but is providing the data and context for a practitioner to make an informed decision using the available data.

Using simulated data for social media relationship attribution presents a challenge to researchers. With the measures implemented by social media platforms to combat the abuse of their platforms [**Section 3.3**], there is a risk that experimental or test accounts may be disabled by the social media platform, causing setbacks and possible loss of time and data.

This research has shown that when the visited URLs are recovered and analysed, the frequency of contact between two entities can be determined. Regular contact in this context infers that the two entities may have a social relationship. To further validate this, other case-specific artefacts can be used to corroborate this inference as demonstrated in the results analysed in **Chapters 4 and 5**, and as discussed in **Chapter 6**, it is also possible to attribute actions and subsequently relationships to a user based on the output of the PoC framework.

7.4 Research Question

The research question as proposed in **Chapter 2** is:

“In the absence of the contents of user communication, can social media artefacts be used to reliably attribute actions to individual users? Can they be

*used to **link other entities involved in the communication?***

To answer the research question, the first step taken was to deconstruct the research question, and identify key terms within it [Section 3.3]. The significance of these key terms (in bold above) and how they were addressed are critically evaluated in this section. These key terms in the research question breaks it into four goals that were individually addressed although there were some overlaps.

7.4.1 absence of the contents of user communication

This is the first key part of the research question and involved defining communication within the bounds of this research [Section 2.4.1]. Communication was contextualised to imply the user’s intent. This definition formed a basis for determining what was within scope of *the contents of user communication*. For this research, this includes the text, images, videos, data files exchanged between two entities on social media. Determining what constitutes communication content was achieved through the experiments outlined in Chapters 4 and 5. However, there was a requirement to contextualise “absence”. In the context of live triage investigations, the absence of communication contents implies that the information is not readily available and its recovery would be resource and time intensive [Section 3.3].

It is worth noting that communication may be defined in different ways by different researchers, depending on the field of study. Thus for digital forensic research and investigations, it was important to apply strict boundaries by defining communication within the context of this research [Section 2.4].

7.4.2 social media artefacts

This is the second key part of the research question and required a method of identifying artefacts created as a result of social media activity [Section 3.3]. This was achieved by analysing the data recovered from the experiments conducted in Chapters 4 and 5. These results were used to define what the social media artefacts of interest were, and how they can be used to infer communication. This step highlighted as shown in Section 7.3.2, the need for categorising social media artefacts. Some of the artefacts analysed were created as a result of user initiated activity e.g. sending and receiving communication on a social media. Subsets of the data recovered were as a result of the underlying technologies used and background processes running during user activity on social media.

Researchers and investigators need to be aware of the limitations of relying on social media artefacts without thorough corroboration. The categorisation process required

7.4. RESEARCH QUESTION

attention to detail in order to identify artefacts directly related to the user; and demonstrated the need to have other artefacts that may be used to corroborate the data recovered. This research demonstrated that the social media artefacts can be used to infer action even when the “content” is absent.

7.4.3 reliably attribute actions

During the literature review, reliability was defined as *“the degree to which a specific artefact can be accurately attributed to a specific event”* [Section 2.7.3]. This research demonstrated that artefacts other than the content of user communication, can be used to attribute actions to a user during a live triage investigation. This was achieved through the manual analysis conducted in **Chapters 4 and 5**.

Determining how reliable these social media artefacts are was important to this research. The experiments conducted in this research presented a way to track and verify that the actions taken during the experiments were consistent [Section 3.9.2; **Chapters 4 and 5**]. In addition, it was necessary to maintain the ability to repeat the experiments with reproducible results. For a live triage investigation, it is important to note that repeatability, as an act itself, may not be possible due to ACPO principle 2, but the processes should be repeatable and produce same types of artefacts.

This is why the expected artefacts were identified, and a weighting system was used to determine the possibility of recovering the artefacts. This is also why the framework developed as part of this research is useful. In a court setting, the defence cannot perform live triage on the target device again [Section 3.5.5] but the documented processes used can be verified [ACPO, 2012] and the data can be validated by a full forensic analysis.

7.4.4 link other entities involved

The last key part of the research question addressed was the ability to link other entities involved in the communication. Social media platforms have a large user base for example, Facebook has billions of users [Facebook, 2018b]. While social media platforms use different metrics to identify and remove suspect accounts, digital investigators need improved ways of identifying, contextualising and then attributing actions to a user based on evidence of social media activity. This implies being able to make inferences about relationships between entities because users tend to form relationships with individuals with similar interests [Section 3.3].

To understand this, it was important to understand what links or relationships mean

in this context. The requirement was to use the social media artefacts recovered and infer user action (i.e. link between user and an action such as viewing a profile). This was achieved and demonstrated by the results in **Chapters 4 and 5** and by the framework (PoC); the results showed that it is possible to identify entities who the user had been communicating with, using a range of social media artefacts which corroborate each other. This research recognises that evidence corroboration is a vital part of contextualising behaviour, inferring and attributing relationships to a user [Section 6.3.3].

7.5 Conclusion

This Chapter evaluated how the research objectives were addressed and highlights the strengths, weaknesses and limitations of the methods and approaches taken to meet the objectives. It evaluated the key parts of the research question formulated as a result of the key problems identified in this research; and how they have been mapped into four specific tasks for this research. This was achieved by characterising what is considered communication content and also categorising the artefacts generated from the use of social media into case-specific and context artefacts. It discussed the framework development process, and highlights the importance of maintaining the integrity of the target device during a live triage, by ensuring that potential alteration is minimal and well documented.

This research reviewed existing live triage methods and approached each objective from the context of a live triage investigation. The caveats associated with social media artefact recovery and user attribution, emphasise the need to corroborate social media artefacts when assessing their reliability or making inferences about user action. Although the research data was not a complete representation of user activity on social media platforms, it creates the opportunity to research/investigate more use cases and operating systems. Finally, the modular and generic nature of the framework means there is scope for expansion and generalisability to include other forms of investigations.

The next chapter presents the research conclusions and future work.

8 Conclusions and future research

8.1 Research Conclusions

This chapter provides a brief description of the research contributions, discusses possible future work stemming from this research, and a summary of the research. The aim of this research was to substantiate user relationships during a live triage investigation, using artefacts other than communication content. Live triage investigations provide the means for an investigator to obtain quick, actionable intelligence at a crime scene. This intelligence can then be used to guide an investigation, including the process of identifying other digital devices that may be of forensic interest. Although the evidence recovered during a live triage is not necessarily intended for court, the established principles for handling and processing digital evidence must be followed to ensure that evidential integrity is maintained.

A primary concern for digital forensics practitioners when conducting a live triage is the modification of the target device which may be unavoidable; unlike in post-mortem digital forensic investigations where the tools are designed to prevent this. However, maintaining a culture of documenting and justifying every action taken during the evidence recovery process, including their potential impact on the target system (such as the creation of registry keys, event logs), would help maintain an audit trail to ensure that the process is repeatable and the reliability of the evidential data can be ascertained.

This research has reviewed the importance of a live triage investigation; as it pertains to time sensitive digital investigations. During this research it was observed that academic research into triage in digital forensics appears to be focused more on the post-mortem (off-site) triage activities. This demonstrates the need for more research, techniques and processes for live triage applications within the digital forensics community.

8.2 Contributions

This section highlights the research contributions as demonstrated in this research.

8.2.1 Established a method for the categorisation of social media artefacts

This research establishes a method for the categorisation of social media artefacts based on perceived user activity, highlighting the importance of social media artefacts to live triage investigations and digital forensics in general. Communication content was characterised [Section 3.3] and evidential data of interest from user social media activity was categorised into case-specific [Chapters 4 and 5] and context artefacts [Section 3.10.1]. Artefacts of interest were recovered using conventional forensic tools in order to define a baseline for specific sets of artefacts.

These categorisations came from the need to identify artefacts deemed to be key to an investigation, followed by the identification of other artefacts that can be used to corroborate them and give context to events leading up to the creation of a case specific artefact. The categorised data consists of artefacts that can be used to prioritise the analysis of live triage data. A subset of the results from this process, and a partial concept of the framework were presented in a two-staged model for investigating user social networking activity and has been published.

This research characterises communication content and highlights evidential data of interest from user social media activity. This involves prior recovery of such artefacts using conventional forensic tools to understand where the evidential material of interest is stored and a suitable technique for its recovery. The output is used to define a baseline for the set of artefacts that are expected to be recovered by the PoC framework.

8.2.2 Criteria for assessing the reliability of social media artefacts

This research proposes the criteria for assessing the reliability of social media artefacts in a live triage investigation. The criteria defined in Section 3.5.5 were the provision of a generalisable means of verifying experimental results for example, logging; this was demonstrated by the multi-tool verification approach [Chapters 4 and 5]. This is followed by the provision of adequate information on the process to enable replication manually or through automation [Chapters 4, 5 and 6]; to ensure that *each artefact can accurately be correlated or attributed to a specific event*. This was demonstrated in Chapters 4 and 5 with clear documentation and corroboration of artefacts of interest.

8.3. FUTURE WORK

8.2.3 Developed a framework for social media investigations

This research, based on the results from the categorisation of evidential data [Chapters 4 and 5], established that social media artefacts other than communications content can be used for the attribution of actions to a user [Section 4.6]. This resulted in the conceptualisation of a framework in Chapter 6, for use in the recovery of social media artefacts in a live triage, in order to verify if the same set artefacts can be recovered and used for attribution.

A PoC was developed to test its applicability and generalisability using real life data [Section 6.5]. This framework (SMURF) is an investigative tool designed to be modular and generic in nature [Section 6.4]; enabling its modification to suit the investigator's needs for specific cases. The framework demonstrates the possibility of user attribution based on case-specific and context artefacts, in the absence of communication content [Section 6.5.3].

8.3 Future Work

Live triage investigations are fast paced and require evidential data quicker than a full digital forensics investigation can provide [Section 3.10.2]. It is important to note as demonstrated in this research, that in-depth information on user activity is not the focus of a live triage as such data would usually be obtained during a detailed forensic analysis, often guided by the evidential data recovered during a live triage [Section 3.5.3].

To this end, several techniques were implemented to categorise the data of interest in a live triage. The technique used for the categorisation of social media artefacts was critically evaluated Section 7.3.2. This process was completed manually thus, further work is required to improve and automate the categorisation of social media artefacts.

A PoC was developed [Section 6.5] to test the framework developed in Section 6.3. It is however noted that further work can be done to explore other ways in which users may interact on social media, and actions such as URL manipulation i.e. altering the parameters in a URL. There is also scope for future work to investigate the extension of the framework for use in other live triage investigations.

Additional cache data such as image files were not recovered as part of the framework [Chapters 4 and 5]. Further work would be required for live investigations that

require the retrieval of image and video files. The framework can be improved to include techniques such as retrieval using known image hashes and skin tone detection.

This research did not implement artificial intelligence (AI) techniques [Section 7.3.3]. It is noted that further work can be done to improve the current functionality of the framework. With machine learning (ML) techniques, the individual components of the framework can be trained to adapt to a specific scenario using pre-set case information.

Finally, the framework produced by this research was tested on Windows 10 using real live data from Firefox and Google Chrome. Therefore, there is the potential for further research, to test the framework on operating systems and other browsers.

8.4 Summary

This research has highlighted that social media platforms present a range of communication opportunities, going beyond the days of relying on emails to keep in touch with family and friends. There are instances where a small percentage of the user community have misused the platform, using it for disagreeable activities [Section 2.4]. In a time sensitive investigation where social media may have been used to commit or facilitate a crime, rapid access to actionable intelligence is required. This can be achieved through the recovery of pertinent data via a live triage.

Live triage has been discussed at various points in this research, highlighting its importance as part of the digital forensics process. It is important to reiterate that live triage is not intended as a replacement for a full digital forensics investigation however, as the evidence recovered may be used to corroborate other outlier evidence, it is important to ensure that it is approached in a forensically sound manner. Several differing schools of thought for live triage were discussed in this research and all viewpoints agree on its use for the *acquisition of quick, actionable intelligence*.

This research was focused on user attribution, using social media artefacts recovered during a live triage; independent of the information exchanged during communication. It identified the limited research in this domain and the need for frameworks, methods and techniques for live triage in social media related investigations.

It demonstrated through the retrieval and analysis of social media artefacts using existing digital forensics tools, that it is possible to infer and attribute user action when the content of communication is absent. Through this analysis process, the

8.4. SUMMARY

research highlighted the need for a framework to recover these artefacts without reliance on conventional forensic tools as these are not wholly compatible with live investigations.

The manual analysis showed that some of the artefacts recovered were unconnected to the user, and emphasises how these artefacts can be verified through corroboration, in order to reliably attribute an action to a user.

This research produced a modular, generalisable framework, that provides multiple levels of functionality through its individual components. It has a reusable design, making it “*practitioner-friendly*” and easy to implement. The components as shown in this research, at a high level, implement the recovery of case-specific and context artefacts. The reusable design means it is not tied to a specific programming language or investigation, making it adaptable.

References

- 7safe (2014). The ACPO Good Practice Guide for Managers of e-Crime investigation. Technical Report. URL www.7safe.com.
- Access Data (2010). FTK Imager Lite 3.1.1. URL <https://marketing.accessdata.com/ftkimagerlite3.1.1>.
- ACPO (2012). Good Practice Guide for Digital Evidence. URL <http://library.college.police.uk/docs/acpo/digital-evidence-2012.pdf>. Version: 5.0.
- Agency (2015). Five internet trolls a day convicted in UK as figures show ten-fold increase. URL <https://www.telegraph.co.uk/news/uknews/law-and-order/11627180/Five-internet-trolls-a-day-convicted-in-UK-as-figures-show-ten-fold-increase.html>.
- Armstrong, H. L. and Forde, P. J. (2003). Internet anonymity practices in computer crime. *Information Management and Computer Security*, volume 11(5):pages 209–215.
- Arshad, H., Jantan, A., and Omolara, E. (2019). Evidence collection and forensics on social networks: Research challenges and directions. *Digital Investigation*, volume 28:pages 126–138.
- Ballenthin, W. (2019). python-evtx. URL <https://github.com/williballenthin/python-evtx>.
- Bancel, Jean-Rémy (2015). Chromagnon (SNSS Branch). GitHub. URL <https://github.com/JRBANCEL/Chromagnon/tree/SNSS>. Latest commit 2cbeeb1 on 28 Mar 2018.
- Barrett, D. and Kipper, G. (2010). *Virtualization and Forensics - A Digital Forensic Investigator's Guide to Virtual Environments*. Syngress.
- Bashir, M. S. and Khan, M. N. A. (2013). Triage in Live Digital Forensic Analysis. *The International Journal of Forensic Computer Science*, volume 1:pages 35–44. doi:10.5769/J201301005.
- Bello, M. and DiBlasio, N. (2013). Twitter: The new face of crime. USA Today. URL <http://www.usatoday.com/story/news/nation/2013/09/29/twitter-crime-dark-side/2875745/>.

REFERENCES

- Benson, R. (2019). Hindsight. URL <https://github.com/obsidianforensics/hindsight>.
- Berghel, H. (2003). The Discipline of Internet Forensics. URL www.giac.org.
- Berners-Lee, T. (1994). Working draft: Uniform Resource Locators (URL) A Syntax for the Expression of Access Information of Objects on the Network. URL <http://www.w3.org/Addressing/URL/url-spec.txt>.
- Berners-Lee, T., Fielding, R., and Frystyk, H. (1996). Hypertext Transfer Protocol – HTTP/1.0. URL <https://www.w3.org/Protocols/HTTP/1.0/spec.html>.
- Berners-Lee, T., Masinter, L., and McCahill, M. (1994). Uniform Resource Locators (URL) - RFC 1738. URL <http://www.ietf.org/rfc/rfc1738.txt>.
- Blumenbach, T. (2015). mozlz4a.py. URL <https://gist.github.com/Tblue/62ff47bef7f894e92ed5>.
- Brown, C. L. T. (2009). *Computer Evidence: Collection and Preservation*. Cengage Learning, Boston, MA 02210 USA.
- Bryce, C. and Miller, P. (2017). *Python Digital Forensics Cookbook*. Packt. ISBN 9781783987467.
- Caithness, A. (2012). The Forensic Implications of SQLite’s Write Ahead Log. *Digital Investigation - Digital evidence: if it’s there, we’ll find it*. URL <http://digitalinvestigation.wordpress.com/2012/05/04/the-forensic-implications-of-sqlites-write-ahead-log/>.
- Cambridge Dictionary (2019). Cambridge Dictionary [Online]. Cambridge University Press. URL <http://dictionary.cambridge.org/>.
- Cantrell, G., Dampier, D., Dandass, Y. S., Niu, N., and Bogen, C. (2012). Research toward a Partially-Automated, and Crime Specific Digital Triage Process Model. *Computer and Information Science*, volume 5(2). doi:10.5539/cis.v5n2p29. URL www.ccsenet.org/cisURL:<http://dx.doi.org/10.5539/cis.v5n2p29>.
- Cantrell, G. and Dampier, D. A. (2012). Implementing the Automated Phases of the Partially-automated Digital Triage Process Model. *Journal of Digital Forensics, Security and Law*, volume 7(4). URL <https://commons.erau.edu/jdfsl/vol7/iss4/5/>.
- Carrier, B. (2003). Defining Digital Forensic Examination and Analysis Tools using Abstraction Layers. *IJDE - International Journal of Digital Evidence*, volume 1(4).

REFERENCES

- Carrier, B. (2005). File System Forensic Analysis. chapter 1. Addison Wiley Professional, pages 3–8.
- Carrier, B. and Spafford, E. H. (2003). Getting Physical with the Digital Investigation Process. *International Journal of Digital Evidence*, volume 2(2):pages 1–20.
- Carrier, B. D. (2006). *A HYPOTHESIS-BASED APPROACH TO DIGITAL FORENSIC INVESTIGATIONS*. Ph.D. thesis, Purdue University, West Lafayette, Indiana.
- Carvey, H. (2011). *Windows Registry Forensics: Advanced Digital Forensic Analysis of the Windows Registry*. Syngress Publishing. ISBN 9781597495806.
- Carvey, H. (2018). RegRipper. URL <https://github.com/keydet89/RegRipper2.8>.
- Case, A. and Marziale, L. (2011). RegistryDecoder. URL <http://www.infosecisland.com/blogview/17867-Open-Source-Registry-Decoder-11-Tool-Released.html>.
- Casey, E. (2002). Error, Uncertainty, and Loss in Digital Evidence. *International Journal of Digital Evidence*, volume 1(2).
- Casey, E. (2004). Digital evidence and computer crime: Forensic Science, Computers and the Internet. *Elsevier Academic Press*:page 215.
- Casey, E. (2005). *Computer Crime and Digital Evidence: Forensic Science, Computers and the Internet*. Encyclopedia of Forensic and Legal Medicine. Elsevier, Oxford. ISBN 978-0-12-369399-0. doi:<https://doi.org/10.1016/B0-12-369399-3/00062-8>.
- Casey, E. (2010). *Handbook of Digital Forensics and Investigations*. Elsevier Academic Press, San Diego, California. ISBN 978-0-12-374267-4.
- Casey, E. (2011). *Digital Evidence and Computer Crime, Forensic Science, Computers and the Internet*, chapter Chapter 1: Foundations of Digital Forensics. Elsevier Inc., third edition edition.
- Casey, E. (2013). Triage in digital forensics. *Digital Investigation*, volume 10:pages 85–86.
- Casey, E., Ferraro, M., and Nguyen, L. (2009). Investigation Delayed Is Justice Denied: Proposals for Expediting Forensic Examinations of Digital Evidence. *Journal of Forensic Sciences*, volume 54(6):pages 1353–1364. doi:10.1111/j.1556-4029.2009.01150.x.

REFERENCES

- Chisum, W. J. and Turvey, B. E. (2000). Evidence Dynamics: Locard's Exchange Principle & Crime Reconstruction. *Journal of Behavioural Profiling*, volume 1(1).
- Chisum, W. J. and Turvey, B. E. (2007). *A History of Crime Reconstruction*. Crime Reconstruction. Elsevier.
- Cranfield University (2019). Research Ethics Policy.
- Crown Prosecution Service (CPS) (n.d.). Cyber / online crime. URL <https://www.cps.gov.uk/crime-info/cyber-online-crime>.
- Cusack, B. and Son, J. (2012). Evidence Examination Tools for Social Networks. In *10th Australian Digital Forensics Conference*. SRI Security Research Institute, Edith Cowan University, Perth, Western Australia, pages 33–40. doi:10.4225/75/57b3afc1fb861.
- Dance, F. E. X. (1967). *Towards a Theory of Human Communication (In Human Communication Theory: Original Essays)*. Holt, Rinehart and Winston, New York.
- DB4S Project (n.d.). DB Browser for SQLite. URL <http://sqlitebrowser.org/>.
- December, J. (1996). Units of Analysis for Internet Communication. *Journal of Communication*, volume 46(1):pages 14–38.
- DFRWS (2001). A Road Map for Digital Forensic Research: DFRWS Technical Report. Technical Report DTR - T001-01, DFRWS: Digital Forensic Research Workshop.
- Ellison, L. and Childs, M. (1999). Computer-Mediated Communications and Criminal Evidence. In *14th BILETA Conference: CYBERSPACE 1999: Crime, Criminal Justice and the Internet*.
- Facebook (2018a). Data Policy. URL <https://www.facebook.com/policy.php>.
- Facebook (2018b). Facebook Publishes Enforcement Numbers for the First Time. URL <https://about.fb.com/news/2018/05/enforcement-numbers/>.
- Facebook (2020a). Community Standards. <https://www.facebook.com/communitystandards/>.
- Facebook (2020b). Facebook Reports Second Quarter 2020 Results. <https://investor.fb.com/investor-news/press-release-details/2020/Facebook-Reports-Second-Quarter-2020-Results/default.aspx>.

REFERENCES

- Facebook Help Centre (2019a). How do I create a Facebook account? URL <https://en-gb.facebook.com/help/188157731232424?helpref=topq>.
- Facebook Help Centre (2019b). How do I use hashtags on Facebook?
- Facebook Inc. (2019). React A JavaScript library for building user interfaces. URL <https://reactjs.org/>.
- G Suite Learning Center (2019). Get started with Drive What can you do with Drive? URL <https://support.google.com/a/users/answer/9310246?hl=en>.
- Garfinkel, S. L. (2006). Forensic Feature Extraction and Cross-Drive Analysis. *Digital Investigation*, volume 3S:pages 71–81. URL <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.581.9553&rep=rep1&type=pdf>.
- Garfinkel, S. L. (2013). Digital media triage with bulk data analysis and bulk_extractor. *Computers & Security*, volume 32:pages 56–72. ISSN 0167-4048. doi:10.1016/J.COSE.2012.09.011. URL <https://www.sciencedirect.com/science/article/pii/S0167404812001472>.
- Gielen, M. and Bolzoni, D. (2014). Prioritizing Computer Forensics Using Triage Techniques. Technical Report. URL <https://essay.utwente.nl/65671/1/Gielen{ }MA{ }EWI.pdf>.
- Global Internet Forum to Counter Terrorism (GIFCT) (2019). Actions to Address the Abuse of Technology to Spread Terrorist and Violent Extremist Content. URL <https://gifct.org/press/actions-address-abuse-technology-spread-terrorist-and-violent-extremist-content/>.
- GlobalStats statcounter (2019). Desktop Browser Market Share Worldwide Apr 2016 - Apr 2018. URL <https://gs.statcounter.com/browser-market-share/desktop/worldwide/#monthly-201604-201804-bar>.
- Goodison, S. E., Davis, R. C., and Jackson, B. A. (2015). Digital Evidence and the U.S. Criminal Justice System: Identifying Technology and Other Needs to More Effectively Acquire and Utilize Digital Evidence. URL <http://www.jstor.org/stable/10.7249/j.ctt15sk8v3.1>.
- Google (a). Share Chrome with others. URL <https://support.google.com/chrome/answer/2364824?co=GENIE.Platform%3DDesktop&hl=en>.
- Google (b). Speed up Google Chrome. Google Chrome Help. URL <https://support.google.com/chrome/answer/1385029?co=GENIE.Platform%3DDesktop&hl=en>.

REFERENCES

- Google (c). User Data Directory. URL https://chromium.googlesource.com/chromium/src/+/_master/docs/user_data_dir.md.
- Google (2013). Get Started With Google Drive - Overview of Google Drive. URL <https://support.google.com/drive/answer/2424384?hl=en>.
- Google (2016). Drive Help: Google Drive storage. URL <https://support.google.com/drive/answer/6558?hl=en>.
- Google (2020a). Abuse Program Policies and Enforcement: Program Policies. <https://support.google.com/docs/answer/148505>.
- Google (2020b). Age requirements on Google Accounts. <https://support.google.com/accounts/answer/1350409>.
- Google (2020c). Google Drive Additional Terms of Service. <https://www.google.com/drive/terms-of-service/>.
- Google (2020d). LIST OF SERVICES AND SERVICE-SPECIFIC ADDITIONAL TERMS. <https://policies.google.com/terms/service-specific>.
- Google (2020e). Terms of Service. <https://policies.google.com/terms>.
- Google Developers (n.d). Manage files and folders overview | Drive REST API | Google Developers. URL <https://developers.google.com/drive/api/v3/about-files>.
- Hannan, M., Frings, S., Broucek, V., and Turner, P. (2003). Forensic Computing Theory & Practice: Towards developing a methodology for a standardised approach to computer misuse. In *First Australian Computer, Network and Information Forensics Conference*. pages 1–9.
- Haroon, S. and Carter, H. (2010). Facebook security measures criticised after Ashleigh Hall murder. The Guardian. URL <http://www.theguardian.com/uk/2010/mar/09/ukcrime-facebook>.
- Hay, A. S. (n.d.). JSONViewer. OS(1) - ash368 - Computer Forensics Software. URL <http://www.ash368.com/os1/4547935104>.
- Hitchcock, B., Le-Khac, N.-A., and Scanlon, M. (2016). Tiered forensic methodology model for Digital Field Triage by non-digital evidence specialists. doi:10.1016/j.diin.2016.01.010. URL <http://dx.doi.org/10.1016/j.diin.2016.01.010>.
- Hosmer, C. (2016). MPE+ | Using Python in Forensics Training Webinar Class. Webinar (YouTube).

REFERENCES

- Huber, M., Mulazzani, M., Leithner, M., Schrittwieser, S., Wondracek, G., and Weippl, E. (2011). Social Snapshots: Digital Forensics for Online Social Networks. In *Proceedings of the 27th Annual Computer Security Applications Conference, ACSAC '11*. ACM, New York, NY, USA. ISBN 978-1-4503-0672-0, pages 113–122. doi:10.1145/2076732.2076748. URL <http://doi.acm.org/10.1145/2076732.2076748>.
- Hubert, K. (2014). Evidence Collection From Social Media Sites. URL <https://www.sans.org/reading-room/whitepapers/legal/evidence-collection-social-media-sites-35647>.
- Humphries, M. (2010). Twitter to route all tweet links through security check. GEEK.COM. URL <https://www.geek.com/news/twitter-to-route-all-tweet-links-through-security-check-1124602/>.
- International Organization for Standardization (ISO) (2012). ISO/IEC 27037:2012 [ISO/IEC 27037:2012] Information technology — Security techniques — Guidelines for identification, collection, acquisition and preservation of digital evidence. URL <https://www.iso.org/standard/44381.html?browse=tc>.
- IOCE (2000). International Organization on Computer Evidence: Principles.
- JetBrains (n.d.). PyCharm for Anaconda. URL <https://www.jetbrains.com/pycharm/promo/anaconda/>.
- Jusas, V., Birvinskas, D., and Gahramanov, E. (2017). Methods and Tools of Digital Triage in Forensic Context: Survey and Future Directions. *Multidisciplinary Digital Publishing Institute (MDPI)*, volume 9(4):page 49. ISSN 2073-8994. doi: 10.3390/sym9040049. URL <http://www.mdpi.com/2073-8994/9/4/49>.
- Lee, H. C., Palmbach, T., and Miller, M. T. (2001). *Henry Lee's crime scene handbook*. Academic Press.
- Legislation.gov.uk (2000). Regulation of Investigatory Powers Act (RIPA) 2000, Chapter 23. URL <http://www.legislation.gov.uk/ukpga/2000/23/contents>.
- Legislation.gov.uk (2016). Investigatory Powers Act 2016. URL <http://www.legislation.gov.uk/ukpga/2016/25/contents>.
- Littlejohn, S. W. (1992). *Theories of Human Communication*. Wadsworth Pub. Co., Belmont, Calif., fourth edition. ISBN 0534161340.
- Magnet Forensics (2014). Forensic Analysis of Prefetch files in Windows. URL <https://www.magnetforensics.com/blog/forensic-analysis-of-prefetch-files-in-windows/>.

REFERENCES

- Magnet Forensics (2017). Digital Forensics: Artifact Profile – Google Chrome. <https://www.magnetforensics.com/blog/artifact-profile-google-chrome/>.
- Mandia, K. and Proise, C. (2001). *Incident Response: Investigating Computer Crime*. McGraw-Hill/Osborne, San Fransisco, California.
- Manning, J. (2014). *Definition and Classes of Social Media*. pages 1158–1162.
- Marcella, A. J. and Menendez, D. (2007). *Cyber Forensics: A Field Manual for Collecting, Examining, and Preserving Evidence of Computer Crime*. CRC PRESS - TAYLOR AND FRANCIS, NEW YORK.
- Matyszczyk, C. (2011). Teen admits trying to hire hit man on Facebook. cnet.com. URL <https://www.cnet.com/news/teen-admits-trying-to-hire-hit-man-on-facebook/>.
- McCarthy, N. (2011). Birmingham riots: West Midlands Facebook pair charged with inciting violence. BirminghamLive. URL <https://www.birminghammail.co.uk/news/local-news/birmingham-riots-west-midlands-facebook-159914>.
- McGuire, M. (2018). Into The Web of Profit: An in-depth study of cybercrime, criminals and money. https://www.bromium.com/wp-content/uploads/2018/05/Into-the-Web-of-Profit_Bromium.pdf.
- McKemmish, R. (1999). What is Forensic Computing? In *Trends and Issues in Crime and Criminal Justice*, No. 118. ISBN 0817-8542, pages 1–6.
- McKemmish, R. (2008). When is Digital Evidence Forensically Sound? *Advances in Digital Forensics*, volume IV:pages 3–15.
- MDN (2019). The Places database. MDN web docs. URL <https://developer.mozilla.org/en-US/docs/Mozilla/Tech/Places/Database>.
- MDN (2019). Using content preferences. MDN web docs. URL https://developer.mozilla.org/en-US/docs/Archive/Misc_top_level/Using_content_preferences.
- MDN (2019). Web Storage API. MDN web docs. URL https://developer.mozilla.org/en-US/docs/Web/API/Web_Storage_API.
- MercoPress (2019). Facebook refuses to suspend encryption plan despite impact on criminal activities. URL <https://en.mercopress.com/2019/10/04/facebook-refuses-to-suspend-encryption-plan-despite-impact-on-criminal-activities>.

REFERENCES

- Microsoft (2012). Audit logon events: Security Configuration Editor; Security Services | Microsoft Docs. URL [https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2003/cc787567\(v=ws.10\)](https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2003/cc787567(v=ws.10)).
- Microsoft (2018). Structure of the Registry,. URL <https://docs.microsoft.com/en-us/windows/desktop/sysinfo/structure-of-the-registry>.
- Microsoft Windows Dev Center (2018). Event Logging. URL <https://docs.microsoft.com/en-gb/windows/win32/eventlog/event-logging?redirectedfrom=MSDN>.
- Miller, P. and Bryce, C. (2019). *Learning Python for Forensics*. Packt, second edition edition. ISBN 9781789341690.
- MiTeC (n.d.). Windows Registry Recovery. <https://www.mitec.cz/wrr.html>.
- Monaghan, N. (2016). *Criminal Law Directions (4th edn)*. Oxford University Press. doi:10.1093/he/9780198753278.001.0001. URL <https://www.oxfordlawtrove.com/view/10.1093/he/9780198753278.001.0001/he-9780198753278>.
- Montasari, R. (2016). Formal Two Stage Triage Process Model (FTSTPM) for Digital Forensic Practice. *International Journal of Computer Science and Security (IJCSS)*, volume 10(2):pages 69–87. URL <https://pure.hud.ac.uk/en/publications/formal-two-stage-triage-process-model-ftstpm-for-digital-forensic>.
- Moore, K. (2014). Social media 'at least half' of calls passed to front-line police. BBC News. URL <https://www.bbc.co.uk/news/uk-27949674>.
- Morgan, T. D. (2009). The Windows NT * Registry File Format Version 0.4. Technical Report. URL <http://www.sentinelchicken.com/data/TheWindowsNTRegistryFileFormat.pdf>.
- Morris, S. L. A. (2013). *An Investigation into the identification, reconstruction, and evidential value of thumbnail cache file fragments in unallocated space*. phdthesis, Cranfield Defence and Security.
- Mozilla (n.d.). 7. Profiles & Backup. URL <https://www-archive.mozilla.org/start/1.5/faq/profile.html>.
- Mozilla Support (n.d.). Dedicated profiles per Firefox installation; What profile changes were made in Firefox 67? URL <https://support.mozilla.org/en-US/kb/dedicated-profiles-firefox-installation>.

REFERENCES

- Mukasey, M. B., Sedgwick, J. L., and Hagy, D. W. (2008). Electronic Crime Scene Investigation: A Guide for First Responders, Second Edition. U.S. Department of Justice, Office of Justice Programs, National Institute of Justice. URL <https://www.ncjrs.gov/pdffiles1/nij/187736.pdf>.
- Murr, M. (2007). The admissibility vs. weight of digital evidence | Forensic Computing. URL <https://forensicblog.org/the-admissibility-vs-weight-of-digital-evidence/>.
- National Police Chiefs' Council (NPCC) (2020). Digital Forensic Science Strategy. [https://www.fc.n.police.uk/sites/default/files/2020-07/Digital Forensic Science Strategy EMAIL VERSION ONLY.pdf](https://www.fc.n.police.uk/sites/default/files/2020-07/Digital_Forensic_Science_Strategy_EMAIL_VERSION_ONLY.pdf).
- NirSoft (2018a). ChromeCacheView. URL https://www.nirsoft.net/utils/chrome_cache_view.html.
- NirSoft (2018b). FullEventLogView. URL https://www.nirsoft.net/utils/full_event_log_view.html.
- NirSoft (2018c). MZCacheView. URL https://www.nirsoft.net/utils/mozilla_cache_viewer.html.
- Oh, J., Lee, S., and Lee, S. (2011). Advanced evidence collection and analysis of web browser activity. *Digital Investigation*, volume 8, Supplem(0):pages S62 – S70. ISSN 1742-2876. doi:<http://dx.doi.org/10.1016/j.diin.2011.05.008>. URL <http://www.sciencedirect.com/science/article/pii/S1742287611000326>.
- Oxford Dictionaries (2019). Oxford Dictionary. Oxford University Press. URL <https://www.lexico.com/en>.
- Oxford Reference (2016). A Dictionary of Computer Science. Oxford University Press.
- PACE 1984 (1984). Police and Criminal Evidence Act. URL <http://www.legislation.gov.uk/ukpga/1984/60/contents>.
- Paganini, P. (2014a). Google Drive once again exploited in a sophisticated phishing attack. <https://securityaffairs.co/wordpress/29766/cyber-crime/google-drive-phishing-attack.html>.
- Paganini, P. (2014b). Sophisticated Google Drive Phishing Scam is becoming popular. <https://securityaffairs.co/wordpress/25158/cyber-crime/google-drive-phishing-scam.html>.

REFERENCES

- Parsonage, H. (2009). Computer Forensics Case Assessment and Triage. Technical Report. URL <http://computerforensics.parsonage.co.uk/triage/ComputerForensicsCaseAssessmentAndTriageDiscussionPaper.pdf>.
- Pennsylvania Superior Court (2017). COMMONWEALTH v. MANGEL WL 1322179. Nos. 703 WDA 2017, 704 WDA 2017. URL <https://www.leagle.com/decision/inpaco20180315753>.
- Pereira, M. T. (2009). Forensic analysis of the Firefox 3 Internet history and recovery of deleted SQLite records. *Digital Investigation*, volume 5(3-4):pages 1–11. doi: 10.1016/j.diin.2009.01.003. URL <http://linkinghub.elsevier.com/retrieve/pii/S1742287609000048>.
- Pexels (2020). Terms and Conditions (“Terms”): 5.5. Pexels License. <https://www.pexels.com/terms-of-service/>.
- Pixabay (2020). Terms of Service: License for Content – Pixabay License. <https://pixabay.com/service/terms/>.
- Police Executive Research Forum (PERF) (2018). NEW NATIONAL COMMITMENT REQUIRED: The Changing Nature of Crime And Criminal Investigations. CRITICAL ISSUES IN POLICING SERIES.
- Pollitt, M. (1995). Computer Forensics: An Approach to Evidence in Cyberspace,. In *National Information Systems Security Conference*,, volume 2. pages 487–491.
- Porter, J. (2014). Prefetch Analysis in Live Response. URL <https://www.symantec.com/connect/blogs/prefetch-analysis-live-response>.
- Python.org (2019). Python 3. URL <https://www.python.org/downloads/release/python-374/>.
- Raphael, A. (2021). Allegedly sinister use for Google Drive. <https://www.themorningbulletin.com.au/news/allegedly-sinister-use-for-google-drive/4205740/>.
- Reith, M., Carr, C., and Gunsch, G. (2002). An Examination of Digital Forensics Models. *International Journal of Digital Evidence*, volume 1(3):pages 5–12.
- Robertson-Steel, I. (2006). Evolution of triage systems. *Emergency medicine journal*, volume 23(2):pages 154–155. doi:doi:10.1136/emj.2005.030270.

REFERENCES

- Rogers, M. K., Goldman, J., Mislan, R., Wedge, T., and Debrota, S. (2006). Computer Forensics Field Triage Process Model. *Journal of Digital Forensics, Security and Law*, volume 1(2). doi:10.15394/jdfsl.2006.1004. URL <https://commons.erau.edu/cgi/viewcontent.cgi?article=1004&context=jdfsl>.
- Roussev, V. and Quates, C. (2012). Content triage with similarity digests: The M57 case study. *Digital Investigation*, volume 9:pages S60–S68. doi:10.1016/j.diin.2012.05.012.
- RSA (2016). Hiding in Plain Sight - The Growth of Crime in Social Media Part 1. <https://www.rsa.com/content/dam/en/white-paper/hiding-in-plain-sight-the-growth-of-cybercrime-in-social-media-part-1.pdf>.
- Russinovich, M. (2018). Process Monitor - Windows Sysinternals | Microsoft Docs. URL <https://docs.microsoft.com/en-us/sysinternals/downloads/procmon>.
- Sammes, A. J. and Jenkinson, B. (2007). *Forensic Computing*. Springer-Verlag London Limited.
- Scambray, L. V., J. and Sima, C. (2011). *HACKING EXPOSED Web Applications*. McGraw-Hill, third edition.
- Schatz, B. (2007). Digital Evidence: Representation and Assurance.
- Science and Committee, T. S. (2019). Forensic science and the criminal justice system: a blueprint for change. <https://publications.parliament.uk/pa/ld201719/ldselect/ldsctech/333/333.pdf>.
- Seals, T. (2015). Europol Busts Operation Triangle in Six-Nation Raid. Info Security Magazine. URL <https://www.infosecurity-magazine.com/news/europol-busts-operation-triangle/>.
- Select Committee on Communications (2014). CHAPTER 2: SOCIAL MEDIA AND THE LAW . URL <https://publications.parliament.uk/pa/ld201415/ldselect/ldcomuni/37/3702.htm>.
- Select Committee on Science and Technology (2018). Corrected oral evidence: Forensic Science. <http://data.parliament.uk/writtenevidence/committeeevidence.svc/evidencedocument/science-and-technology-committee-lords/forensic-science/oral/93059.html>.

REFERENCES

- Shafqat, N. (2016). Forensic Investigation of User's Web Activity on Google Chrome using various Forensic Tools. Technical Report 9. URL <http://paper.ijcsns.org/07{ }book/201609/20160919.pdf>.
- Shaw, U., Das, D., and Mehdi, S. P. (2016). Social Network Forensics: Survey and Challenges. *International Journal of Computer Science and Information Security (IJCSIS)*, volume 14(11):pages 310–316.
- Shklar, L. and Rosen, R. (2009). *Web Application Architecture: Principles, Protocols and Practices*. John Wiley & Sons Ltd., second edition. ISBN 978-0-470-51860-1.
- Sommer, P. (1999). Intrusion Detection Systems as Evidence. *Computer Networks*, volume 31(23–24):pages 2477–2487.
- Southern District of New York (S.D.N.Y) (2012). U.S. v. MEREGLIDO 2012 WL 3264501.
- Taylor, C. (2013). Mother Tried to Sell Her Kids on Facebook for \$4,000. MashableUK. URL <https://mashable.com/2013/03/11/facebook-kids-sale-arrested/?europe=true>.
- Taylor, M., Haggerty, J., Gresty, D., Almond, P., and Berry, T. (2014). Forensic investigation of social networking applications. *Digital Investigation*, volume 11:pages 9–16.
- Telerik (2018). Fiddler - Free Web Debugging Proxy - Telerik. URL <https://www.telerik.com/fiddler>.
- The Chromium Projects (). Disk Cache. URL <https://www.chromium.org/developers/design-documents/network-stack/disk-cache>.
- The Chromium Projects (n.d). User Experience - Omnibox. URL <https://www.chromium.org/user-experience/omnibox>.
- The Pallets Projects (2019). Jinja. URL <https://jinja.palletsprojects.com/en/2.10.x/>.
- TWGECSI (2001). Technical Working Group for Electronic Crime Scene Investigation - Electronic Crime Scene Investigation: A Guide for First Responders.
- Twitter Help Center (2019). New user FAQs. URL <https://help.twitter.com/en/new-user-faq>.
- Twitter Help Center (2020). The Twitter Rules. <https://help.twitter.com/en/rules-and-policies/twitter-rules>.

REFERENCES

- Ultimate IT Security (n.d.). Windows Security Log Encyclopedia. URL <https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/default.aspx>.
- U.S.D.C., NORTHERN DISTRICT OF ILLINOIS (2018). UNITED STATES OF AMERICA v. ASHRAF AL SAFOO, also known as Abu Al’-Abbas Al-Iraqi, Abu Shanab and Abbasi, AO 91 (Rev. 11/11) Criminal Complaint. UNITED STATES DISTRICT COURT, NORTHERN DISTRICT OF ILLINOIS, EASTERN DIVISION.
- VMware (2019). VMware Timeline. URL <https://www.vmware.com/uk/timeline.html>.
- Wall, D. S. (1999). On the Politics of Policing the Internet: Striking the right balance. In *14th BILETA Conference: “Cyberspace 1999: Crime, Criminal Justice and the Internet”*. pages 1–12.
- Wiles, J. and Reyes, A. (2007). *Incident Response: Live Forensics and Investigations*, chapter 7. The Best Damn Cybercrime and Digital Forensics Book Period. Syngress, pages 89–109.
- Wilson, C. (2009). DCode v4.02a (Build:9306). Digital Detective. URL <https://www.digital-detective.net/dcode/>.
- WinPython (2019). winpython. URL <https://github.com/winpython/winpython>.
- Woan, M. (2013). PrefetchForensics. URL <https://github.com/woanware/woanware.github.io/blob/master/downloads/PrefetchForensics.v.1.0.4.zip>.
- Wood, C. (2018). WhatsApp photo drug dealer caught by ‘groundbreaking’ work. URL <https://www.bbc.co.uk/news/uk-wales-43711477>.
- World Wide Web Foundation (2019). History of the Web. URL <https://webfoundation.org/about/vision/history-of-the-web/>.
- X-Ways Software Technology AG (2018). WinHex. URL <https://www.x-ways.net/winhex/>.
- Yang, B., Li, N., and Jiang, J. (2016). *A New Triage Process Model for Digital Investigations*. ISBN 9781467396134. URL <https://ieeexplore.ieee.org/ielx7/7861599/7867100/07867302.pdf?tp=&arnumber=7867302&isnumber=7867100&ref=aHR0cHM6Ly9pZWVleHBsb3JlLmllZWUub3JnL2RvY3VtZW50Lzc4NjczMDI=>.

REFERENCES

- Zeng, D., Chen, H., Lusch, R., and Li, S. (2010). Social Media Analytics and Intelligence. *IEEE Intelligent Systems*, volume 25(6):pages 13–16. ISSN 1941-1294. doi:10.1109/MIS.2010.151.
- Zimmerman, E. (2019). Prefetch Explorer Command Line - PECmd version 1.3.4.5. URL <https://github.com/EricZimmerman/PECmd>.

Appendices

Appendix A: SMURF Example Output

A.1 Data Categories and Locator Information

Table A.1: Examples of artefacts recovered by the PoC

Item	Data Categories	Locator Information	Example Output (Extracts)
Visited URLs	<ul style="list-style-type: none"> • Date and time of visit, • Platform visited, • User action e.g. <ul style="list-style-type: none"> – login, – compose tweet, – profile visits, – searches, – downloaded content, – documents accessed. 	<p>SQL query: [see Section 4.2.2]</p> <pre># store query output in a file (e.g. sqlout.csv) SELECT datetime(visit_date/1000000, 'unixepoch') as visitDate, url, title FROM moz_places, moz_historyvisits WHERE moz_places.id = moz_historyvisits.place_id ORDER BY visitDate</pre> <p># example keywords to search for in query output keywords → “drive docs.google, facebook, twimg, twitter”</p> <p>Regex: <code>r'((?<=\/)[^:\/\s]*[?*\\w\\w]+)'</code> # Regex search pattern: exclude the URL protocol, colon (:), slash (/), and white spaces. # return alphanumeric word characters plus underscores (_).</p> <p>Example Python implementation:</p> <pre>with open(sqlout.csv): readCSV = csv.reader(sqlout.csv, delimiter=',') for row in readCSV: for item in row: social_media_platform = ['twitter', \ 'pbs.twimg', 'facebook', \ 'drive.google', 'docs.google'] for host in social_media_platform: if host in item: pattern = r'((?<=\/)[^:\/\s]*[?*\\w\\w]+)' regexp = re.compile(pattern, \ re.IGNORECASE) matches = regexp.findall(item)</pre>	<p>Ex. 1: 2018-10-27 15:11:17, twitter.com, login? redirect_after_login=%2Fxxx%2Fwith_replies</p> <p>Ex. 2: 2020-08-12 18:18:12, twitter.com, compose, tweet , (12)Xxx Xxx on Twitter: "Hilarious. Do you remember this episode? 🤔"</p> <p>Ex. 3: 2020-01-30 12:00:54, en-gb.facebook.com, Xxxxx, photos</p> <p>Ex. 4: 2020-03-03 00:00:41, docs.google.com, document, d, 1sxxxxxx_Jxx_Ixxo, edit</p> <p>Ex. 5: 2018-12-01 18:41:58, pbs.twimg.com, media, Be7x_xxxYR?format=jpg&name=small, Be7x_xxxYR.jpg</p>

Continued on next page...

Table A.1 ... continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Session information	<ul style="list-style-type: none"> • Browser session start and last accessed times, • Visited URLs, • The status of windows and tabs, • The order of page/site navigation 	<p>Firefox (≥ 56):</p> <p>Decompress the .jsonlz4 or .baklz4 file and write the JSON output to a file for processing.</p> <p>Parse the JSON output for the values of: “windows, tabs, url, title, referrer, last accessed”.</p> <p>Example Python implementation:</p> <pre>[SNIP] get_file_data = open(lz4_file, "rb") read_bytes = get_file_data.read() get_sig= read_bytes[:8] # verify header if bytearray(get_sig).startswith(b"mozLz40\0"): json_data = json.loads(lz4.decompress \ (read_bytes[8:]).decode("utf-8")) # open decompressed json output with open(json_data_output, 'a+'): printable = set(string.printable) jsd = filter(lambda x: x in printable, \ json_data) json_data_output.write(str(jsd)) for win in json_data.get("windows"): for tab in win.get("tabs"): i = int(tab.get("index"))-1 urls = tab.get("entries")[i] \ .get("url") title = tab.get("entries")[i] \ .get("title") last_accessed = win.get("tabs")\ [i].get("lastAccessed") tab_last_accessed = dt.datetime \ .utcfromtimestamp \ (last_accessed//1000.0)</pre>	<p>Ex. 1:</p> <p>URL: https://twitter.com/XXX/followers; Page Title: (2)People following Xxx Xxx (@XxxXxx)/Twitter; Referrer: None</p> <p>Ex. 2:</p> <p>URL: https://twitter.com/XxxXxx; Page Title: Xxx (@XxxXxx)/ Twitter; Referrer: None; Last Accessed: 2020-04-11 16:28:01</p>

Continued on next page...

Table A.1 ... continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Session information	<ul style="list-style-type: none"> • Browser session start and last accessed times, • Visited URLs, • The status of windows and tabs, • The order of navigation 	<p>Chrome: [see Section 6.3.4.5]</p> <p>Read the Current/Last Session, and Current/Last Tabs files using an SNSS parser, write the output to a file for processing.</p> <p># example keywords to search for in Chrome session files</p> <p>keywords → “SelectedNavigationInTab, UpdateTabNavigation”</p> <p>Match and return lines that start with the keywords</p> <p>Example Python implementation:</p> <p>[SNIP]</p> <pre># get contents of current session subprocess.run(["python", \ r"chromagnonSession.py", session_file, \ ">", session_output], shell=True) with codecs.open(session_output, 'r', \ encoding='utf-8',errors='ignore') for line in session_output: if line.startswith ("UpdateTabNavigation"): session_data = session_output.readlines() [SNIP]</pre>	<p>Ex. 1:</p> <pre>UpdateTabNavigation - Tab: 1532, Index: 0, URL: https://docs.google.com/document/d/1 xxxxxxx_Jxx_Ixxo/edit?, Title: <DocumentTitle > - Google Docs, State: None, Transition: 0</pre>

Continued on next page...

Table A.1 ...continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Form field data	<ul style="list-style-type: none">• Usernames,• User IDs,• Phone numbers,• Email addresses,• Search terms.	<p>SQL query:</p> <pre># store query output in a file (e.g. form_history.csv) SELECT fieldname, value, datetime(firstUsed /1000000, 'unixepoch')as timeFirstUsed, datetime (lastUsed/1000000, 'unixepoch')as timeLastUsed, timesUsed, guid FROM moz_formhistory WHERE firstUsed NOT NULL ORDER BY firstUsed</pre> <p>Keyword search:</p> <pre># example keywords to search for in query output keywords → "twitter, facebook, google" Match and return lines in the output file that contain keywords.</pre> <p>Example Python implementation:</p> <pre>[SNIP] # get form data entries. # open the csv file using the alias 'f_hist' with open(form_history.csv, 'r')as f_hist: form_data_entries = [] for line in f_hist: form = f_hist.readlines() for item in form: if "facebook" in item or "google" in item \ or "twitter" in item: data_entry = form_data_entries \ .append(item)</pre> <p>[SNIP]</p>	<p>Ex. 1:</p> <pre>searchbar-history, bbcgoodfood twitter, 2020-01-30 11:45:52, 2020-01-30 11:45:52, 1, J7...We</pre> <p>Ex. 2:</p> <pre>session[username_or_email], xxx@gmail.com, 2020-08-23 15:05:12, 2020-08-23 15:05:12, 1, sD...5L</pre>

Continued on next page...

Table A.1 ... continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Cookies	<ul style="list-style-type: none"> • Date & time cookie was set, • Date & time cookie was last accessed, • Host(s) that set the cookie, • Name & value of cookie. 	<p>SQL query: [see Section 4.2.2]</p> <pre># store SQL output in a file (e.g. cookies.csv) SELECT datetime(creationTime/1000000, 'unixepoch')as createTime, datetime(lastAccessed/1000000, 'unixepoch')as lastAccessedTime, datetime(expiry/1000000, 'unixepoch')as expiryTime, host, name, value FROM moz_cookies WHERE lastAccessedTime NOT NULL AND moz_cookies.host LIKE '%twitter%' or moz_cookies.host LIKE '%google%' or moz_cookies.host LIKE '%facebook%' ORDER BY moz_cookies.lastAccessed</pre> <p>Example Python implementation:</p> <pre>[SNIP] # get cookie details # open the csv file using the alias 'c_file' with open(cookies.csv, 'r')as c_file: # create empty list to add the cookie values cookie_entries = [] for line in c_file: cookies = c_file.readlines() for item in cookies: if "facebook" in item or "twitter" \ in item or "drive.google" in item \ or "docs.google" in item: social_media_cookie = cookie_entries \ .append(item)</pre> <p>[SNIP]</p>	<p>Ex. 1:</p> <pre>2018-12-01 18:35:02, 2020-08-30 12:59:31, 1970-01-01 00:29:15, .twitter.com, remember_checked_on, 1</pre> <p>Ex. 2:</p> <pre>2019-12-28 09:04:55, 2019-12-28 09:04:55, 1970-01-01 00:26:17, google.com, S, spreadsheet_forms=4xxx-8xxx-exxx_Fxxx</pre>

Continued on next page...

Table A.1 ... continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Cached data	<ul style="list-style-type: none"> • Cache data size, • Last modified timestamp, • Content cached by social media platform e.g. <ul style="list-style-type: none"> – image files, – originating URL. 	<pre># interpret the structure of the entry file # store the output of the entry files in a file (e.g. cache-entries.csv) Use the format specification for the cache entry file. Parse and extract the values for: "file size, last modified time, cache expiration, fetch count, last fetched, start of cache metadata, key size, key and key hash" # example keywords to search for in the cache output file keywords → "facebook, drive docs.google, twitter" Return content with keyword in the cache key field. Example Python implementation: # open the csv file using the alias 'entry_file' with open(cache_entries.csv, 'rb') as entry_file: entry_file.seek(0, 0) chunk_size = 256 * 1024 entry_file.seek(-4, os.SEEK_END) [SNIP] if version > 1: last_modified_int = struct.unpack('>I', \ entry_file.read(4))[0] last_modified = datetime.utcfromtimestamp \ (last_modified_int).strftime \ ('%a, %d %b %Y %H:%M:%S') key = entry_file.read(key_size) key_hash = hashlib.sha1(key).hexdigest() \ .upper() key = key.decode("ascii") if "https://facebook.com" in key or \ "https://docs.google.com" in key: key = key</pre>	<p>Ex. 1: [SNIP] Cache file is written in version: 3 Cache entry file size: 302 Cache last modified: Sat, 28 Dec 2019 09:04:56 Start of cache metadata: 0 Cache file fetch count: 1 Cache file last fetched: Sat, 28 Dec 2019 09:04:55 Key size: 43 Key (URL): ~predictor-origin,:https://docs.google.com/ Key hash (SHA1 hash of url): F982...7B7</p> <p>Ex. 2: [SNIP] parser: hindsight/2.4.0, interpretation: format: jpg name: orig usageCounter: 1, reuseCounter: 0, state: 0, keyLength: 78, keyAddress: 0, data: [null, null], key: https://pbs.twimg.com/ad_img/103...162/Nxxx? format=jpg&name=orig, file_size: 38348, location: data_2 [5939200]; f_000152, datetime: 2018-08-19T11:50:52.433680+00:00, timestamp_desc: Last Visit Time, original_url: https://pbs.twimg.com/ad_img/103...162/Nxxx- ?format=jpg&name=orig, cached_state: Normal (data cached)</p>

Continued on next page...

Table A.1 ... continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Storage access permissions	<p>Domains with permission to:</p> <ul style="list-style-type: none"> • Store data locally using the Storage Access API, • Open pop-up windows, • Display notifications. 	<p>SQL query:</p> <pre># store SQL output in a file (e.g. perms.csv) SELECT origin, type, datetime(modificationTime /1000, 'unixepoch')as timeModified, datetime(expireTime/1000, 'unixepoch')as expiryTime FROM moz_perms ORDER BY timeModified</pre> <p># example keywords to search for in query output keywords → “twitter, facebook, google” Search for, and return records matching “keywords”</p> <p>Example Python implementation: [SNIP]</p> <pre># store the recovered records in a list recovered_records = [] for each_row in perms.csv: # store each field from a record in a # dictionary perms_record = { "origin": each_row[0], "type": each_row[1], "timeModified": each_row[2], "expiryTime": each_row[3], } # append to the list of records recovered_records.append(perms_record)</pre>	<p>Ex. 1: https://twitter.com,desktop-notification, 2018-08-19 08:43:01,1970-01-01 00:00:00</p> <p>Ex. 2: https://docs.google.com, storageAccessAPI, 2020-04-11 09:17:15, 2020-05-11 09:17:15</p>

Continued on next page...

Table A.1 ... continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Local Machine user(s)	<ul style="list-style-type: none"> • Active logon sessions, • Logged on user(s), • Date and time of logon, • Logon type (e.g. remote, interactive), • Registered owner of system, • System uptime, • Kernel version, • System volumes. 	<p># Determine accounts and logged in users on the local machine</p> <p># store the output in individual files</p> <p>Run the PsLoggedon, logonsessions, and PsInfo commands from the SysInternals suite.</p> <p># example keywords to search for in the output keywords → “Logon time, User name, SID, Logon type” Return content that contain keyword with their values.</p> <p>Use the <code>reg.exe save hklm\sam</code> command to save a copy of the SAM file, parse the file for the logged in user’s “ProfileImagePath”.</p> <p>Example Python implementation:</p> <pre>[SNIP] # Get User SID sid_list = [] for i in range(QueryInfoKey(sid_key)[0]): name = EnumKey(sid_key, i) sid_list.append(name) profile_img_path = sid_list[-1] # Tools from Windows SysInternals: # logonsessions, PsLoggedon, PsInfo subprocess.run([r"third_party\bin\sysinternals \ \logonsessions64", "-accepteula", \ ">", logonsession.txt], shell=True) subprocess.run([r"third_party\bin\sysinternals \ \PsiLoggedon64", "-accepteula", ">", \ loggedonuser.txt], shell=True) subprocess.run([r"third_party\bin\sysinternals \ \PsiInfo64", "-accepteula", \ ">", system_info.txt], shell=True)</pre>	<p>Ex. 1:</p> <p>Users logged on locally: 27/08/2020 03:16:49 DESKTOP-XXX<LocalUser> No one is logged on via resource shares.</p> <p>Ex. 2:</p> <p>Logon session 00000000:000f9571: User name: DESKTOP-XXX<UserName> Auth package: NTLM Logon type: Interactive Session: 1 Sid: S-1-5-21-X-X-...0751-1001 Logon time: 27/08/2020 03:16:48</p> <p>Ex. 3:</p> <p>System information for \DESKTOP-XXX: Uptime: 3 days 11 hours 5 minutes 47 seconds Kernel version: Windows 10 Education, Multiprocessor Free, Product type: Professional Registered owner: <name> or <email address>, System root: C:\Windows</p> <p>Ex. 4:</p> <p>ProfileImagePath:S-1-5-X-X-...0751-1001</p>

Continued on next page...

Table A.1 ... continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Mounted devices	<ul style="list-style-type: none"> Mounted devices including network shares, Date and time device was connected, Device name, Device vendor and product ID, System generated log (e.g. driver installation for devices plugged in by user). 	<p># store output in a file Extract the values from the System\MountedDevices registry key. Extract information about USB devices from setupapi.dev.log.</p> <p># example keywords to search for in the output keywords → “_??_usbstor#, disk&ven, &prod, ven, vid”. Return lines in the output file that contain the keywords.</p> <p>Example Python implementation: [SNIP] # MountedDevices registry value = mounted_val if bytearray(mounted_val).startswith \ (b'_x00?x00?x00_x00Ux00Sx00B'): mounted_device = mounted_val m_device = mounted_device.decode("utf-16", \ "ignore").replace(" ", "")</p> <p> # open a text file using the alias 'mounted' with open(mounted.txt, 'a+') as mounted: mounted.write(m_device + '\n')</p> <p>[SNIP] for each_line in log_file: # an adaptation of setupapi_parser.v2.py change_case = each_line.lower() if 'device install (hardware initiated)' in \ change_case and ('_??_usbstor#' \ in change_case or [...] \ 'vid' in change_case): device_name = each_line.strip() # For clarity, strip '>>>' from the results device_list.append((device_name.strip \ ('>>> '), time_stamp))</p>	<p>Ex. 1: 2020/02/23 07:52:59 - USB\VID_091E&PID_00035 &9882ff5&0&3</p> <p>Ex. 2: 2020/08/23 10:17:23 _??_USBSTOR#Disk&Ven_Generic &Prod_Flash_Disk&Rev_8.07#D785E683A1BEFD3F5463 &0#53f56307-b6bf-11d0-94f2-00a0c91efb8b</p>

Continued on next page...

Table A.1 ... continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Recent documents	<ul style="list-style-type: none">Document(s) recently opened or executed by the current user,<ul style="list-style-type: none">images,text files,executable files etc.	<p>Read and parse the contents of the Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs registry key.</p> <p>The data type of the value is REG_BINARY. Decode and convert the individual binary values to string.</p> <p># RegEx search pattern</p> <p>Search pattern to return the decoded values (filename and extension) in the RecentDocs key:</p> <pre>r'((^\\d \\w+ \\d \\s)\\.\\w\\w\\w)\\S\\S(\\w+\\.lnk)\\S\\S(\\w+\\.lnk)\\b'</pre> <p># RegEx search pattern: return alphanumeric word characters, non-whitespace characters plus</p> <p># a literal match of the characters 'lnk' (file shortcuts)for accessed documents.</p> <p>Example Python implementation:</p> <pre>[SNIP] convert_value = codecs.decode recent = convert_value(recent_docs_value, \ "utf-8", "ignore") recent = "".join(i for i in recent if i in \ string.printable and 31 < ord(i) < 127) docs_matched = re.findall(r'((^\\d \\w+ \\d \\s)\\.\\w\\w\\w)\\S\\S(\\w+\\.lnk)\\S\\S(\\w+\\.lnk)\\b', recent, flags=re.IGNORECASE re.MULTILINE re.UNICODE)</pre>	<p>Ex. 1:</p> <pre>('template.htm', 'template', 'template.lnk', 'template.lnk')</pre> <p>Ex. 2:</p> <pre>('avatar.txt', 'avatar', 'avatar.lnk', 'avatar.lnk')</pre> <p>Ex. 3:</p> <pre>('DSCN0042.JPG', 'DSCN0042', 'DSCN0042.lnk', 'DSCN0042.lnk')</pre> <p>Ex. 4:</p> <pre>('Notes.doc', 'Notes', 'Notes.lnk', 'Notes.lnk')</pre>

Continued on next page...

Table A.1 . . . continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
System auditing	<ul style="list-style-type: none">• Security Events e.g.<ul style="list-style-type: none">– account creation,– login/logout,– password change.• Application Events e.g.<ul style="list-style-type: none">– application installation.• System Events e.g.<ul style="list-style-type: none">– system boot/shutdown,– driver installation.	<p>Parse the event logs and return the results for activities of interest.</p> <p>Example Python implementation:</p> <p>[SNIP]</p> <pre>elif i.endswith("Security.evtx"): evtx_file = i subprocess.run([r"python", r"third_party\bin \ \python_evtx\scripts\evtx_dump.py," \ evtx_file, ">>", security_evtx.xml],\ shell=True) else: continue</pre> <p>[SNIP]</p> <p># Parse the xml output from evtx_dump.py</p> <pre>tree = etr.iterparse(security_evtx.xml, \ remove_pis=True, huge_tree=True, \ remove_blank_text=True, \ resolve_entities=True, recover=True) for event, elem in tree: for child in elem: results = child.tag, child.text # open an xml file using the alias 'xml_out' with open(parsed_results.xml, 'a', \ encoding="utf-8") as xml_out: xml_out.write("{0}{1}{2}{3}{4}".format \ (child.tag, ': ', \ child.text, ' ', '\n'))</pre>	<p>[SNIP]</p> <pre>{http://schemas.microsoft.com/win/2004/08/events/ event}Provider: None {http://schemas.microsoft.com/win/2004/08/events/ event}EventID: 1001 {http://schemas.microsoft.com/win/2004/08/events/ event}Level: 4 [SNIP] {http://schemas.microsoft.com/win/2004/08/events/ event}EventRecordID: 769 {http://schemas.microsoft.com/win/2004/08/events/ event}Channel: Application {http://schemas.microsoft.com/win/2004/08/events/ event}Computer: DESKTOP-XXX {http://schemas.microsoft.com/win/2004/08/events/ event}Data: <string>1505762319964187795</string> <string>5</string> <string>RADAR_PRE_LEAK_WOW64</string> <string>firefox.exe</string> <string>79.0.0.7506</string> <string>10.0.18362.2.0.0</string></pre>

Continued on next page...

Table A.1 ... continued from previous page

Item	Data Categories	Locator Information	Example Output (Extracts)
Application execution	<ul style="list-style-type: none"> • Application name, • Timestamp for first execution, • Number of times the application was executed, • Timestamps for the last 8 times it was executed, • The volume where the application was executed from (path and serial number), • Path of files and directories loaded/accessed by the application. 	<p># Convert the prefetch output into a Python dictionary</p> <p>Use a GUI or command-line tool to parse the Prefetch files.</p> <p>Example Python implementation:</p> <pre>subprocess.call([r"PECmd\PECmd.exe", "-d", \ prefetch_directory, "--json", \ prefetch_output.json], \ shell=True)</pre> <p># open the json file using the alias 'pf_out'</p> <pre>with open(prefetch_output.json) as pf_out: for json_object in pf_out: prefetch_dict = json.loads(json_object) prefetch_dump = json.dumps(prefetch_dict, \ indent=2)</pre> <p># open a file using the alias 'parsed_pf'</p> <pre>with open(parsed_prefetch.json, "a+") \ as parsed_pf: parsed_pf.write(prefetch_dump)</pre>	<pre>"SourceFilename": "C:\WINDOWS\Prefetch\CHROME.EXE-CCF9F3F4.pf" "SourceCreated": "2019-11-05T20:36:33. [SNIP]", "SourceModified": "2020-01-21T17:32:14. [SNIP]", "SourceAccessed": "2020-01-21T17:32:14. [SNIP]", "ExecutableName": "CHROME.EXE", "Hash": "CCF9F3F4", "Size": "397218", "Version": "Windows 10", "RunCount": "116", "LastRun": "2020-01-21T17:32:11. [SNIP]", "PreviousRun0": "2020-01-19T15:16:23. [SNIP]", [SNIP] "PreviousRun6": "2020-01-09T01:20:53. [SNIP]", "Volume0Name": "\VOLUME{01d054405ce90ea5-2e5d8b2c}", "Volume0Serial": "2E5D8B2C", "Directories": "\VOLUME{01d054405ce90ea5-2e5d8b2c}\USERS\XXX\ APPDATA\LOCAL\GOOGLE\CHROME\USER DATA\DEFAULT\ CACHE\DATA_1" [SNIP] "FilesLoaded": " \VOLUME{01d054405ce90ea5-2e5d8b2c}\PROGRAM FILES (X86)\GOOGLE\CHROME\APPLICATION\CHROME.EXE" [SNIP]</pre>

A.2 Sample HTML report

This section contains an example output from SMURF. Figure A.1 shows the main view of the report.

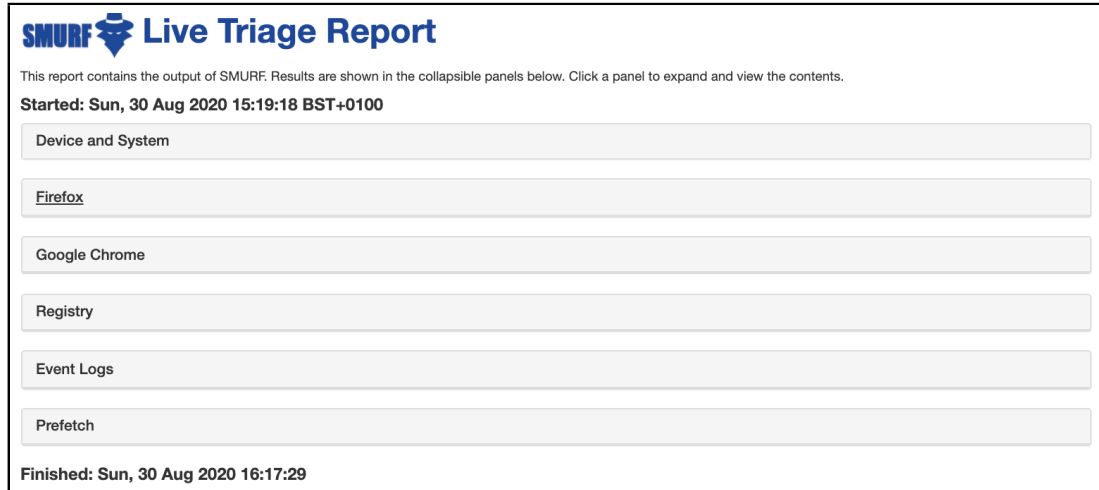


Figure A.1: SMURF - Example HTML report (main view)

Figures A.2 - A.7 show previews from individual sections of the report.

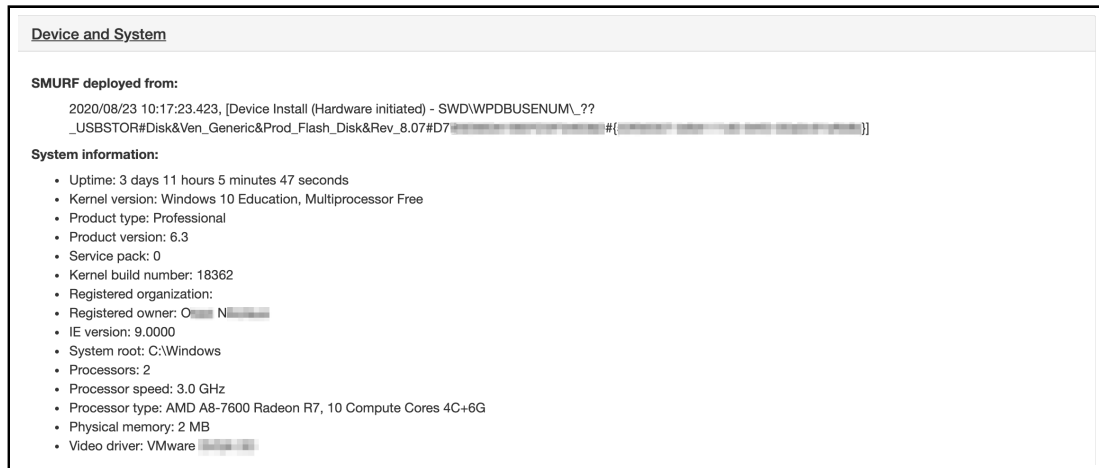


Figure A.2: Deployment device, system information

A.2. SAMPLE HTML REPORT

Firefox
Visited URLs (preview): <ul style="list-style-type: none">2020-08-12 18:11:48, twitter.com, J [REDACTED], with_replies, (11) Tweets with replies by J [REDACTED] S [REDACTED] (@J [REDACTED]) / Twitter2020-08-12 18:12:00, twitter.com, [REDACTED], status, 109 [REDACTED], (11) [REDACTED] on Twitter: "Dilbert comic (February 16) https://t.co/[REDACTED]" / Twitter2020-08-12 18:15:18, twitter.com, [REDACTED], (13) S [REDACTED] P [REDACTED] (@S [REDACTED]) / Twitter2020-08-12 18:18:12, twitter.com, compose, tweet, (12) O [REDACTED] N [REDACTED] on Twitter: "Hilarious. Do you remember this episode? 🤔" / Twitter2020-08-12 18:24:51, twitter.com, messages, (13) Messages / Twitter2020-08-12 18:24:55, twitter.com, messages, 109 [REDACTED], (13) J [REDACTED] S [REDACTED]2020-08-12 18:25:00, twitter.com, messages, 103 [REDACTED]-103 [REDACTED], (14) J [REDACTED] S [REDACTED] / Twitter2020-08-12 18:44:43, twitter.com, N [REDACTED], status, 129 [REDACTED]?ctx=I [REDACTED], (14) O [REDACTED] N [REDACTED] on Twitter: "@J [REDACTED] https://t.co/[REDACTED]" / Twitter2020-08-12 18:45:03, twitter.com, [REDACTED], status, 129 [REDACTED] photo, 1, (14) Home / Twitter2020-08-12 18:52:22, twitter.com, explore, (14) Explore / Twitter2020-08-12 18:54:47, twitter.com, S [REDACTED], following, (14) People followed by S [REDACTED] M [REDACTED] (@s [REDACTED]) / Twitter2020-08-25 18:36:31, twitter.com, settings, contacts, (2) Discoverability and contacts / Twitter2020-08-29 08:49:29, twitter.com, search?q=s [REDACTED]%40gmail.com%20&src=typed_query, s [REDACTED]@gmail.com - Twitter Search / Twitter2020-08-29 08:50:24, twitter.com, search?q=074 [REDACTED]2&src=typed_query, 074 [REDACTED]2 - Twitter Search / Twitter2020-01-30 11:58:43, www.facebook.com, [REDACTED], groups2020-01-30 11:58:52, www.facebook.com, groups, [REDACTED], ?ref=pages_profile_groups_tab&source_id=[REDACTED]2020-01-30 12:00:52, en-gb.facebook.com, pg, [REDACTED], events, ?ref=page_internal2020-01-30 12:00:54, en-gb.facebook.com, [REDACTED], photos2020-01-30 12:01:13, en-gb.facebook.com, groups, [REDACTED], ?ref=pages_profile_groups_tab&source_id=[REDACTED]2020-02-29 07:38:48, www.facebook.com, [REDACTED]2020-03-03 00:00:10, docs.google.com, document, d, 1sQ [REDACTED], edit?rm=demo2020-03-03 00:01:28, drive.google.com, start, apps2020-04-11 12:18:13, docs.google.com

A.3(a): Visited URLs

Sessionstore URLs (preview): <ul style="list-style-type: none">URL: https://twitter.com/home?lang=en; Page Title: (2) Home / Twitter; Referrer: NoneURL: https://twitter.com/S [REDACTED]; Page Title: S [REDACTED] M [REDACTED] (@S [REDACTED]) / Twitter; Referrer: NoneURL: https://twitter.com/N [REDACTED]/followers; Page Title: (2) People following O [REDACTED] N [REDACTED] (@N [REDACTED]) / Twitter; Referrer: NoneURL: https://twitter.com/home?lang=en; Page Title: Home / Twitter; Referrer: NoneURL: https://twitter.com/notifications; Page Title: Notifications / Twitter; Referrer: NoneURL: https://twitter.com/home?lang=en; Page Title: Home / Twitter; Referrer: NoneURL: https://twitter.com/home; Page Title: Home / Twitter; Referrer: NoneURL: https://docs.google.com/document/d/1sQ [REDACTED]/edit; Page Title: Quick - General Profiling - Google Docs; Referrer: None; Last Accessed: 2020-06-20 09:07:07 Formhistory - fieldname, value, timeFirstUsed, timeLastUsed, timesUsed, guid: <ul style="list-style-type: none">session[username_or_email], o [REDACTED]@gmail.com, 2020-08-23 15:05:12, 2020-08-23 15:05:12, 1, [REDACTED]session[username_or_email], o [REDACTED]@gmail.com, 2020-08-12 17:44:21, 2020-08-12 17:44:21, 1, [REDACTED]searchbar-history, mail.google.com, 2020-08-12 17:44:54, 2020-08-12 17:44:54, 1, [REDACTED]searchbar-history, google chrome, 2020-08-22 16:38:16, 2020-08-22 16:38:16, 1, [REDACTED]
--

A.3(b): Sessionstore, formhistory

Cookies - createdTime, lastAccessedTime, expiryTime, baseDomain, name, value: <ul style="list-style-type: none">2018-12-01 18:41:41, 2018-12-01 18:41:41, 1970-01-01 00:30:59, .twitter.com, dnt, 12018-12-01 19:56:20, 2018-12-01 19:58:28, 1970-01-01 00:25:43, .twitter.com, fm, 02018-12-01 18:35:04, 2018-12-01 21:24:41, 1970-01-01 00:26:15, .twitter.com, csrf_same_site_set, 12020-08-23 13:38:23, 2020-08-23 15:04:49, 1970-01-01 00:26:38, .twitter.com, gt, [REDACTED]2018-12-01 18:21:14, 2020-08-30 12:59:31, 1970-01-01 00:26:46, .twitter.com, personalization_id, "v1 [REDACTED]"2018-12-01 18:35:02, 2020-08-30 12:59:31, 1970-01-01 00:29:15, .twitter.com, remember_checked_on, 12020-08-12 19:47:24, 2020-08-12 20:01:09, 1970-01-01 00:26:39, contacts.google.com, OTZ, [REDACTED]2020-08-12 18:08:49, 2020-08-12 20:09:16, 1970-01-01 00:26:39, mail.google.com, _Host [REDACTED], 12020-08-12 18:09:14, 2020-08-22 16:38:22, 1970-01-01 00:26:39, ogs.google.com, OTZ, [REDACTED]2020-08-12 18:08:47, 2020-08-22 16:39:38, 1970-01-01 00:35:46, .google.co.uk, CONSENT, YES+GB.en+2020-08-22 16:39:32, 2020-08-22 16:39:54, 1970-01-01 00:26:38, .google.com, _gid, [REDACTED] Permissions - origin, type, timeModified, expiryTime: <ul style="list-style-type: none">https://twitter.com, storageAccessAPI, 2020-01-30 11:46:36, 2020-02-29 11:46:36https://twitter.com, storageAccessAPI, 2020-08-30 12:06:46, 2020-10-14 12:06:46https://twitter.com, storageAccessAPI, 2020-08-12 19:10:20, 2020-09-26 19:10:20https://accounts.google.com, storageAccessAPI, 2020-08-12 20:00:25, 2020-09-26 20:00:25https://mail.google.com, storageAccessAPI, 2020-08-12 20:01:16, 2020-09-26 20:01:16https://myaccount.google.com, storageAccessAPI, 2020-08-12 20:09:55, 2020-09-26 20:09:55https://www.google.com, storageAccessAPI, 2020-08-22 16:40:08, 2020-10-06 16:40:08https://twitter.com, 3rdPartyStorage^https://twitter.com, 2020-08-29 08:49:48, 2020-08-29 09:04:48https://docs.google.com, storageAccessAPI, 2020-02-29 07:40:03, 2020-03-30 07:40:03https://www.facebook.com, storageAccessAPI, 2020-02-29 07:45:34, 2020-03-30 07:45:34https://en-gb.facebook.com, storageAccessAPI, 2020-02-29 07:47:47, 2020-03-30 07:47:47

A.3(c): Cookies, permissions

A.2. SAMPLE HTML REPORT

Cache data (preview):

- Cache file is written in version: 3
- Cache entry file size: 524
- Cache last modified: Wed, 12 Aug 2020 19:16:37
- Cache expiration: Sun, 07 Feb 2106 06:28:15
- Start of cache metadata: 0
- Cache file fetch count: 3
- Cache file last fetched: Wed, 12 Aug 2020 18:08:00
- Key size: 39
- Key (URL): ~predictor-origin,https://twitter.com/
- Key hash (SHA1 hash of url): 3D30EB268AFA67D0E713C46B9F1D759D1F1BEF85
-
- Parsing file: C:\Users\O N\AppData\Local\Mozilla\Firefox\Profiles\ default-release\cache2\entries\5E1B850D9D7FE6080F1717A1D6A7139104E83E1F
- Cache file is written in version: 3
- Cache entry file size: 27146
- Cache last modified: Wed, 12 Aug 2020 18:08:36
- Cache expiration: Wed, 12 Aug 2020 19:47:04
- Start of cache metadata: 18469
- Cache file fetch count: 0
- Cache file last fetched: Thu, 01 Jan 1970 00:00:00
- Key size: 75
- Key (URL): ~unique:https://twitter.com,https://www.google-analytics.com/analytics.js
- Key hash (SHA1 hash of url): 5E1B850D9D7FE6080F1717A1D6A7139104E83E1F
-
- Parsing file: C:\Users\O N\AppData\Local\Mozilla\Firefox\Profiles\ default-release\cache2\entries\
- Cache file is written in version: 3
- Cache entry file size: 9993
- Cache last modified: Wed, 12 Aug 2020 18:09:08
- Cache expiration: Thu, 01 Jan 1970 00:00:00
- Start of cache metadata: 0
- Cache file fetch count: 1
- Cache file last fetched: Wed, 12 Aug 2020 18:09:05
- Key size: 196
- Key (URL): ~unique:https://twitter.com,https://

A.3(d): Cache data

Figure A.3: Firefox - Example output

Google Chrome

Visited URLs (Chrome - preview):

- 2020-01-21 13:23:48, drive.google.com, drive, my-drive
- 2020-01-21 13:23:54, drive.google.com, drive, folders, 0
- 2020-01-21 13:23:58, drive.google.com, drive, folders, 1Y
- 2020-01-21 13:24:11, docs.google.com, spreadsheets, d, 1K
- 2020-01-27 17:40:47, twitter.com, search?q=apple%20cake&src=typed_query
- 2020-01-30 16:59:40, drive.google.com, drive, folders, 1
- 2020-02-10 16:56:54, docs.google.com, document, create?usp=drive_web&oid=1
- 2020-08-23 19:05:40, twitter.com, search?q=good&src=typed
- 2020-08-23 19:06:12, twitter.com, P, J, status, 6
- 2020-08-23 19:06:22, twitter.com, , status, 1
- 2020-08-23 19:06:22, twitter.com, search?q=f&src=typed
- 2020-08-23 19:06:24, twitter.com, search?q=iam&src=typed
- 2020-08-23 19:06:31, twitter.com, iam, status, 102
- 2020-08-23 19:06:31, twitter.com, N, id
- 2020-08-23 19:08:59, twitter.com, explore
- 2020-08-23 19:09:00, twitter.com, home

URLs in current session (preview):

- UpdateTabNavigation - Tab: 314, Index: 1, URL: https://twitter.com/login, Title: None, State: None, Transition: 0
- UpdateTabNavigation - Tab: 316, Index: 0, URL: https://twitter.com/explore, Title: , State: None, Transition: 0
- UpdateTabNavigation - Tab: 322, Index: 0, URL: https://twitter.com/S /media, Title: , State: None, Transition: 0
- UpdateTabNavigation - Tab: 323, Index: 0, URL: https://twitter.com/P J/with_replies, Title: , State: None, Transition: 1
- UpdateTabNavigation - Tab: 324, Index: 0, URL: https://twitter.com/P J/status/665, Title: , State: None, Transition: 0
- UpdateTabNavigation - Tab: 325, Index: 0, URL: https://twitter.com/P J, Title: , State: None, Transition: 136
- UpdateTabNavigation - Tab: 327, Index: 0, URL: https://twitter.com/P /status/102, Title: , State: None, Transition: 0
- UpdateTabNavigation - Tab: 329, Index: 0, URL: https://twitter.com/t, Title: , State: None, Transition: 120
- UpdateTabNavigation - Tab: 334, Index: 0, URL: https://twitter.com/search?q=good&src=typed, Title: , State: None, Transition: 5
- UpdateTabNavigation - Tab: 335, Index: 0, URL: https://twitter.com/N d, Title: O s (@N d) / Twitter, State: None, Transition: 0
- UpdateTabNavigation - Tab: 336, Index: 0, URL: https://www.google.com/intl/en-GB/gmail/about/#, Title: None, State: None, Transition: 0
- UpdateTabNavigation - Tab: 337, Index: 0, URL: https://www.google.com/intl/en-GB/gmail/about/#, Title: None, State: None, Transition: 0

Tabs in current session (preview):

- SelectedNavigationInTab - Tab: 7346, Index: 6, Time: 132
- UpdateTabNavigation - Tab: 7346, Index: 4, Url: https://drive.google.com/drive/folders/1aR
- UpdateTabNavigation - Tab: 7346, Index: 5, Url: https://drive.google.com/drive/folders/1Y
- UpdateTabNavigation - Tab: 7346, Index: 9, Url: https://drive.google.com/drive/folders/1aR
- UpdateTabNavigation - Tab: 7346, Index: 10, Url: https://drive.google.com/drive/folders/1FN
- SelectedNavigationInTab - Tab: 8087, Index: 6, Time: 132
- UpdateTabNavigation - Tab: 8087, Index: 9, Url: https://www.facebook.com/
- UpdateTabNavigation - Tab: 8087, Index: 10, Url: https://www.facebook.com/?

A.4(a): Visited URLs, current session data, formhistory

A.2. SAMPLE HTML REPORT

URLs in last session (preview):

- UpdateTabNavigation - Tab: 241, Index: 0, URL: chrome://newtab/, Title: New Tab, State: None, Transition: 0
- UpdateTabNavigation - Tab: 241, Index: 1, URL: chrome://history/, Title: None, State: None, Transition: 0
- UpdateTabNavigation - Tab: 245, Index: 0, URL: https://www.google.com/intl/en-GB/gmail/about/#, Title: None, State: None, Transition: 0
- UpdateTabNavigation - Tab: 246, Index: 0, URL: https://www.google.com/intl/en-GB/gmail/about/#, Title: None, State: None, Transition: 0
- UpdateTabNavigation - Tab: 247, Index: 0, URL: https://twitter.com/?logged_out=1&lang=en, Title: None, State: None, Transition: 0
- UpdateTabNavigation - Tab: 247, Index: 1, URL: https://twitter.com/login, Title: None, State: None, Transition: 0
- UpdateTabNavigation - Tab: 247, Index: 2, URL: https://twitter.com/home?logged_out=1&lang=en, Title: None, State: None, Transition: 0
- UpdateTabNavigation - Tab: 247, Index: 3, URL: https://twitter.com/home?lang=en, Title: Home / Twitter, State: None, Transition: 176
- UpdateTabNavigation - Tab: 249, Index: 0, URL: https://twitter.com/explore, Title: , State: None, Transition: 0
- UpdateTabNavigation - Tab: 254, Index: 0, URL: https://twitter.com/ /lists, Title: , State: None, Transition: 0
- UpdateTabNavigation - Tab: 260, Index: 0, URL: https://twitter.com/P /status/102 , Title: , State: None, Transition: 0
- UpdateTabNavigation - Tab: 268, Index: 0, URL: https://twitter.com/N d, Title: O s (@N d) / Twitter, State: None, Transition: 0
- UpdateTabNavigation - Tab: 254, Index: 0, URL: https://twitter.com/ /lists, Title: , State: None, Transition: 0
- UpdateTabNavigation - Tab: 255, Index: 0, URL: https://twitter.com/ /media, Title: , State: None, Transition: 0
- UpdateTabNavigation - Tab: 269, Index: 0, URL: https://www.google.com/intl/en-GB/gmail/about/#, Title: None, State: None, Transition: 0

Tabs in last session (preview):

- SelectedNavigationInTab - Tab: 4456, Index: 2, Time: 132
- UpdateTabNavigation - Tab: 4456, Index: 0, Url: chrome://newtab/
- SelectedNavigationInTab - Tab: 7346, Index: 6, Time: 132
- UpdateTabNavigation - Tab: 7346, Index: 4, Url: https://drive.google.com/drive/folders/1aR
- UpdateTabNavigation - Tab: 7346, Index: 5, Url: https://drive.google.com/drive/folders/1Yu
- UpdateTabNavigation - Tab: 7346, Index: 10, Url: https://drive.google.com/drive/folders/1FN
- SelectedNavigationInTab - Tab: 7348, Index: 5, Time: 132
- UpdateTabNavigation - Tab: 7348, Index: 0, Url: chrome://newtab/
- UpdateTabNavigation - Tab: 7348, Index: 1, Url: https://drive.google.com/drive/my-drive
- UpdateTabNavigation - Tab: 7348, Index: 2, Url: https://drive.google.com/drive/folders/0B9
- UpdateTabNavigation - Tab: 7348, Index: 7, Url: https://drive.google.com/drive/folders/1w5
- SelectedNavigationInTab - Tab: 8087, Index: 6, Time: 132
- UpdateTabNavigation - Tab: 8087, Index: 9, Url: https://www.facebook.com/
- UpdateTabNavigation - Tab: 8087, Index: 10, Url: https://www.facebook.com/?
- UpdateTabNavigation - Tab: 8087, Index: 11, Url: https://www.facebook.com/?

Chrome downloads:

- 2015-10-25 15H:43M:23S C:\Users\ Downloads\ .PDF
- 2015-11-15 13H:27M:27S C:\Users\ Downloads\ .exe
- 2016-02-08 10H:31M:04S E\ .txt
- 2016-02-08 12H:00M:41S E\ .zip
- 2018-08-04 13H:18M:09S https://pexels.com/ C:\Users\ Downloads\ .jpeg
- 2018-08-04 13H:19M:13S https://pexels.com/ C:\Users\ Downloads\ .jpeg
- 2018-08-04 13H:19M:59S https://pexels.com/ C:\Users\ Downloads\ .jpeg
- 2019-06-08 15H:46M:43S C:\Users\ Downloads\ .xlsx
- 2019-06-08 16H:27M:35S C:\Users\ Documents\ .zip
- 2020-01-05 22H:44M:34S C:\Users\ Desktop\ .tar.gz
- 2020-01-06 00H:48M:42S C:\Users\ Downloads\ .

A.4(b): Cookies, permissions

Cache data:

```
{
  "source_short": "WEBHIST",
  "source_long": "Chrome Cache",
  "parser": "hindsight/2.4.0",
  "profile": "C:\\Users\\ \\AppData\\Local\\Google\\Chrome\\User Data\\Default",
  "url": "https://pbs.twimg.com/profile_images/11 /76/ _pic_bigger.jpg",
  "value": "",
  "hash": "",
  "next": 0,
  "rankingNode": "",
  "usageCounter": 0,
  "reuseCounter": 0,
  "state": 0,
  "creationTime": "2018-08-19T11:27:08.013887+00:00",
  "keyLength": 80,
  "keyAddress": 0,
  "data": [
    null,
    null
  ],
  "flags": 0,
  "key": "https://pbs.twimg.com/profile_images/11 /76/ _pic_bigger.jpg",
  "etag": "",
  "server_name": "",
  "last_modified": "",
  "file_size": 4991,
  "location": "data_2 [3719168]; data_2 [3723264]",
  "http_headers_str": "",
  "datetime": "2018-08-19T11:27:08.013887+00:00",
  "timestamp_desc": "Last Visit Time",
  "data_type": "chrome:cache:entry",
  "original_url": "https://pbs.twimg.com/profile_images/11 /76/ _pic_bigger.jpg",
  "cache_type": "cache",
  "cached_state": "Normal (data cached)",
  "message": "Original URL: https://pbs.twimg.com/profile_images/11 /76/ _pic_bigger.jpg"
}
```

A.4(c): Cache data

Figure A.4: Google Chrome - Example output

A.2. SAMPLE HTML REPORT

Registry

Users logged on locally:

- 27/08/2020 03:16:49 [redacted] N
-

A.5(a): Logged on user(s)

```
Active logon sessions:
• [0] Logon session 00000000:000003e7:
• Auth package: NTLM
• Session: 0
• Logon time: 27/08/2020 03:12:36
• DNS Domain:
•
• User name:
• Logon type: (none)
• Sid: (none)
• Logon server:
• UPN:
• [2] Logon session 00000000:0000b9a2:
• Auth package: Negotiate
• Session: 1
• Logon time: 27/08/2020 03:12:37
• DNS Domain:
•
• User name: Font Driver Host\UMFD-0
• Logon type: Interactive
• Sid: S-1-5-███
• Logon server:
• UPN:
• [4] Logon session 00000000:000003e4:
• Auth package: Negotiate
• Session: 0
• Logon time: 27/08/2020 03:12:37
• DNS Domain:
•
• User name: Window Manager\DWM-1
• Logon type: Interactive
• Sid: S-1-5-███
• Logon server:
• UPN:
• [6] Logon session 00000000:00011293:
• Auth package: Negotiate
• Session: 1
• Logon time: 27/08/2020 03:12:38
• DNS Domain:
•
• User name: NT AUTHORITY\LOCAL SERVICE
• Logon type: Service
• Sid: S-1-5-███
• Logon server:
• UPN:
• [8] Logon session 00000000:000f9571:
• Auth package: NTLM
• Session: 1
• Logon time: 27/08/2020 03:16:48
• DNS Domain:
•
• User name: ██████████\O ████████ N ████████
• Logon type: Interactive
• Sid: S-1-5-████████████████████████████████████████-1001
• Logon server: ██████████-██████████
• UPN:
```

A.5(b): Active logon session

A.2. SAMPLE HTML REPORT

SAM:

- ProfileImagePath: S-1-5-█-█-█-1001
- Path to SAM hive: \SystemRoot\System32\Config\SAM
- Hive last reorganised timestamp [FILETIME (UTC)]: 2020-08-22 11H:39M:03S

CURRENT USER - Recent Docs (preview):

- [(['netpredict.csv', 'netpredict', 'netpredict.Ink', 'netpredict.Ink'])]
- [(['yawn.mp4', 'yawn', 'yawn.Ink', 'yawn.Ink'])]
- [(['README.txt', 'README', 'README.Ink', 'README.Ink'])]
- [(['top_level.txt', 'top_level', 'top_level.Ink', 'top_level.Ink'])]
- [(['imports2impo', 'imports2', 's.Ink', 'imports.Ink'])]
- [(['DSCN0042.JPG', 'DSCN0042', 'DSCN0042.Ink', 'DSCN0042.Ink'])]
- [(['Programsf2Progr', 'Programsf2P', 's.Ink', 'Programs.Ink'])]
- [(['settingsf2setti', 'settingsf2s', 's.Ink', 'settings.Ink'])]
- [(['new_template.html', 'new_template', '2new_template.Ink', 'new_template.Ink'])]
- [(['visio_diagrams.vsd', 'visio_diagrams', '2visio_diagrams.Ink', 'visio_diagrams.Ink'])]
- [(['100NI', '1', 'N12100NIKON.Ink', '100NIKON.Ink'])]
- [(['JSON.zip', 'JSON', 'JSON.Ink', 'JSON.Ink'])]
- [(['en_GB.dic', 'en_GB', 'en_GB.Ink', 'en_GB.Ink'])]
- [(['Example1.iss', 'Example1', 'Example1.Ink', 'Example1.Ink'])]
- [(['Highlights.doc', 'Highlights', '2Highlights.Ink', 'Highlights.Ink'])]
- [(['taxonomy.pdf', 'taxonomy', 'taxonomy.Ink', 'taxonomy.Ink'])]

SECURITY:

- Path to SECURITY hive: \emRoot\System32\Config\SECURITY
- Hive last reorganised timestamp [FILETIME (UTC)]: 2020-08-22 11H:39M:02S

Auditing (preview):

- (Security) Get audit policy from the Security hive file
-
- auditpol
- Policy\PolAdtEv
- LastWrite Time Thu Jan 1 00:00:00 1970 (UTC)
-
- Possible Win10(1607+)/Win2016
- System:Security State Change S
- System:Security System Extension N
- System:System Integrity S/F
- System:IPsec Driver N
- System:Other System Events S/F
- Logon/Logoff:Logon S/F
- Logon/Logoff:Logoff S
- Logon/Logoff:Account Lockout S
- Logon/Logoff:IPsec Main Mode N
- Logon/Logoff:Special Logon S
- Logon/Logoff:IPsec Quick Mode N
- Logon/Logoff:IPsec Extended Mode N
- Logon/Logoff:Other Logon/Logoff Events N
- Logon/Logoff:Network Policy Server S/F
- Logon/Logoff:User/Device Claims N
- Logon/Logoff:Group Membership N
- Object Access:File System N
- Object Access:Registry N
- Object Access:Kernel Object N
- Object Access:SAM N

A.5(c): SAM, Current User, Security (key, value, data)

SYSTEM:

Mounted devices (preview):

- _??_ USBSTOR#Disk&Ven_SanDisk&Prod_Cruzer_Blade&Rev_8.02# _____#{_____}
- _??_ USBSTOR#Disk&Ven_Generic&Prod_Flash_Disk&Rev_8.07# _____#{_____}
- _USBSTOR#Disk&Ven_Kingston&Prod_DataTraveler_3.0&Rev_PMAP# _____#{_____}

List of System Keys (preview):

- System\CurrentControlSet\Control\ComputerName\ActiveComputerName, Default, _____, REG_SZ
- System\CurrentControlSet\Control\ComputerName\ComputerName, Default, mnmsrv, REG_SZ
- System\CurrentControlSet\Control\Session Manager\Memory Management, Default, [?:\pagefile.sys], REG_MULTI_SZ
- System\MountedDevices, Default, b'VeraCryptVolumeP', REG_BINARY

[System Keys](#)

SOFTWARE (preview):

- Software\Microsoft\Windows\CurrentVersion\Authentication\LogonUI, Default, 0, REG_DWORD
- Software\Microsoft\Windows NT\CurrentVersion\WinLogon, Default, , REG_SZ
- Software\Microsoft\Windows NT\CurrentVersion\WinLogon, Default, {_____}, REG_SZ
- Software\Microsoft\Windows NT\CurrentVersion\WinLogon, Default, explorer.exe, REG_SZ
- Software\Microsoft\Windows NT\CurrentVersion\WinLogon, Default, 0, REG_DWORD

ROOT KEYS:

- Subkey: http\DefaultIcon
- Value Name: Default
- Value: C:\Program Files (x86)\Google\Chrome\Application\chrome.exe,0
- Value Data Type: REG_SZ
-
- Subkey: https\DefaultIcon
- Value Name: Default
- Value: C:\Program Files (x86)\Google\Chrome\Application\chrome.exe,0
- Value Data Type: REG_SZ

A.5(d): System, Software, Root (key, value, data)

Figure A.5: Windows Registry - Example Output

A.2. SAMPLE HTML REPORT

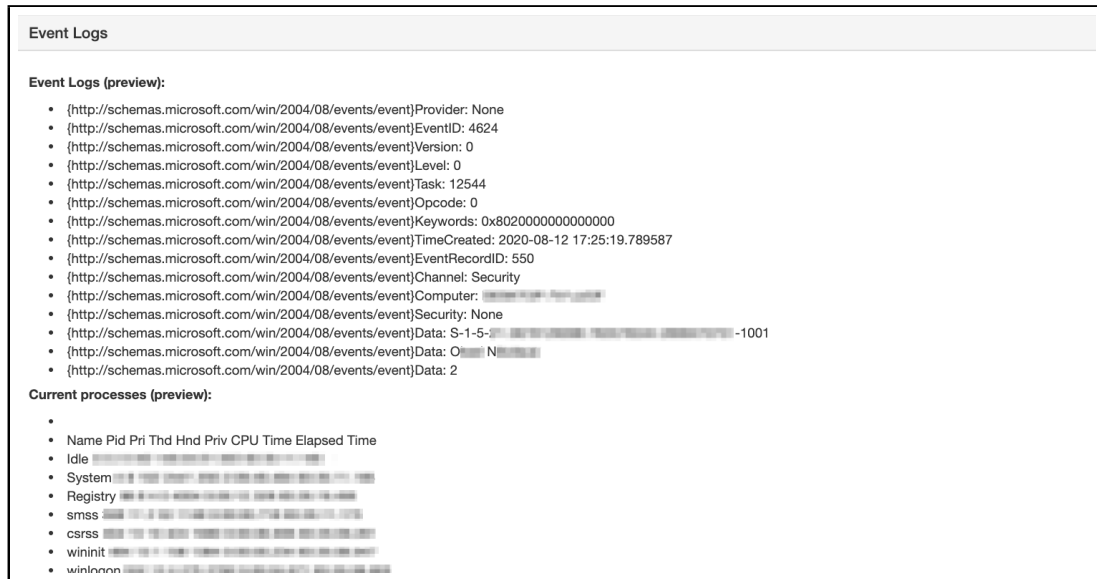


Figure A.6: Event logs - Example output



Figure A.7: Prefetch - Example output

Appendix B: Context Artefacts

B.1 Examples from Registry

Hives path:

`%SYSTEMROOT%\System32\config`

NTUSER.DAT path:

`%SYSTEMROOT%\Users\[Username]\`

UsrClass.dat path:

`%SYSTEMROOT%\Users\[Username]\AppData\Local\Microsoft\Windows`

B.1.1 NTUSER.DAT

NTUSER.DAT is a user-specific registry hive. It contains user-specific configuration for a Windows computer. It can be found in the user's home directory. Where multiple user accounts exist on a Windows computer, separate NTUSER.DAT files will exist in each user's home directory. Information contained in NTUSER.DAT includes applications installed for the user, paths to files, etc.

To provide context for the web app artefacts and subsequent features extracted, artefacts from NTUSER.DAT can be used to identify installed programs/applications, default web browser, mapped network share for file upload or downloads, recently accessed documents etc.

B.1.2 UsrClass.dat

UsrClass.dat is another user specific hive located in:

`%SYSTEMROOT%\Users\[Username]\AppData\Local\Microsoft\Windows.`

It contains user-specific configuration for access controlled processes or applications that cannot be written to the Windows registry hives. In order to recover user-specific activity such as most recently used file/application and accessed folders; artefacts from NTUSER.dat and UsrClass.dat need to be analysed for each user account on a computer.

Extracts from RegRipper shows the path to photos from the mapped network drive [Section 4.3.2]; the filenames of images used as twitter profile images:

```
Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32
OpenSavePidlMRU\jpg
LastWrite Time: Sun Aug 19 09:02:17 2018
Note: All value names are listed in MRUListEx order.
```

```
[...]
```

```
My Computer\Z:\shared_folders_josh\pics\beautiful-bloom-blooming-268432.jpg
My Computer\Z:\shared_folders_josh\pics\alaska-forest-hd-wallpaper-792.jpg
My Computer\Z:\shared_folders_josh\pics\animal-bird-feathers-36762.jpg
```

B.2 Procmon

Procmon (Process Monitor) is a free tool from Microsoft Sysinternals [Russovich, 2018] It shows real-time filesystem activity. This includes but is not limited to the creation, opening and querying of registry keys; read/write activities on files and directories, the creation of OS Event logs and drivers loaded/dll files accessed.

Procmon captures both system generated (filesystem, registry, network connectivity) and user generated (installing applications such as Firefox; Chrome) Input/Output operations. In the context of this research, relationships can be established by understanding the sequence of events that led to the creation of an artefact. For example, the user installed a web browser - prefetch shows the path to the executable installer, Event Logs show when the application was installed, Registry keys were created for the browser.

Procmon logs in this example can be useful when trying to understand the sequence of events i.e. user starts a browsing session, the sessionstore file is created, updated, tagged and eventually closed at the end of the session as seen in Figure B.1. This provides context as to how recovered files came to be; and may also include recovered fragments of the same file.

An extract from Procmon depicting registry operations during user activity.

Process Name	PID	Sequence	Operation	Path	Result	Detail
svchost.exe	648	48048	RegOpenKey	HKLM\System\CurrentControlSet\Enum\USBSTOR\DISK&VEN_&PROD_FREECOM_DATABAR&REV_5.00\07B71A07CD4...	SUCCESS	Desired Access: Query Value
Drvinst.exe	3008	48049	CreateFile	C:\Windows\System32\DriverStore\infstor.dat	SUCCESS	Desired Access: Generic Read, Disposition: Open, Options: Synchronous ...
Drvinst.exe	3008	48050	QueryStandardInformationFile	C:\Windows\System32\DriverStore\infstor.dat	SUCCESS	AllocationSize: 143,360, EndOfFile: 143,360, NumberOfLinks: 1, DeletePen...
Drvinst.exe	3008	48051	CreateFileMapping	C:\Windows\System32\DriverStore\infstor.dat	FILE LOCKED WITH ONLY REA...	SyncType: SyncTypeCreateSection, PageProtection: PAGE_READONLY
Drvinst.exe	3008	48052	QueryStandardInformationFile	C:\Windows\System32\DriverStore\infstor.dat	SUCCESS	AllocationSize: 143,360, EndOfFile: 143,360, NumberOfLinks: 1, DeletePen...
Drvinst.exe	3008	48053	CreateFileMapping	C:\Windows\System32\DriverStore\infstor.dat	SUCCESS	SyncType: SyncTypeOther
svchost.exe	648	48054	RegCloseKey	HKLM\System\CurrentControlSet\Enum\USBSTOR\Disk&Ven_&Prod_Freecom_Databar&Rev_5.00\07B71A07CD4...	SUCCESS	
Drvinst.exe	3008	48055	CloseFile	C:\Windows\System32\DriverStore\infstor.dat	SUCCESS	
svchost.exe	648	48056	RegOpenKey	HKLM\System\CurrentControlSet\Enum\USBSTOR\DISK&VEN_&PROD_FREECOM_DATABAR&REV_5.00\07B71A07CD4...	SUCCESS	Desired Access: Read
Drvinst.exe	3008	48057	CloseFile	C:\Windows\System32\DriverStore\infpub.dat	SUCCESS	
svchost.exe	648	48058	RegQueryValue	HKLM\System\CurrentControlSet\Enum\USBSTOR\Disk&Ven_&Prod_Freecom_Databar&Rev_5.00\07B71A07CD4...	NAME NOT FOUND	Length: 144
svchost.exe	648	48059	RegCloseKey	HKLM\System\CurrentControlSet\Enum\USBSTOR\Disk&Ven_&Prod_Freecom_Databar&Rev_5.00\07B71A07CD4...	SUCCESS	
svchost.exe	648	48060	RegOpenKey	HKLM\System\CurrentControlSet\Enum\USBSTOR\DISK&VEN_&PROD_FREECOM_DATABAR&REV_5.00\07B71A07CD4...	SUCCESS	Desired Access: Read
svchost.exe	648	48061	RegQueryValue	HKLM\System\CurrentControlSet\Enum\USBSTOR\Disk&Ven_&Prod_Freecom_Databar&Rev_5.00\07B71A07CD4...	NAME NOT FOUND	Length: 144
svchost.exe	648	48062	RegCloseKey	HKLM\System\CurrentControlSet\Enum\USBSTOR\Disk&Ven_&Prod_Freecom_Databar&Rev_5.00\07B71A07CD4...	SUCCESS	
svchost.exe	648	48063	RegOpenKey	HKLM\System\CurrentControlSet\Enum\USBSTOR\DISK&VEN_&PROD_FREECOM_DATABAR&REV_5.00\07B71A07CD4...	SUCCESS	Desired Access: Read
svchost.exe	648	48064	RegQueryValue	HKLM\System\CurrentControlSet\Enum\USBSTOR\Disk&Ven_&Prod_Freecom_Databar&Rev_5.00\07B71A07CD4...	SUCCESS	Type: REG_SZ, Length: 54, Data: Freecom Databar USB Device
svchost.exe	648	48065	RegCloseKey	HKLM\System\CurrentControlSet\Enum\USBSTOR\Disk&Ven_&Prod_Freecom_Databar&Rev_5.00\07B71A07CD4...	SUCCESS	
Drvinst.exe	3008	48066	CreateFile	C:\Windows\inf\setupapi.dev.log	SUCCESS	Desired Access: Generic Read/Write, Disposition: OpenIf, Options: Synchr...
Drvinst.exe	3008	48067	QueryStandardInformationFile	C:\Windows\inf\setupapi.dev.log	SUCCESS	AllocationSize: 4,055,040, EndOfFile: 4,051,089, NumberOfLinks: 1, DeleteP...
Drvinst.exe	3008	48068	CreateFileMapping	C:\Windows\inf\setupapi.dev.log	FILE LOCKED WITH WRITERS	SyncType: SyncTypeCreateSection, PageProtection: PAGE_READWRITE
Drvinst.exe	3008	48069	QueryStandardInformationFile	C:\Windows\inf\setupapi.dev.log	SUCCESS	AllocationSize: 4,055,040, EndOfFile: 4,051,089, NumberOfLinks: 1, DeleteP...
Drvinst.exe	3008	48070	QueryStandardInformationFile	C:\Windows\inf\setupapi.dev.log	SUCCESS	AllocationSize: 4,055,040, EndOfFile: 4,051,089, NumberOfLinks: 1, DeleteP...
Drvinst.exe	3008	48071	SetEndOfFileInformationFile	C:\Windows\inf\setupapi.dev.log	SUCCESS	EndOfFile: 4,051,227
SearchIndexe...	2784	48072	FileSystemControl	C:	SUCCESS	Control: FSCTL_QUERY_USN_JOURNAL
SearchIndexe...	2784	48073	FileSystemControl	C:	SUCCESS	Control: FSCTL_READ_USN_JOURNAL
SearchIndexe...	2784	48074	FileSystemControl	C:	SUCCESS	Control: FSCTL_READ_USN_JOURNAL
Drvinst.exe	3008	48075	CreateFileMapping	C:\Windows\inf\setupapi.dev.log	SUCCESS	SyncType: SyncTypeOther
Drvinst.exe	3008	48076	SetEndOfFileInformationFile	C:\Windows\inf\setupapi.dev.log	SUCCESS	EndOfFile: 4,051,156
Drvinst.exe	3008	48077	SetAllocationInformationFile	C:\Windows\inf\setupapi.dev.log	SUCCESS	AllocationSize: 4,051,156
Drvinst.exe	3008	48078	CloseFile	C:\Windows\inf\setupapi.dev.log	SUCCESS	

Figure B.1: Device driver installation when a USB is plugged in

B.3 Event Logs

Table B.1: Example: Event IDs of interest

Event ID	Description	Log Name
4624	Successful Logon	Security
4625	Failed Login	Security
4776	Successful or Failed Account Authentication	Security
4720	A user account was created	Security

B.4 Example: SQLite statements for Firefox (Chapter 4)

B.4.1 Favicons

```

1 SELECT datetime(page_url_hash/1000000, "unixepoch") as pageURLHash, id, page_url,
2 page_url_hash
3 FROM moz_pages_w_icons
4 WHERE moz_pages_w_icons.page_url LIKE "%twitter%"
5 OR moz_pages_w_icons.page_url LIKE "%google%" OR
6 moz_pages_w_icons.page_url LIKE "%facebook%"
7 ORDER BY page_url_hash

```

B.4.2 Formhistory

```

1 SELECT fieldname, value, datetime(firstUsed/1000000, "unixepoch")
2 as timeFirstUsed, datetime(lastUsed/1000000, "unixepoch") as timeLastUsed,
3 timesUsed, guid
4 FROM moz_formhistory
5 WHERE firstUsed != ""
6 ORDER BY firstUsed

```

B.4.3 Permissions.sqlite

```

1 SELECT id, origin, type, datetime(modificationTime/1000, "unixepoch") as
2 timeModified, datetime(expireTime/1000, "unixepoch") as expiryTime
3 FROM moz_perms
4 ORDER BY modificationTime

```

B.4.4 Parse typed URLs in Firefox

```
1 SELECT moz_inpuhistory.place_id, moz_places.typed, moz_inpuhistory.input,  
2 moz_places.url, moz_places.visit_count, moz_inpuhistory.use_count  
3 FROM moz_places, moz_inpuhistory  
4 WHERE moz_places.id = moz_inpuhistory.place_id;
```

B.5 Example: SQLite statements for Chrome (Chapter 5)

B.5.1 Cookies

```
1 SELECT datetime((creation_utc/1000000) + strftime('%s', '1601-01-01 00:00:00'),  
2 'unixepoch') as createTime, datetime((last_access_utc/1000000) + strftime('%s',  
3 '1601-01-01 00:00:00'), 'unixepoch') as lastAccessed, host_key, name, value,  
4 path, is_persistent  
5 FROM cookies  
6 ORDER BY last_access_utc
```

```
1 # filter for facebook cookies  
2  
3 SELECT datetime((creation_utc/1000000) + strftime('%s', '1601-01-01 00:00:00'),  
4 'unixepoch') as createTime, datetime((last_access_utc/1000000) + strftime('%s',  
5 '1601-01-01 00:00:00'), 'unixepoch') as lastAccessed, host_key, name, value,  
6 path, is_persistent  
7 FROM cookies  
8 WHERE host_key like "%facebook%"  
9 ORDER BY last_access_utc
```

B.5.2 History SQLite database

B.5.2.1 Search term analysis

```
1 SELECT datetime((visits.visit_time/1000000) + strftime('%s',  
2 '1601-01-01 00:00:00'), 'unixepoch') as visitTime, keyword_search_terms.term,  
3 urls.url  
4 FROM keyword_search_terms, urls, visits  
5 WHERE keyword_search_terms.url_id = urls.id = visits.url  
6 ORDER BY visits.visit_time asc
```

B.5.2.2 Sort history by timeline

```

1 SELECT datetime((visits.visit_time/1000000) + strftime('%s',
2 '1601-01-01 00:00:00'), 'unixepoch') as visitTime, urls.url,urls.title
3 FROM urls, visits
4 WHERE visits.url = urls.id
5 ORDER BY visits.visit_time asc

```

B.5.2.3 URLs visited

```

1 SELECT datetime((last_visit_time/1000000) + strftime('%s',
2 '1601-01-01 00:00:00'), 'unixepoch') as lastVisited, url, title, typed_count,
3 visit_count
4 FROM urls

```

B.5.2.4 Login data

```

1 SELECT datetime((date_created/1000000) + strftime('%s', '1601-01-01 00:00:00'),
2 'unixepoch') as createDate, origin_url, action_url, username_element,
3 username_value, password_element, password_value, submit_element, signon_realm,
4 preferred, blacklisted_by_user, password_type, times_used, form_data,
5 date_synced, possible_username_pairs
6 FROM logins
7 ORDER BY date_created asc

```

B.5.2.5 Shortcuts

```

1 SELECT datetime((last_access_time/1000000) + strftime('%s',
2 '1601-01-01 00:00:00'), 'unixepoch') as lastAccessed, id, text, fill_into_edit,
3 url, contents, description, keyword, number_of_hits
4 FROM omni_box_shortcuts
5 ORDER BY last_access_time asc

```

B.5.2.6 Web Data

```

1 SELECT name, value, value_lower, datetime(date_created, "unixepoch") as
2 createDate, datetime(date_last_used, "unixepoch") as lastUsedDate
3 FROM autofill
4 ORDER BY date_last_used asc

```


B.6. EXAMPLE: CACHE ENTRY VALIDATION

B.5.2.7 Keywords

```
1 SELECT datetime((date_created/1000000) + strftime('%s', '1601-01-01 00:00:00'),
2 'unixepoch') as createDate, datetime((last_visited/1000000) + strftime('%s',
3 '1601-01-01 00:00:00'), 'unixepoch') as lastVisited,
4 datetime((last_modified/1000000) + strftime('%s', '1601-01-01 00:00:00'),
5 'unixepoch') as lastModified, usage_count, id, short_name, keyword, favicon_url,
6 url, originating_url
7 FROM keywords
8 ORDER BY last_visited asc
```

B.6 Example: Cache entry validation

An extract of keyword search result for “pbs.twimg” is shown below.

00002224	3A 68 74 74 70 73 3A 2F 2F 70 62 73 2E 74 77	:https://pbs.tw
00002240	69 6D 67 2E 63 6F 6D 2F 70 72 6F 66 69 6C 65 5F	img.com/profile_
00002256	69 6D 61 67 65 73 2F 31 30 33 31 31 33 39 36 37	images/103113967
00002272	32 31 37 35 35 39 39 36 31 36 2F 39 6E 6B 74 65	2175599616/9nkte
00002288	4E 4B 62 5F 6E 6F 72 6D 61 6C 2E 6A 70 67	NKb_normal.jpg

To identify and verify the cache entry filename, i.e. SHA1 hash of the URL; the *sha1sum* for URL(s) returned in the keyword search is calculated:



```
Dee@Dee MINGW64 ~
$ echo -n "https://pbs.twimg.com/profile_images/103..._normal.jpg" | sha1sum | tr [:lower:] [:upper:]
42D850C6B261A524943DCEE9A9B6CE1C1E07E9D8 *-
Dee@Dee MINGW64 ~
$ echo -n ":https://pbs.twimg.com/profile_images/103..._normal.jpg" | sha1sum | tr [:lower:] [:upper:]
AC403D157C8D0F09176C1FC3A7EC035E6D1DC17E
```

Figure B.2: Verification of Firefox Cache entry filename

B.6.1 Cache entry filename

“AC403D157C8D0F09176C1FC3A7EC035E6D1DC17E” location:

00003616	AC 40 3D 15	7C 8D 0F 09 17 6C 1F C3
00003632	A7 EC 03 5E 6D 1D C1 7E	

Appendix C: Terms of Acceptable Use

This Appendix contains extracts from the applicable terms of service, and acceptable use policies for the social media platforms selected for this research.

C.1 Facebook

To read the policies in full, see Facebook Community Standards [Facebook, 2020a].

Part I. Violence and criminal behaviour

Section 1: Violence and incitement

“We aim to prevent potential offline harm that may be related to content on Facebook. While we understand that people commonly express disdain or disagreement by threatening or calling for violence in non-serious ways, we remove language that incites or facilitates serious violence. We remove content, disable accounts and work with law enforcement when we believe that there is a genuine risk of physical harm or direct threats to public safety. We also try to consider the language and context in order to distinguish casual statements from content that constitutes a credible threat to public or personal safety. In determining whether a threat is credible, we may also consider additional information such as a person’s public visibility and the risks to their physical safety”.

Compliance:

No content used in this research constitutes a threat to public safety.

Section 2: Dangerous individuals and organisations

This policy was considered however, it was not applicable because the research was not conducted by or on behalf of an organisation or individual with a violent disposition.

Compliance:

No content used in this research portrays violence; promotes unlawful/dangerous organisations, individuals, or any acts committed by them.

C.1. FACEBOOK

Section 3: Coordinating harm and publicising crime

“In an effort to prevent and disrupt offline harm and copycat behaviour, we prohibit people from facilitating, organising, promoting or admitting to certain criminal or harmful activities targeted at people, businesses, property or animals. We allow people to debate and advocate for the legality of criminal and harmful activities, as well as draw attention to harmful or criminal activity that they may witness or experience as long as they do not advocate for or coordinate harm”.

Compliance:

No content used in this research promotes criminal or harmful activities.

Section 4: Regulated goods

This policy was considered however, it was not applicable because the research did not involve the sale or purchase of goods and/or services.

Compliance:

No trading, exchange or gifting of drugs, firearms, or ammunition took place as part of this research.

Section 5: Fraud and deception

“In an effort to prevent fraudulent activity that can harm people or businesses, we remove content that purposefully deceives, wilfully misrepresents, or otherwise defrauds or exploits others for money or property. This includes content that seeks to coordinate or promote these activities using our services”.

Compliance:

No content in this research was used to conduct or promote fraudulent activities.

Part II. Safety

Section 6: Suicide and self-injury

This policy was considered however, it was not applicable as nothing in this research intentionally or unintentionally identifies and negatively targets victims or survivors of suicide or self-injury.

Compliance:

No content in this research deliberately or inadvertently encourages suicide or self-injury.

Section 7: Child sexual exploitation, abuse and nudity

This policy was considered however, it was not applicable because no part of this research involves or endangers minors.

Compliance:

No inappropriate content was used in this research.

Section 8: Sexual exploitation of adults

This policy was considered however, it was not applicable because no part of this research “depicts, threatens or promotes sexual violence, sexual assault or sexual exploitation”.

Compliance:

No inappropriate content was used in this research.

C.1. FACEBOOK

Section 9: Bullying and harassment

“Bullying and harassment happen in many places and come in many different forms, from making threats to releasing personally identifiable information, to sending threatening messages and making unwanted malicious contact. We do not tolerate this kind of behaviour because it prevents people from feeling safe and respected on Facebook.

We distinguish between public figures and private individuals because we want to allow discussion, which often includes critical commentary of people who are featured in the news or who have a large public audience. For public figures, we remove attacks that are severe as well as certain attacks where the public figure is directly tagged in the post or comment. For private individuals, our protection goes further: we remove content that’s meant to degrade or shame, including, for example, claims about someone’s sexual activity”.

Compliance:

No content in this research was used for threatening or malicious purposes.

Section 10: Human exploitation

This policy was considered however it was not applicable because no part of the research promotes the exploitation of humans for the “economic or material benefits of a third party”.

Compliance:

No content in this research promotes, or facilitates the exploitation of humans.

Section 11: Privacy violations and image privacy rights

“Privacy and the protection of personal information are fundamentally important values for Facebook. We work hard to safeguard your personal identity and information and we do not allow people to post personal or confidential information about yourself or of others.

We remove content that shares, offers or solicits personally identifiable information or other private information that could lead to physical or financial harm, including financial, residential and medical information, as well as private information obtained from illegal sources”.

Compliance:

Measures are in place as part of this research to redact information that may be used to identify other persons. No financial, residential or medical information was used in this research.

Part III. Objectionable content

Section 12: Hate speech

“We believe that people use their voice and connect more freely when they don’t feel attacked on the basis of who they are. That’s why we don’t allow hate speech on Facebook. It creates an environment of intimidation and exclusion, and in some cases may promote offline violence”.

Compliance:

This research did not create an environment of intimidation and exclusion.

Section 13: Violent and graphic content

“We remove content that glorifies violence or celebrates the suffering or humiliation of others because it may create an environment that discourages participation. We allow graphic content (with some limitations) to help people raise awareness about issues”.

Compliance:

None of the images used in this research displays violence.

C.1. FACEBOOK

Section 14: Adult nudity and sexual activity

“We restrict the display of nudity or sexual activity because some people in our community may be sensitive to this type of content. Additionally, we default to removing sexual imagery to prevent the sharing of non-consensual or underage content. Restrictions on the display of sexual activity also apply to digitally created content unless it is posted for educational, humorous or satirical purposes”.

Compliance:

No images used in this research can be classed as obscene in nature.

Section 15: Sexual solicitation

This policy was considered however, it was not applicable because no part of this research facilitates transactions that may lead to sexual solicitation.

Compliance:

No sexually explicit language was used and no content used in this research encourages sexual solicitation.

Section 16: Cruel and insensitive

“We believe that people share and connect more freely when they do not feel targeted based on their vulnerabilities. As such, we have higher expectations for content that we call cruel and insensitive, which we define as content that targets victims of serious physical or emotional harm”.

Compliance:

No content used in this research can be classed as cruel and insensitive in nature.

Part IV. Integrity and authenticity

Section 17: Account integrity and authentic identity

“Authenticity is the cornerstone of our community. We believe that authenticity helps create a community where people are accountable to each other, and to Facebook, in meaningful ways. We want to allow for the range of diverse ways that identity is expressed across our global community, while also preventing impersonation and identity misrepresentation. That’s why we require people to connect on Facebook using the name that they go by in everyday life. Our Authenticity Policies are intended to create a safe environment where people can trust and hold one another accountable”.

Compliance:

The accounts used for this research were private and only communicated with each other thus, were not used for impersonation.

Section 18: Spam

“We work hard to limit the spread of spam because we do not want to allow content that is designed to deceive, or that attempts to mislead users to increase viewership. This content creates a negative user experience, detracts from people’s ability to engage authentically in online communities, and can threaten the security, stability and usability of our services”.

Compliance:

The accounts used for this research were private and only communicated with each other thus, did not spam other users.

C.1. FACEBOOK

Section 19: Cybersecurity

“We recognise that the safety of our users extends to the security of their personal information and accounts and our services. Attempts to gather sensitive personal information by deceptive or invasive methods are harmful to the authentic, open and safe atmosphere that we want to foster, and can be used to compromise user accounts and our services. Therefore, we do not allow attempts to gather sensitive user information through the abuse of our platform and products”.

Compliance:

The accounts used for this research were private and only communicated with each other thus, did not attempt to gather sensitive personal information from other users.

Section 20: Inauthentic behaviour

“In line with our commitment to authenticity, we don’t allow people to misrepresent themselves on Facebook, use fake accounts, artificially boost the popularity of content or engage in behaviours designed to enable other violations under our Community Standards. This policy is intended to protect the security of user accounts and our services, and create a space where people can trust the people and communities they interact with”.

Compliance:

The accounts used for this research were private and only communicated with each. There were no attempts to engage in behaviours that violate the Community Standards.

Section 21: False news

“Reducing the spread of false news on Facebook is a responsibility that we take seriously. We also recognise that this is a challenging and sensitive issue. We want to help people stay informed without stifling productive public discourse. There is also a fine line between false news and satire or opinion. For these reasons, we don’t remove false news from Facebook, but instead significantly reduce its distribution by showing it lower in the News Feed”.

Compliance:

No content in this research was used to disseminate false news.

Section 22: Manipulated media

This policy was considered however, it was not applicable because no part of this research involves the manipulation and dissemination of image, audio or video content.

Compliance:

No media used in this research was manipulated.

Section 23: Memorialisation

This policy was considered however, it was not applicable because no memorial site(s) were used in this research.

Compliance:

No memorial sites were used in this research.

Part V. Respecting intellectual property

Section 24: Intellectual property

“Facebook takes intellectual property rights seriously and believes that they are important to promoting expression, creativity and innovation in our community. You own all of the content and information that you post on Facebook, and you control how it is shared through your privacy and application settings. However, before sharing content on Facebook, please make sure that you have the right to do so. We ask that you respect other people’s copyrights, trademarks and other legal rights”.

Compliance:

Images used in this research are covered by the Pexels [2020] and Pixabay [2020] Licenses. They were not distributed publicly nor used for commercial purposes.

Part VI. Content-related requests and decisions

Policies under this part of the Community Standards are related to Facebook’s compliance with requests from users and/or law enforcement, including board oversight.

These were reviewed but were not applicable to this research as they are not specific to user behaviour on the platform.

C.2 Google Drive

To read the Google Drive policies in full, see “The Terms” [Google, 2020a,e].

Program Policies

Account Hijacking

“Do not access another user’s account without their permission. We may take appropriate action if we are notified of account hijacking, which may include removing access to some of our products or disabling your Google Account”.

Compliance:

This research did not access or attempt to access the accounts of users other than those specifically created as part of this research.

Account Inactivity

“Use the product to remain active. Activity includes accessing the product’s content or storing new content on the product at least once every 2 years. We may take action on inactive accounts, which may include deleting your content from the product”.

Compliance:

Accounts used were kept active however, they will be deleted at the conclusion of this research.

Child Sexual Abuse and Exploitation

This policy was considered however, it was not applicable because no part of this research involves or endangers minors.

Compliance:

No inappropriate content was used in this research.

Circumvention

“Do not engage in actions intended to bypass our policies or subvert restrictions placed on your account. This includes the creation or use of multiple accounts or other methods intended to engage in a behavior that was previously prohibited. We may take appropriate action if we are notified of circumvention, which may include removing access to some of our products or disabling your Google Account”.

Compliance:

Accounts used in this research were not used to bypass Google’s terms of use.

C.2. GOOGLE DRIVE

Dangerous and Illegal Activities

“Do not use this product to engage in illegal activities or to promote activities, goods, services, or information that cause serious and immediate harm to people or animals. While we permit general information for educational, documentary, scientific, or artistic purposes about this content, we draw the line when the content directly facilitates harm or encourages illegal activity. We will take appropriate action if we are notified of unlawful activities, which may include reporting you to the relevant authorities, removing access to some of our products, or disabling your Google Account”.

Compliance:

No trading or exchange of goods and/or services that directly facilitates harm or encourages illegal activity occurred as part of this research.

Harassment, Bullying, and Threats

“Do not harass, bully, or threaten others. We also don’t allow this product to be used to engage or incite others in these activities. This includes singling someone out for malicious abuse, threatening someone with serious harm, sexualizing someone in an unwanted way, exposing private information of someone else that could be used to carry out threats, disparaging or belittling targets of violence or tragedy, inciting others to carry out these activities, or harassing someone in other ways”.

Compliance:

No content in this research was used for threatening or malicious purposes.

Hate Speech

“Do not engage in hate speech. Hate speech is content that promotes or condones violence against or has the primary purpose of inciting hatred against an individual or group on the basis of their race or ethnic origin, religion, disability, age, nationality, veteran status, sexual orientation, gender, gender identity, or any other characteristic that is associated with systemic discrimination or marginalization”.

Compliance:

This research did not create an environment of intimidation and exclusion.

Impersonation and Misrepresentation

“Do not impersonate a person or organization or misrepresent yourself. This includes impersonating any person or organizations you don’t represent; or providing misleading information about a user/site’s identity, qualifications, ownership, purpose, products, services, or business.

This also includes content or accounts misrepresenting or concealing their ownership or primary purpose such as misrepresenting or intentionally concealing your country of origin or other material details about yourself when directing content about politics, social issues, or matters of public concern to users in a country other than your own. We do allow parody, satire, and the use of pseudonyms or pen names – just avoid content that is likely to mislead your audience about your true identity”.

Compliance:

The accounts used for this research were private and only communicated with each other thus, were not used for impersonation or to mislead other users.

Malware and Similar Malicious Content

“Do not transmit malware or any content that harms or interferes with the operation of the networks, servers, end user devices, or other infrastructure. This includes the direct hosting, embedding, or transmission of malware, viruses, destructive code, or other harmful or unwanted software or similar content. This also includes content that transmits viruses, causes pop-ups, attempts to install software without the user’s consent, or otherwise impacts users with malicious code”.

Compliance:

No content in this research was used to transmit malware or interfere with the operation of the network.

C.2. GOOGLE DRIVE

Misleading Content

This policy was considered however, it was not applicable because no part of this research involves the manipulation and dissemination of image, audio or video content.

Compliance:

No content used in this research was manipulated or designed to mislead.

Non-Consensual Explicit Imagery (NCEI)

This policy was considered however, it was not applicable because no part of this research involves sexually explicit, or non-explicit intimate content.

Compliance:

No content used in this research promotes NCEI.

Personal and Confidential Information

“Do not store or distribute other people’s personal or confidential information without authorization. This includes the use of sensitive information, such as U.S. Social Security numbers, bank account numbers, credit card numbers, images of signatures, and personal health documents. In most cases where this information is broadly available elsewhere on the internet or in public records, like national ID numbers listed on a government website, we generally don’t process enforcement actions”.

Compliance:

The accounts used for this research were private and only communicated with each other thus, did not attempt to store/distribute other users’ sensitive personal information.

Phishing

“Do not use this product for phishing. This includes soliciting or collecting sensitive data such as passwords, financial details, and social security numbers”.

Compliance:

No content in this research was used to conduct or promote fraudulent activities.

Regulated Goods and Services

This policy was considered however, it was not applicable because the research did not involve the sale or purchase of goods and services - regulated or otherwise.

Compliance:

No trading of drugs, alcohol, unapproved supplements, health/medical devices, firearms, or ammunition occurred as part of this research.

Sexually Explicit Material

“Do not distribute content that contains sexually explicit material, such as nudity, graphic sex acts, and pornographic material. This includes driving traffic to commercial pornography sites. We allow nudity for educational, documentary, scientific, or artistic purposes”.

Compliance:

No images used in this research can be classed as obscene in nature.

Spam

“Do not spam. This may include unwanted promotional or commercial content, unwanted content that is created by an automated program, unwanted repetitive content, nonsensical content, or anything that appears to be a mass solicitation”.

Compliance:

The accounts used for this research were private and only communicated with each other thus, did not spam other users.

C.2. GOOGLE DRIVE

System Interference and Abuse

“Do not abuse this product and do not harm, degrade, or negatively affect the operation of networks, devices, or other infrastructure of Google or others.

This includes degrading, disabling, or negatively interfering with any aspect of the product or its services. We may take appropriate action if we are notified of system interference and abuse, which may include removing access to some of our products or disabling your Google Account”.

Compliance:

No part of this research was used to interfere with the operation of network.

Terrorist Activities

This policy was considered however, it was not applicable because the research was not conducted by or on behalf of an organisation or individual with a violent disposition.

Compliance:

No content used in this research promotes unlawful/dangerous organisations, individuals, or any acts committed by them. None of the images used in this research portrays violence.

Unauthorized Images of Minors

This policy was considered however, it was not applicable because no part of this research involves or endangers minors.

Compliance:

No inappropriate content was used in this research.

Violence and Gore

“Do not store or distribute violent or gory content involving real-life people or animals that’s primarily intended to be shocking, sensational, or gratuitous.

This includes ultra-graphic violence, such as dismemberment or close-up footage of mutilated corpses. Graphic material, such as content containing significant amounts of blood, may be allowed in an educational, documentary, scientific, or artistic context, but please be mindful to provide enough information to help people understand the context”.

Compliance:

None of the images used in this research displayed violence.

Additional Policies

Copyright Infringement

“Respect copyright laws. Do not share copyrighted content without authorization or provide links to sites where people can obtain unauthorized downloads of copyrighted content. It is our policy to respond to clear notices of alleged copyright infringement. Repeated infringement of intellectual property rights, including copyright, will result in account termination. If you see a violation of Google’s copyright policies, report copyright infringement”.

Compliance:

Images used in this research are covered by the Pexels [2020] and Pixabay [2020] Licenses. They were not distributed publicly nor used for commercial purposes.

C.3 Twitter

To read these policies in full, see The Twitter Rules [Twitter Help Center, 2020].

Safety

Violence

“You may not threaten violence against an individual or a group of people. We also prohibit the glorification of violence”.

Compliance:

No content used in this research constitutes promotes or facilitates violence.

C.3. TWITTER

Terrorism/violent extremism

“You may not threaten or promote terrorism or violent extremism”.

Compliance:

No content used in this research portrays violence or constitutes a threat to public safety.

Child sexual exploitation

This policy was considered however, it was not applicable because no part of this research involves or endangers minors.

Compliance:

No inappropriate content was used in this research.

Abuse/harassment

“You may not engage in the targeted harassment of someone, or incite other people to do so. This includes wishing or hoping that someone experiences physical harm”.

Compliance:

No content in this research was used for threatening or malicious purposes.

Hateful conduct

“You may not promote violence against, threaten, or harass other people on the basis of race, ethnicity, national origin, caste, sexual orientation, gender, gender identity, religious affiliation, age, disability, or serious disease”.

Compliance:

This research did not create an environment of intimidation and exclusion.

Suicide or self-harm

This policy was considered however, it was not applicable as nothing in this research intentionally or unintentionally promotes suicide or self-harm.

Compliance:

No content in this research deliberately or inadvertently encourages suicide or self-harm.

Sensitive media, including graphic violence and adult content

“You may not post media that is excessively gory or share violent or adult content within live video or in profile or header images. Media depicting sexual violence and/or assault is also not permitted”.

Compliance:

No images used in this research can be classed as obscene in nature.

Illegal or certain regulated goods or services

This policy was considered however, it was not applicable because the research did not involve the sale or purchase of goods and services - regulated or otherwise.

Compliance:

No trading of goods or services took place as part of this research.

Privacy**Private information**

“You may not publish or post other people’s private information (such as home phone number and address) without their express authorization and permission. We also prohibit threatening to expose private information or incentivizing others to do so”.

Compliance:

Measures are in place as part of this research to redact information that may be used to identify other persons. No financial, residential or medical information was requested or used in this research.

C.3. TWITTER

Non-consensual nudity

This policy was considered however, it was not applicable because no part of this research involves sexually explicit, or non-explicit intimate content.

Compliance:

No part of this research involves the distribution of intimate content.

Authenticity

Platform manipulation and spam

“You may not use Twitter’s services in a manner intended to artificially amplify or suppress information or engage in behavior that manipulates or disrupts people’s experience on Twitter”.

Compliance:

The accounts used for this research were private and only communicated with each other and thus, did not interfere with the operation of the network or spam other users.

Civic Integrity

This policy was considered however, it was not applicable because no part of this research involves engagement/participation in civic processes.

Compliance:

The accounts used for this research were private and only communicated with each other and thus, did not interact with the public.

Impersonation

“You may not impersonate individuals, groups, or organizations in a manner that is intended to or does mislead, confuse, or deceive others”.

Compliance:

The accounts used for this research were private and only communicated with each other. Thus, were not used for impersonation or to mislead other users.

Synthetic and manipulated media

This policy was considered however, it was not applicable because no part of this research involves the distribution of manipulated or synthetic content.

Compliance:

No content used in this research was manipulated or designed to mislead.

Copyright and trademark

“You may not violate others’ intellectual property rights, including copyright and trademark”.

Compliance:

Images used in this research are covered by the Pexels [2020] and Pixabay [2020] Licenses. They were not distributed publicly nor used for commercial purposes.

Enforcement and Appeals

This part of Twitter’s rules discusses the policy development process, the approach to enforcement, including any possible consequences of policy violations. It also discusses how to appeal enforcement actions.

These were reviewed but no further action was taken as there was no requirement to appeal enforcement actions as part of this research.

Third-party advertising in video content

This part of Twitter’s rules prohibits the submission/display of third-party advertising in video content without prior consent.

This section was reviewed and no further action was taken as there was promotion of sponsored content in this research.

Appendix D: Peer Reviewed Publications

D.1 SMURF - Manuscript

David, Anne; Morris, Sarah; and Appleby-Thomas, Gareth (2021) "Social Media User Relationship Framework (SMURF)," Journal of Digital Forensics, Security and Law: Vol. 16 , Article 1.

DOI: <https://doi.org/10.15394/jdfsl.2021.1698>

Available at: <https://commons.erau.edu/jdfsl/vol16/iss1/1>

SOCIAL MEDIA USER RELATIONSHIP FRAMEWORK (SMURF)

Anne David¹, Sarah Morris², Gareth Appleby-Thomas³

^{1,2}Centre for Electronic Warfare, Information, and Cyber

³Centre for Defence Engineering

Cranfield University, Shrivenham, UK

^{1,2,3}{digitalinvestigation}@cranfield.ac.uk

ABSTRACT

The use of social media has spread through many aspects of society, allowing millions of individuals, corporate as well as government entities to leverage the opportunities it affords. These opportunities often end up being exploited by a small percentage of the user community who use it for objectionable or unlawful activities; for example, trolling, cyber bullying, grooming, luring. In some cases, these unlawful activities result in investigations where swift retrieval of critical evidence required in order to save a life.

This paper presents a proof of concept (PoC) framework for social media user attribution. The framework aims to provide digital evidence that can be used to substantiate user activity in live triage investigations. This paper highlights the use of live triage as a viable technique for the investigation of social media activity, contextualizing user activity and attributing actions to users. It discusses the reliability of artefacts other than the communications content as a means of drawing inferences about user social media activity, taking into account the proportionality and relevance of such evidence.

Keywords: digital forensics, digital investigation, digital investigation framework, live triage, relationship attribution, social media

1. INTRODUCTION

Social media presents investigators with a plethora of useful digital evidence as shown by a range of criminal prosecutions and convictions as well as civil investigations with outcomes such as disciplinary measures. Thus, the timely recovery of these artefacts is considered of interest to investigators during a live triage.

Triage as a practice emerged in the field of medicine where it was used to assess hospital patients in order to establish the order of prioritization of treatment (Robertson-Steel,

2006). In digital forensics however, triage is a fast-growing practice with two streams; live and post-mortem triage.

In time sensitive investigations, the need to quickly obtain actionable intelligence, in a usable format from a live system (Wiles & Reyes, 2007) may take precedence over waiting to conduct a detailed forensic analysis. This paper proposes a proof of concept (PoC) framework for social media user attribution, to support investigators through the automated, targeted recovery artefacts from social media activity in a live triage.

This paper is focused on social media access through a web browser and aims to highlight the potential in using artefacts other than communication content to infer and contextualise user activity, and to attribute actions to a user. The proposed framework is intended to be applicable and generalisable to other triage investigations. The web browser(s) referenced in this paper is a demonstration for the framework.

1.1 Contribution

The key contributions of this paper are as follows:

- ⇒ the proposal of a PoC framework for live triage investigations involving social media
- ⇒ defines a baseline for the artefacts of interest as a method for grouping social media artefacts based on the perceived user activity. This can be used in the prioritization of live triage data analysis
- ⇒ a demonstration to test its applicability, generalisability and reliability through the identification of how SMURF can extend or complement existing triage tools

1.2 Paper Structure

The rest of this paper is structured as follows: related work is discussed in Section 2; SMURF is introduced in Section 3, discussing the problem definition, characteristics and development. Section 4 discusses the build process and how SMURF works. In Section 5, the PoC implementation is discussed including data generation, analysis and the PoC deployment. Section 6 presents a case study and reliability assessment. Section 7 contains the discussion, and Section 8 presents the conclusion and future work.

2. LITERATURE REVIEW

This section provides a background on related work in the investigation of social media activity, the rules and reliability of digital evidence, and live triage investigations. It discusses the challenges faced by practitioners in the course of investigating user social media activity, and acquiring digital evidence that can be used for the attribution of actions and relationships to a user.

2.1 Social Media and Digital Evidence

Social media provides an opportunity for users to connect with others of similar interests, form personal or business relationships, and to communicate in real time through sharing multimedia files, instant messaging, microblogging, advertising etc.

It also provides investigators with valuable evidence that can be used to connect a suspect (or suspects) to a criminal activity of interest or to other persons of interest.

Social media has been described by Zeng et al. (2010) as a distributed mode of generating, disseminating content and communication among communities. Its use as a source of digital evidence has been recognised as an area of interest (Arshad et al., 2019) deserving of further research in the digital forensics community.

There have been a number of approaches taken by researchers and practitioners exploring social media activity. For example, Shaw et al. (2016) takes a network forensics approach, reviewing the vulnerabilities of social media platforms. Hubert (2014) presents techniques that can be applied to non-criminal investigations such as employee misconduct.

To understand social media communication within the context of this paper, it is

necessary to define communication in a ‘traditional’ sense and how social media fits into the concept of communication.

Communication, according to Dance (1967) has been described as “**the elicitation of a response from another**, without making a distinction between the interaction of animate and inanimate matter, or the interaction between humans and animals”. Littlejohn (1992) suggests that communication can be defined by “looking at **the conceptual components of the act**, for example, speech, symbols, interaction, and intention”.

These definitions outline, in a simplistic way, what communication entails i.e. “eliciting a response from another” through writing, speaking, the use of signs or symbols. It is also worthy to note that Littlejohn includes ‘**intention**’ as one of the conceptual components of the act of communicating.

Viewing these from a digital forensics perspective, communication on social media occurs through interactions/exchanges (**action**) that ‘elicit a response’ from other users, and these interactions tend to involve the use of textual, graphic, audio content amongst others. These exchanges to a degree, often reflect the ‘intent’ of the user initiating the interaction and may be used to form associations (**relationships**) on social media platforms.

This could be considered as useful evidence to digital forensics investigators. The content shared, when recovered can be submitted as digital evidence while the nature of the interaction when contextualised may be used to infer the user’s intent (**motive**) for example, the user may intend to provoke other users, commit a criminal act, or incite another user into committing a criminal act.

2.1.1 Challenges Posed by Social Media Communication

Although it can be a valuable source of digital evidence, as with every technology, social media also presents challenges to investigators. Some of which include the use of proprietary data formats across supported devices, the type of artefacts written to disk, and the ability of a portion of the platform’s user community to engage in undesirable or unlawful activities (Bello & DiBlasio, 2013; Moore, 2014; Select Committee on Communications, 2014).

There has been several social media related prosecutions and convictions over the years, highlighting cases where social media has been used for activities such as trolling, murder, drug trafficking, and fraud (Haroon & Carter, 2010; Bello & DiBlasio, 2013).

There is also the lack of standardisation (Cusack & Son, 2012) which often results in social media investigations being approached as a web browser focused investigation when the social media platform is accessed through a web browser. Self-contained social media applications may handle things differently but were not explored as they were deemed out of scope. It has been suggested that ACPO Guidelines can be used as a starting point when investigating social media activity in the UK.

Other challenges associated with the investigation of social media activity include communication content not being immediately available or accessible on disk (David et al., 2020) without additional measures/-efforts to retrieve it for example, the use of file carving scripts; there are also possible privacy implications as highlighted by Casey (2013) because some artefacts may contain information about unrelated users or activities.

2.1.2 Requirements of Admissibility of Digital Evidence

Digital evidence has been defined in different ways depending on the context of the investigation (i.e. criminal or non-criminal). Casey (2004) defines digital evidence as “*data stored or transmitted using a computer, which supports or refutes a theory on how an event occurred*”; thus addressing the critical elements of a crime such as determining the intent of a suspect.

Mukasey et al. (2008) defines digital evidence as “*information or data of investigative value that is stored on, received or transmitted by a computer*”; highlighting the role of digital evidence - its “**investigative value**” in criminal (or non-criminal) proceedings.

Digital evidence is known to exhibit a number of characteristics e.g.:

- ⇒ Latency: the evidence must be processed in order to derive meaningful and usable data from it.
- ⇒ Fidelity: the original evidence is not necessarily required. A copy of the original would suffice provided that the copying process does not invalidate the evidence.
- ⇒ Volatility: digital evidence is at risk of intentional or unintentional alteration or deletion. Intentional changes may be an attempt by an individual to corrupt and invalidate the evidence (McKemmish, 2008) however, this can be addressed by maintaining an audit trail and a comprehensive log of the evidence recovery process.

In order to be considered valuable to an investigation, and to be admissible in a court of law, digital evidence needs to meet certain requirements which include being relevant to the case, reliable, authentic, complete (i.e. paints an understandable, full picture), believable and the recovery method

must be repeatable and should produce the same results when conducted by an independent third party (ACPO, 2012).

These requirements complement the characteristics of digital evidence and guide its admissibility in court. It is important to note that there may be circumstances where changes to digital evidence in the course of an investigation is unavoidable; this can be managed by maintaining a comprehensive record of the interactions between the investigator and the target device and also the possible implications of those interactions.

2.2 Live Triage Investigations

Triage, historically has been in practice in the medical field and has been defined as “*the process of quickly examining patients who are taken to a hospital in order to decide which ones are the most seriously ill and must be treated first*” (Cambridge University Press, 2019). In digital forensics however, this definition can be used to describe two scenarios: an on-site/live triage or an off-site/post-mortem triage, depending on the type of investigation. Post-mortem triage is out of scope for this paper, for more information, see (Garfinkel, 2013; Jusas et al., 2017; Parsonage, 2009; Roussev & Quates, 2012).

Live (on-site) triage is a technique used by investigators to quickly retrieve actionable intelligence at a crime scene. It is implemented in time sensitive investigations such as kidnapping, missing persons; where there is an imminent threat to life; the use of encryption on a device; the need to extract and preserve volatile artefacts (Bashir & Khan, 2013) or the practicality of seizing a device.

Live triage is appropriate when the requirement for actionable intelligence outweighs the need to perform a lab-based, detailed forensic analysis on all potential sources of digital evidence discovered at a crime scene (Wiles & Reyes, 2007). The data recovered during a live triage can be used to

guide the investigation by highlighting possible areas or persons of interest prior to a full forensic analysis of the digital storage media.

Existing research has proposed a number of techniques in the use of live triage in digital forensics investigations (Gielen & Bolzoni, 2014; Montasari, 2016). These highlight live triage as a viable means of recovering intelligence that can be used to generate new leads and to assess the severity of a crime.

The Computer Forensic Field Triage Process Model (CFFTPM) developed by Rogers et al. (2006) can be applied to a wide spectrum of investigations and is considered a foundation for modelling the live triage process. It is focused on on-site analysis of digital devices, to obtain information that can be used during the “search and execution” stage of the investigation or during a suspect’s interview. CFFTPM highlights the need to consider live triage as part of the digital forensics process.

The CFFTPM is not automated and is designed to be used by competent users. This requirement for examiner competency presents some challenges as additional resources may be expended on-site where the on-site investigation takes priority over existing cases (Hitchcock et al., 2016), resulting in the creation of backlogs in the forensic lab as existing cases may need to be put on hold.

Hitchcock et al. however, in the Digital Field Triage Model (DFT) proposed that providing non-specialists with training on the skills required to supplement/support investigators and analysts on-site would help digital forensics units maximize resources and would also provide the information required during a live triage. This will ensure that analysts required in the lab are able to focus on lab-based work.

Cantrell et al. (2012); Cantrell and Dampier (2012) in the Digital Triage Process Model, also addressed the challenges of

appropriately managing resources and the utilization of non-digital forensics specialists during live triage investigations by introducing and implementing semi-automated steps in the proposed process model. Semi-automation, it was argued, would make it easier for non-specialists to use; speed up the live triage process and make it possible for these processes to be scripted into a case-specific framework.

There are concerns around the use of live triage during investigations, specifically highlighting the possibility of data being written to the device being investigated. This is often dependent on the type of investigation. For example, during an incident response investigation, the objective is usually business continuity and ensuring that compromised systems are returned to a ‘good working state’ thus, creating files or writing to the target device is not often considered a problem (Gielen & Bolzoni, 2014).

During digital forensics investigations, the state of digital data is to remain unmodified where possible. Thus, in compliance with the relevant jurisdictional guidance and best practice, (e.g. ACPO Principle #2 as UK Guidance), all actions taken during a live triage including any possible impact must be documented and justified.

There are also concerns that a live triage may overlook pertinent evidence (Casey, 2013) however, this can be addressed by conducting a detailed forensic analysis of the device, building on the evidence recovered during the live triage. The detailed analysis may be conducted on-site or in the lab as appropriate.

In addition to the on-going research into the use of and applicability of live triage processes in digital forensics, further work is required to capture and address how triage tools can be used to:

⇒ supplement the limitations of existing tools and processes.

- ⇒ manage targeted capture of specific volatile or case-based artefacts that can provide investigators with actionable intelligence.

2.3 Digital Evidence: Challenges

Digital evidence aims to link an offender to a crime or to exonerate a suspect. A number of factors however, pose a challenge to the recovery and use of digital evidence. For example, differences in operating systems, applications and storage methods; the use of concealment and cryptographic techniques (DFRWS, 2001); the complexity of digital devices (7safe, 2014; TWGECSI, 2001); the volume of data to be analysed (ACPO, 2012); data compression and the storage capacity of the digital devices (McKemmish, 2008; Sommer, 1999); spoliation techniques (Marcella & Menendez, 2007).

Evidence from user Internet activity may change with little or no notice for example, changes to the browser data storage format (e.g. proprietary compression format) and the type of data stored (e.g. cookies, history) could influence the type of data available for recovery. This often means that it takes longer to understand and decode the data in order to extract meaningful information from it.

Marcella and Menendez (2007) suggests reviewing and assessing new technologies from an offender's perspective in order to understand how it can be used to hinder an investigation. This is because technology often ends up being used in a manner that deviates from its original purpose. Encryption and evidence eliminating tools are widely available, often for free on the Internet and can be used to attempt to evade detection or prosecution.

2.4 The Reliability of Digital Evidence

In a generic sense, reliability has been defined as “*the quality of being able to be trusted or believed because of working or behaving well*” (Cambridge University Press, 2019).

Digital evidence could be user or system generated (David et al., 2020) thus when determining its reliability, it is important to assess how trustworthy it is. This is because a number of system activities outside the control of the user may create artefacts that appear related to a case but may not be for example, background processes such as web browser caching.

Casey (2011) suggests that an effective way of assessing the reliability of digital evidence is to focus on the evidence itself (e.g. checking the evidence for signs of tampering/damage), rather than the process through which it was created.

Due to the nature of live triage investigations, where repeated access to a target device is not advisable, reliability can be assessed by:

- ⇒ logging and detailed documentation of the recovery process.
- ⇒ how believable the evidence is;
- ⇒ and how accurately such evidence can be correlated, corroborated and attributed to specific events. For example, artefact ‘A’ infers that the user ‘Bob’ added user ‘Chuck’ as a friend and has visited ‘Chuck’s’ profile ‘X’ times; this inference is supported by JSON and cache artefacts which provide context and imply that ‘Bob’ and ‘Chuck’ have a social relationship.

2.4.1 Relationship Attribution

Attribution in digital forensics is used to show a link between a suspect to the artefacts found on a digital device. In the con-

text of this paper, relationship attribution is aimed at determining the degree to which a specific artefacts can be attributed to a specific event, using the artefacts recovered during a live triage.

This includes leveraging corroborating artefacts such as browser cached content, system generated artefacts etc. (David et al., 2020) to show the connection between a user and social media activity or social relationships and associations. This would facilitate the prompt identification of persons of interest to an investigation prior to a detailed forensic analysis.

3. SMURF

SMURF is a framework built from a collection of components, and designed for live triage investigations focused on user activity on social media in digital forensics.

SMURF is a command-line interface (CLI) tool designed to automate the recovery of artefacts that can be used by investigators to infer and attribute actions to a user.

This section discusses the characteristics and development of SMURF. It also discusses how SMURF can be used by investigators to effectively and quickly build and deploy triage tools during an investigation.

3.1 Framework Problem Definition

David et al. (2020) presented results from a manual analysis of user activity on Twitter. This was done as a “dead disk” forensic analysis and provided some examples of the type of artefacts that an investigator may expect to find on digital storage media where a user has engaged in activity on Twitter.

In some time sensitive cases, digital forensic investigators are required to perform a live triage in order to obtain actionable intelligence that will help save the life of an

individual (e.g. missing person). In such instances, this immediate need for actionable intelligence takes precedence over a full disk analysis which can be conducted later.

SMURF intends to address the need for a generalisable, reusable, and adaptable technique for automated evidence recovery from a variety of usage scenarios, providing a reusable design with multiple levels of functionality. To do this, SMURF is designed for:

- I. use in a live triage investigation
- II. the furtherance of the automated recovery, categorising recovered artefacts based on possible usage scenarios and the enabling the attribution of actions or relationships to a user.
- III. the provision of quick and usable information that would enable investigators define the initial stages of an investigation, generate new leads, and highlight possible areas of interest to be looked at during a detailed forensic analysis.

3.2 Framework Characteristics

The development process for SMURF involved considering the characteristics that would define its functionality. This includes determining the availability and location of artefacts of interest, determining the types of artefacts that can be used to provide context to user activity.

This process also includes understanding that certain types of artefacts such as the contents of communication (e.g. text, multimedia files) may not be readily available to view or accessible on disk as they may be in unallocated space. Thus, it may be necessary to recover and use other artefacts that could be used in making inferences about user activity.

It is also important to understand how these artefacts can be used to attribute ac-

tions and relationships to a user based on the amount and usefulness of the actionable intelligence SMURF provides an investigator. This will enable an investigator to understand the context of the artefact, how it came to be, and what possible usage scenario could have created it. It will also assist the investigator when making a judgement on how to proceed.

3.3 Framework Development

The limitations observed with the ability to recover artefacts from social media activity (Section 3.2, i.e. communications content in unallocated space) highlights the need to identify and define the artefacts that may be of interest during a triage investigation.

Examples of these artefacts include browser history data such as presented in David et al. (2020) i.e. URLs, authentication credentials, session information, cached data etc. as well as system artefacts that provide context about the user accounts, programs and applications running on their digital storage media.

3.3.1 Components

This stage of the framework development process involved identifying and defining the major parts that constitute the framework. It involved evaluating the artefacts recovered during the manual investigation of the disk image and determining whether the artefacts would provide quick, easy to assimilate information for an investigator during a live triage.

The manual artefacts were subsequently categorised into context and case specific artefacts; then they were broken down into specific artefacts from which the framework components were derived.

Examples include:

- I. a component that recovers artefacts containing user account information, such as the number of user accounts on

- the computer, the user Security Identifier (SID), and logon/logoff information
- II. component for the recovery of user data from social media activity

As SMURF is intended for live triage investigations, it was determined that a component would be required for logging the interaction between the deployment device and the target device in compliance with ACPO #2.

3.3.2 Development Environment

This stage of the framework involved identifying a suitable programming language for the proof of concept (PoC). It also involved reviewing existing work with open source live triage tools to enable a gap analysis as part of this research. The gap analysis identified the capabilities of existing tools and how SMURF augments them.

The development environment setup is as follows:

- ⇒ Windows 10 Education (Local Machine)
- ⇒ Python 3
- ⇒ PyCharm Professional Edition with Anaconda Plugin

Scripting Individual Components

Each component of the framework (as described in Section 3.3.1) was scripted to recover specific artefacts. In addition to the artefact-specific components, a ‘core’ component was also scripted to collate the recovered artefacts, and present them in a HTML report for the investigator. This approach can be described as a “*practitioner friendly*” approach to automated evidence recovery in a live triage.

This process is described further in Section 4.2 where a graphical illustration of the components is presented.

4. BUILDING THE SMURF FRAMEWORK

This section discusses at a high level, the theoretical aspects of SMURF. The intention was to build a linear, modular investigative

tool, with the potential to extend and mature its functionality. SMURF's modularity allows for the addition or exclusion of components based on the use case.

Figure 1 is a graphical representation of how SMURF works. The components used to achieve these are discussed in Section 5.

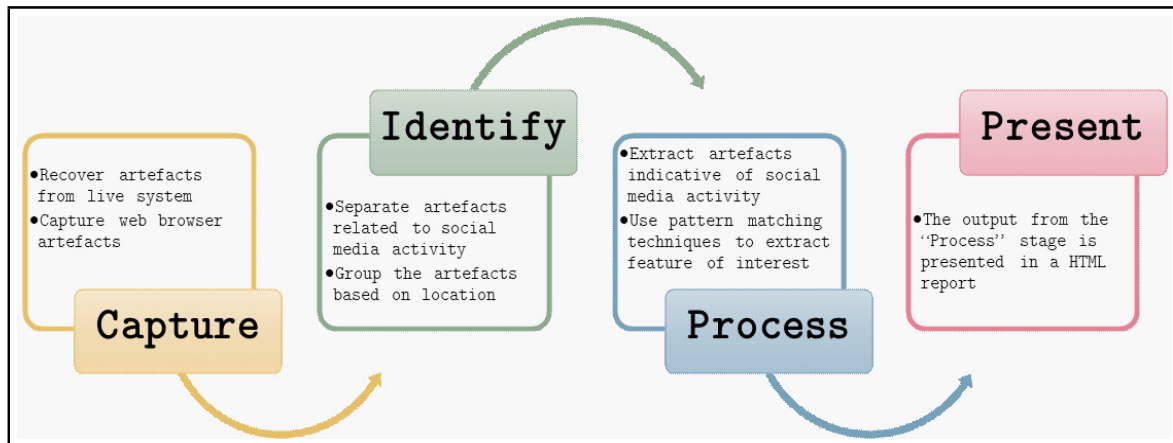


Figure 1. An illustration of how SMURF works

4.1 Capture

This stage involves recovering data from a live machine. As this is the stage that initiates interaction between the triage device and the target device, it was necessary to first programmatically record the triage device details, the date and time it was connected to the target device before proceeding with the data collection.

The data collection at this stage focused on system generated and user generated artefacts (e.g. web browser artefacts).

4.2 Identify

This stage involves segregating the data collected during the capture. The artefacts are grouped based on type and the location found. For example, if the data contains the social media platform name or an associated name, and it is a URL (**type**), put it in the "visited places group" for further processing.

4.3 Process

The artefacts recovered and grouped at the "Identify" stage are processed to highlight the type of activity that could have created them. For example, user account registration, viewing and sharing content etc.

This stage also involves the use of pattern matching techniques as presented in David et al. (2020) to extract features of interest.

4.4 Present

The data from preceding stages feed into the "Present" stage. Here, the results from the processing stage are parsed and returned in a text based HTML report. Following the theme of a live triage, this report provides the investigator with a quick view of the data recovered, in a usable manner.

5. CREATING A POC SMURF IMPLEMENTATION

This section presents the data generation and analysis methodology for a PoC.

5.1 Data Generation

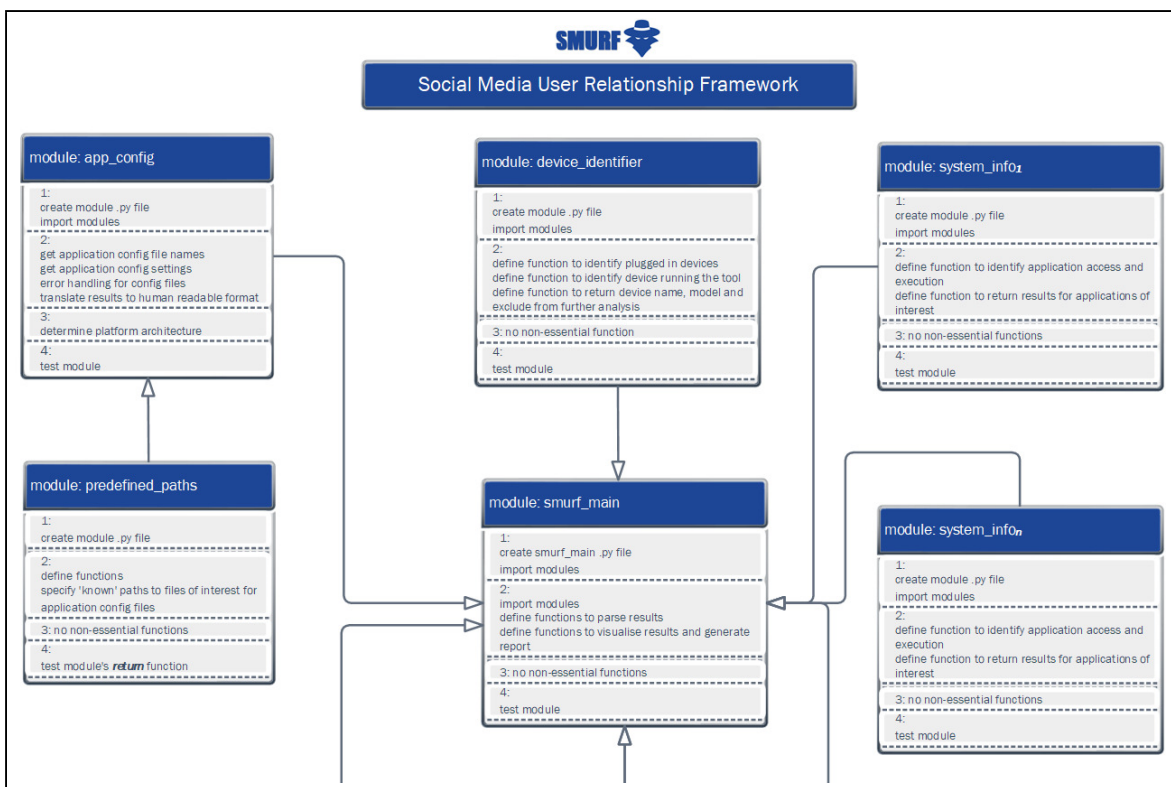
The dataset used in this paper was generated from previous experiments conducted as part of the work presented in (David et al., 2020). It involved the simulation of a range of normal user activity on Twitter, some of which include a variety of the activities listed below:

- ⇒ Power on and log in to the experimental VM
- ⇒ Launch the web browser (Google Chrome and Firefox were used)
- ⇒ Login to test user account on Twitter via the browser
- ⇒ Search for users to follow (publicly accessible accounts e.g. @bbcgoodfood)
- ⇒ Sending tweets (text, images, links)
- ⇒ Reply and retweet messages
- ⇒ Viewing and sending Direct Messages (DMs)
- ⇒ Sending photos via DM
- ⇒ Retweets via DM
- ⇒ Send follow requests to other test user account
- ⇒ Viewing and accepting follow requests
- ⇒ Updating the account privacy settings
- ⇒ Scrolling through the test user account timelines and that of its followers

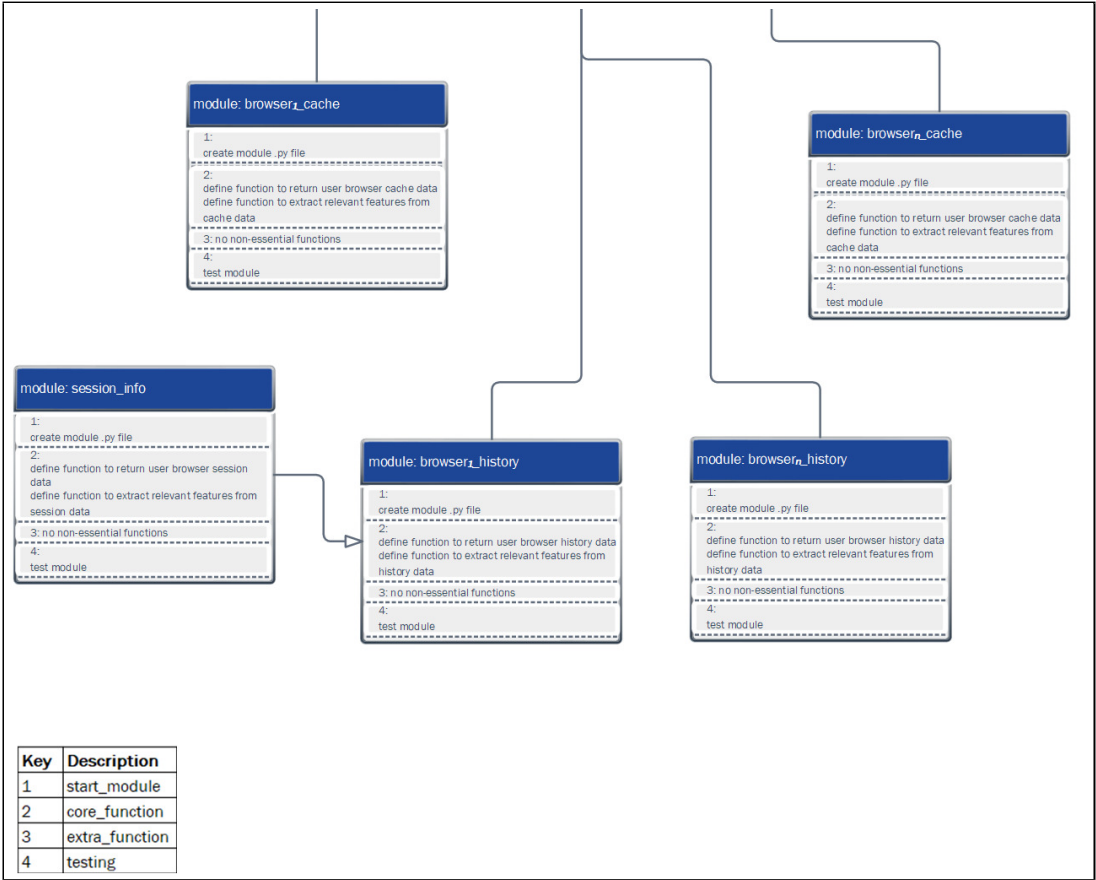
5.2 Data Analysis

The data analysis for this paper was conducted in a Windows 10 desktop environment using the PoC framework - SMURF. This section discusses how individual components of the SMURF were used for data analysis; it includes code extracts with several lines removed for brevity.

A graphical illustration of SMURF is shown in Figures 2(a) and 2(b). It shows the components that make up the framework and how they feed data into the core component enabling the generation of a report with the relevant information required by an investigator.



(a) Part 1



(b) Part 2

Figure 2. A “practitioner friendly” illustration of SMURF components and how data is fed into the main component

5.2.1 Component: device_identifier

This component was used to retrieve the identifying information for the triage device on which SMURF was running. This includes the Vendor ID (VID) and Product ID

(PID) of the device.

This information is written to the report to account for the interaction between the triage device and the target device being investigated.

Extracts from device_id.py

```
# Path to log file = "C:\Windows\INF"
windows_dir = os.path.abspath(os.sep)
path_to_logfile = pathlib.Path(windows_dir).joinpath('Windows', 'INF')

# Read data from setupapi.dev.log to get
# 'Device Install' Events for USB Devices
def get_devices():
    # Check that the setupapi.dev.log file exists on this machine
    if pathlib.Path(path_to_logfile).joinpath('setupapi.dev.log').is_file():

        # Read lines in setupapi file and search for device identifiers
        try:
            device_list = []

            with open(os.path.join(path_to_logfile, 'setupapi.dev.log'), 'r') as log_file:
                # Get the OS info from the first 6 lines of the log file (currently in Win 10).
                # This may change in the future if it does, amend the number of lines to read in
                .
                for i in range(6):
                    os_info = str(log_file.__next__().strip(""))
                    device_list.append(os_info)
```

5.2.2 Component: predefined_paths

This component was written to support artefact recovery from Windows Registry. It contains known paths of interest in the SAM, SECURITY, SOFTWARE, SYSTEM, NTUSER.DAT and UsrClass.dat hives and was used as a global settings file for the registry parsers.

The paths held in this component can be used to provide context to user activity in a live triage. For example, identifying applications (e.g. browsers) installed by the user, mounted devices (e.g. shared network drives used for sharing files on social media). See Appendix A.1 for extracts.

5.2.3 Component: app_config

This component is made up of individual parsers for the registry hives. It is important to note that the SAM and SECURITY hives are locked and inaccessible when the

device is powered on. This is because Windows locks the file and prevents read/write activity from any accounts other than the SYSTEM. To address this limitation, the parsers used the Python subprocess module to call the *reg.exe save hklm** command (e.g. *reg.exe save hklm\sam*). This was used to dump the SAM and SECURITY hives. The HARDWARE hive was also dumped to grab the current state of the system hardware information as this hive is volatile and is lost when the device is powered off.

The winreg module was used to access the Windows registry API in read-only mode to minimise any possible impact on the target device. See Appendix A.2 for extracts.

5.2.4 Component: system_info

This component was used for the recovery of system generated artefacts from Prefetch and Event Logs. This was used to provide corroborating information on application in-

stallation and access times; and also to provide context to user activity.

This component was used to demonstrate the ability to use third party command-line interface (CLI) tools within SMURF. For example:

- ⇒ PECmd (Zimmerman, 2019) is a CLI tool that was used to process the Prefetch files and return the contents in JSON format.
- ⇒ python-evt (Ballenthin, 2019), a Python parser for Windows event logs (post-Windows XP), was used to process the Application and Security event logs. The output was saved as an XML file and a function was written to parse the XML file to display the content in an “easy to digest” format in the report.
- ⇒ SysInternals tools (Rusinovich, 2016, 2018) were used to recover additional/-corroborating system related and generated artefacts such as processes, registered owner, system uptime etc.

The subprocess module was used to call these third party tools from the *system_info* component. See Appendix A.3 for extracts.

5.2.5 Component: session_info

This component was used to recover artefacts from the browser session files. For this paper, two types of session related artefacts were recovered. Extracts can be found in Appendix A.4.

- I. sessionstore.mozlz4: this is used by Firefox to store session data. Artefacts from this file were recovered using a Python script derived from (Blumenbach, 2015)’s mozlz4a script.
- II. Current/Last Session; Current/Last Tabs: these are used by Google Chrome, and as implied by the file names, the window and tab activities of the user are stored here.

When Chrome is running, these files are

locked by the host and are inaccessible, so it was necessary to use the psutil module to programmatically terminate chrome.exe. The output was saved in text files for subsequent processing using RegEx.

5.2.6 Component: browser_history

This component was used to recover artefacts related to user browser activity. It is also designed to be extensible, allowing for the recovery of artefacts from multiple browsers and for session information to be fed into the framework to support browser history artefacts. It calls third party tools such as *chromagnon* (Bancel, Jean-Rémy, 2015) which was used to parse the Chrome History file, and to facilitate the recovery of session artefacts.

For the purposes of this paper, artefacts from Firefox and Chrome were analysed. See Appendix A.5 for extracts.

5.2.7 Component: browser_cache

This component was used to recover data in the browser cache that can be used for the corroboration of other browser artefacts.

The manual analysis conducted by David et al. (2020), observed and highlighted that content may be cached independent of user interaction. The cache returns a sizeable volume of data thus, for a live triage use case, this component focuses on the recovery of URL related cached content to reduce the amount of time and effort required to process cache information.

This component also demonstrates the extensibility and flexibility of SMURF. It calls third party tools such as *Hindsight* by Obsidian Forensics (Benson, 2019), which was used to facilitate the parsing of the browser cache artefacts. See Appendix A.6 for extracts.

5.2.8 Component: smurf_main

This is the main component of the framework. It was used to collate, manage the data ingested from the artefacts from the other components.

It was designed to format the data recovered and to generate a HTML report that was used to display the results in an ‘easy to assimilate’ format for the investigators (see Appendix A.7 for the HTML template extracts).

Extracts from smurf_main.py

```
# SMURF Main

def smurf():
    # read content of output files from framework components
    # write data of interest to SMURF Report.html

    # Content to be published
    title = "Triage Report"
    devices = "SMURF deployed from: "

[...]
```

```
# Rename existing template and load the new report template
sys.stdout.write("Creating report template... \n")
suffix = "smurf_report/" + str(dt.datetime.now().strftime("%Y%m%d-%H%M%S")) + "-smurf_report.html"

html_template = "smurf_report/report_template.html"
if os.path.exists(html_template):
    os.rename("smurf_report/smurf_report.html", suffix)
    shutil.copy2(html_template, "smurf_report/smurf_report.html")

[...]
```

```
with open(url_feats, 'r') as uf:
    twitter_url = []
    for line in uf:
        twitter_feats = uf.readlines()
        for item in twitter_feats:
            if "twitter" in item:
                url_data = twitter_url.append(item)
                twitter_url_data = twitter_url[-50:] # negative indexing, enumerate from tail
            else:
                if len(twitter_feats) == 0:
                    url_data = twitter_url.append("No Twitter URLs found...")
                twitter_url_data = twitter_url[:1]

[...]
```

5.3 Method of Deployment

Being a collection of components, some of which call third party tools/utilities, a number of options were considered as possible methods of deployment. This was done to ensure:

- ⇒ good ethical conduct and compliance with the legal use of the third party tools (i.e. adherence to potential licence ramifications or legal implications);

⇒ respect for the authors and their licences by using the tool in the way author consent implies.

Some of the utilities explored as a method of deployment can be used to bundle a python script and its dependencies into an executable file; however these were deemed impractical for the purposes of demonstrating the extensibility and adaptability of SMURF and ensuring that third party tools

are used as intended as mentioned above.

WinPython (WinPython, 2019) a free, portable scientific distribution of Python for Windows was deployed to a USB device and used to test the PoC in a live Windows 10 VM.

6. CASE STUDY: SMURF ON TWITTER

This section discusses the framework testing process, using Twitter as a case study. The results discussed in this section are grouped based on potential user activity. The weight-

ing scale used to determine the feasibility of SMURF is also discussed in this section.

6.1 Twitter as a Case Study

Twitter is a very popular, real-time information network with a global user base. It has been described as “a service for friends, family and coworkers to communicate and stay connected through the exchange of quick, frequent messages. People post Tweets, which may contain photos, videos, links and text. These messages are posted to your profile, sent to your followers, and are searchable on Twitter search” (Twitter Help Center, 2019).

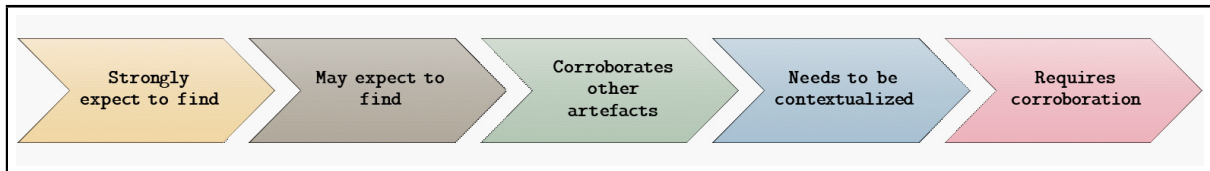


Figure 3. An illustration of the weighting scale

Twitter in its FAQs for new users, provides a description of how the platform can be used. This information was used as a guidance to extrapolate the basic activities that can be carried out on Twitter. These activities were placed into categories and the weighting scale shown in Figure 3 was derived from the analysis conducted in David et al. (2020).

In the context of a live triage investigation, the following is proposed:

- ⇒ **Strongly expect to find:** these are a range of artefacts that should generally be available to the investigator e.g. username, email address.
- ⇒ **May expect to find:** these are a range of artefacts that may often be available but not always e.g. login/logout activity, profile IDs of other users.
- ⇒ **Corroborates other artefacts:** these are artefacts that validates previously recovered artefacts e.g. search param-

eters in a URL.

- ⇒ **Needs to be contextualized:** this refers to any artefact that could be interpreted to mean different things and thus needs to be put in context of the previously recovered artefacts to verify its relevance to the case e.g. the user’s email address, email addresses of other users.
- ⇒ **Requires corroboration:** this refers to any artefact that could mean something else and cannot be validated on its own. It needs to be supported by existing artefacts e.g. communication content i.e. text extracts, images.

It is important to note that there may be overlaps when weighting the availability and relevance of artefacts recovered during a live triage investigation. For example, an investigator **may expect** to find the username of another user but would need to **corroborate** that information and put it in **context**

to determine if it occurred as a result of direct user interaction or background processes on the social media platform.

6.1.1 Registration/Authentication Activities

Users can access Twitter in a ‘read-only’ mode if they are not registered or logged in i.e. they are able to view content but can-

not respond or interact with other users. In order to get the full benefits of the platform, a registered account is required.

Creating an account allows a user to personalise the account by amending the account settings to allow or restrict the public from viewing tweets; and adding profile photos and banners.

User Registration/Authentication		Weighting			
User artefact		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised
	Email address	✓			✓
	Email addresses of other users			✓	✓
	Username	✓			
	User ID		✓		
	Username of other users		✓		
	User ID other users		✓		
	Profile name	✓			
	Profile ID		✓		✓
	Profile name of other users		✓	✓	
	Profile ID of other users		✓	✓	
	Search/Query parameters		✓		
	Login	✓		✓	
	Logout	✓		✓	
	Communication content		✓		✓

Figure 4. User registration/authentication artefact weighting

Figure 4 shows the range of artefacts associated with user registration/authentication on Twitter as recovered by SMURF as part of the first stage of the framework testing. The results indicate that there is a strong possibility of recovering artefacts such as the email address that was used for account registration.

Subsequent to that, URLs indicating login/logout activity can also be recovered and used to infer user account authentication activity.

There are other artefacts that may be used to support account activity. These include finding the usernames, profile names, user/profile IDs for other users. This may occur during the registration process where Twitter suggests people the user may like to

follow.

In the context of user activity, to infer/confirm possible account ownership, ‘**strongly expect to find**’ provides stronger evidence to validate the assumption. Table 1 shows the possible locations of these types of artefacts from Firefox and Google Chrome.

Table 1. Possible locations of registration/authentication artefacts

Chrome	Firefox
Last/Current Tabs	formhistory.sqlite
Last/Current Session	places.sqlite
Login Data	sessionstore.mozlz4
Web Data	login.json

6.1.2 Search and View Contents

As mentioned in Section 6.1.1, it is possible for users who are not logged in or registered to search for and view content in Twitter. This may be through typing in a Twitter URL with the *@handle* of a Twitter user e.g. `twitter.com/joebloggs` or through a search engine e.g. “joe bloggs twitter”.

The second stage of the framework testing involved the identification and recovery of artefacts that can be used to infer ‘search and view’ activities. This applies to both registered and unregistered users; the main difference being the availability of artefacts indicating that the user has a Twitter account for example, the presence of the artefacts described in Section 6.1.1.

Search and View Content		Weighting				
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised	Requires corroboration
User artefact	Email address		✓			
	Email addresses of other users	✓			✓	
	Username		✓			
	User ID	✓				
	Username of other users	✓		✓	✓	
	User ID other users	✓				
	Profile name		✓	✓		
	Profile ID		✓	✓		
	Profile name of other users	✓		✓	✓	
	Profile ID of other users	✓		✓	✓	
	File name		✓		✓	✓
	File ID		✓		✓	✓
	Search/Query parameters		✓	✓	✓	
	Login		✓			
	Logout		✓			
	Communication content	✓		✓	✓	✓

Figure 5. User search and view artefact weighting

Using the weighting matrix, the PoC was used to recover artefacts such as user/profile names and IDs in the ‘**strongly expect to find**’ category. Other artefacts that can be used to build context were recovered in the ‘**may expect to find**’ category. For example, where a Twitter user profile has been searched for by email address; the search parameters (**may expect to find**) can be used to contextualise the profile ID (**strongly expect to find**) and thus the profile visit/view.

Regular Expressions (RegEx) were used at this stage to parse the output files from the framework components to extract artefacts inferring a range of search and view activities that occurred on Twitter during the experi-

ments. Table 2 shows the possible locations of these types of artefacts from Firefox and Google Chrome.

Table 2. Possible locations of artefacts inferring search and view activity

Chrome	Firefox
Last/Current Tabs	places.sqlite
Last/Current Session	sessionstore.mozlz4
Cache	Cache
Cookies	Cookies

6.1.3 Share Content

Twitter is used for content dissemination through “Tweeting” or “Retweeting” content - images, videos, text, and/or audio. As part of content sharing, users may

also choose to communicate privately by exchanging direct messages (DMs).

Content exchanged on Twitter (tweets, retweets, DMs) may be accompanied by hashtags to highlight trending i.e. currently popular topics; the keyword(s) referencing the topic is prefixed with the hash symbol (#) for example, #fitness, #freebies, #LatestNews.

Social media communication in the con-

text of this paper has been described in Section 2.1 as the messages exchanged between two (or more) users, which may contain text, audio, video or image files.

As discussed in Section 2.1.1 some of the aforementioned may not be readily available or accessible on disk during a live triage investigation thus, it may be necessary to use other artefacts to build a picture of the user's activity.

Share content		Weighting				
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised	Requires corroboration
User artefact	Email address	✓				
	Email addresses of other users	✓	✓		✓	✓
	Username	✓				
	User ID	✓			✓	✓
	Username of other users	✓			✓	✓
	User ID other users	✓			✓	✓
	Profile name	✓			✓	✓
	Profile ID	✓			✓	✓
	Profile name of other users	✓			✓	✓
	Profile ID of other users	✓			✓	✓
	File name		✓	✓	✓	✓
	File ID		✓	✓	✓	✓
	Search/Query parameters		✓	✓	✓	
	Login	✓				
	Logout	✓				
	Communication content	✓	✓	✓		✓

Figure 6. User shared content artefact weighting

SMURF was used to search for evidence of these activities to determine the reliability of the weighting matrix applied to the manual analysis of the artefacts presented in David et al. (2020).

Figure 6 shows the range of user artefacts that may be used to infer content sharing on Twitter. For example, user/profile names and IDs, in addition to URLs parameters such as file names can be used in conjunction with page titles and login/logout URLs to infer a user was logged and possibly shared the file.

The framework did not focus on or attempt the recovery of the exact contents of communication as the goal of a live triage

is the timely and efficient provision of reliable intelligence, for the intended audience (e.g. investigators), in a usable and easy to understand format. However, the recovered artefacts will need to be corroborated.

It is important to note that non-registered users cannot share but are able to view publicly available content (Section 6.1.2) so **context** is necessary in order to draw inferences.

Table 3 shows the possible locations of these types of artefacts from Firefox and Google Chrome.

Table 3. Possible locations of artefacts inferring content sharing activity

Chrome	Firefox
Last/Current Tabs	places.sqlite
Last/Current Session	sessionstore.mozlz4
Cache	Cache

6.1.4 Associations/Relationships

On Twitter, users can choose to subscribe and be notified of another user’s tweets. This

is described as “following” (Twitter Help Center, 2019) and is designed as a two-way process where users follow each other. However, there are instances where one user does not reciprocate for example, a corporate entity would have several, perhaps thousands of followers but will not follow all its followers.

Followership may be used to determine a user’s interests, infer associations and relationships.

Association/Relationship		Weighting			
		Strongly expect to find	May expect to find	Corroborates other artefacts	Needs to be contextualised
User artefact	Email address	✓			
	Email addresses of other users		✓		✓
	Username	✓			
	User ID	✓			
	Username of other users	✓		✓	✓
	User ID other users	✓		✓	✓
	Profile name	✓		✓	✓
	Profile ID	✓		✓	✓
	Profile name of other users	✓		✓	✓
	Profile ID of other users	✓		✓	✓
	File name		✓	✓	✓
	File ID		✓	✓	✓
	Search/Query parameters		✓	✓	
	Login		✓		
	Logout		✓		
	Communication content	✓		✓	✓

Figure 7. User association/relationship attribution

Figure 7 shows a range of artefacts recovered by SMURF and their weighting. These artefacts can be used to draw inferences about a user’s associations and by extension relationships on Twitter.

In order to determine the relationship(s) between users on Twitter, it is important to put the evidence inferring contact in context. This is because it is possible to form associations without a direct relationship. Examples include but are not limited to where a user has connections with or follows Government or Commercial entities with a Twitter profile; following celebrity fan pages or profiles or following other users with similar in-

terests (photography, hiking etc.).

It may also be possible to use Twitter search and query parameters to aid the attribution of associations and relationships to a user. This may be useful in highlighting where particular user profiles were searched for and then added as a social connection.

Table 4 shows the possible locations of these types of artefacts from Firefox and Google Chrome.

Table 4. Possible locations of artefacts inferring associations/relationships

Chrome	Firefox
Last/Current Tabs	formhistory.sqlite
Last/Current Session	places.sqlite
Login Data	sessionstore.mozlz4
Cache	Cache

6.2 SMURF: Assessing Artefact Reliability with AUTOPSY[®]

As part of assessing the reliability of the artefacts recovered by the framework, and to highlight its contributions to the digital forensics community, SMURF was tested against Autopsy[®] 4.15.0 (Carrier, 2020); an open source, peer reviewed, in-use digital forensics tool.

Autopsy[®] is a digital forensics platform and graphical interface to The Sleuth Kit[®] and other digital forensics tools (Carrier,

2020). It has a broad range of features including “Communications Visualization” which provides a consolidated view of events of interest and allows the investigator to visualize communication based on commonly used accounts and the timeframe of an event of interest.

The process consists of the following activities:

- ⇒ Download Autopsy 4.15.0 from GitHub
- ⇒ Install Autopsy and create a new case
- ⇒ Add the experimental VM as a “Data Source”
- ⇒ Select and run relevant ingest modules (“Recent Activity”, “Keyword Search”)
- ⇒ Review results

A comparison of the artefacts inferring user activities as recovered by SMURF and the web browser activities recovered by Autopsy showed that SMURF was able to capture these activities as shown in Figures 8 and 9.

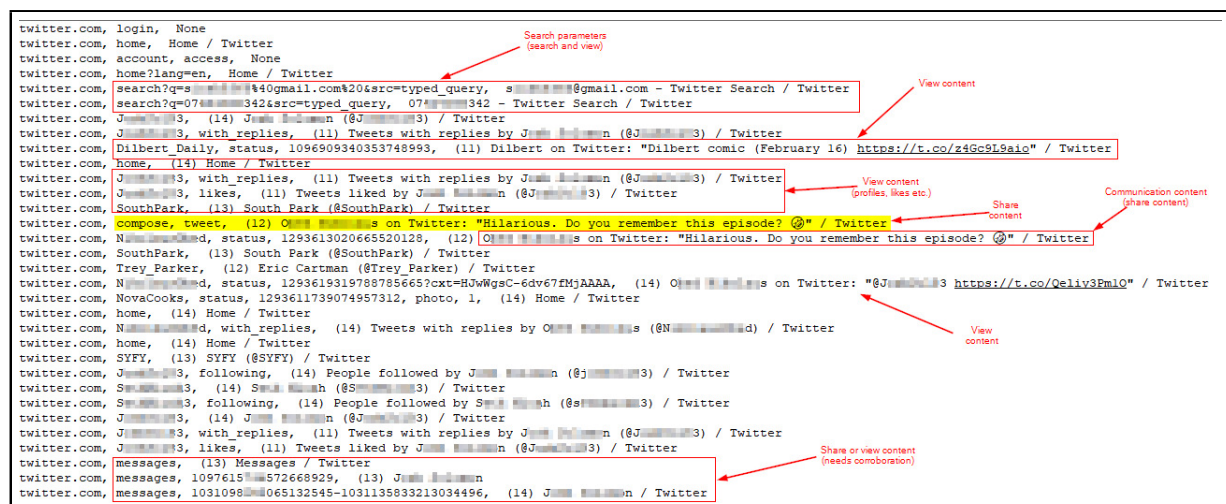


Figure 8. Social media URL artefacts recovered by SMURF

URL	Referrer URL
https://twitter.com/J[REDACTED]3/with_replies	https://twitter.com/J[REDACTED]3
https://twitter.com/J[REDACTED]3/with_replies	https://twitter.com/J[REDACTED]3
https://twitter.com/N[REDACTED]ed	https://twitter.com/home
https://twitter.com/N[REDACTED]ed	https://twitter.com/home
https://twitter.com/N[REDACTED]ed/following	https://twitter.com/N[REDACTED]ed/with_replies
https://twitter.com/N[REDACTED]ed/following	https://twitter.com/N[REDACTED]ed
https://twitter.com/N[REDACTED]ed/moments	https://twitter.com/home
https://twitter.com/N[REDACTED]ed/status/1293613020665520128	https://twitter.com/SouthPark
https://twitter.com/N[REDACTED]ed/status/1293619319788785665?cxt=HJwWgsC-6dv67fMjAAAA	https://twitter.com/home
https://twitter.com/N[REDACTED]ed/with_replies	https://twitter.com/N[REDACTED]ed
https://twitter.com/NovaCooks/status/1293611739074957312/photo/1	https://twitter.com/home
https://twitter.com/SYFY	https://twitter.com/home
https://twitter.com/S[REDACTED]h3	https://twitter.com/J[REDACTED]3/following
https://twitter.com/S[REDACTED]h3/following	https://twitter.com/S[REDACTED]h3
https://twitter.com/SouthPark	https://twitter.com/J[REDACTED]3/likes
https://twitter.com/Trey_Parker	https://twitter.com/SouthPark

Figure 9. Social media URL artefacts recovered by Autopsy

An attempt was made to visualise social media activities through the “Communications Visualization” feature, based on the Twitter URLs recovered by Autopsy however, it appears that web history visualisation is currently not supported although it works well with accounts found in the Data Source e.g. email accounts (PST, EML, MBOX), call logs and accounts from mobile devices. These were out of scope and not explored further.

In May 2020, Brian Carrier through Basis Technology (Basis Technology, 2020) offered free Autopsy training. The feedback from the digital forensics community evidenced the value of the tool. It was suggested that users can leverage the extensibility of Autopsy by adding plugins to the tool.

This research has identified the possibility of enhancing the “Communications Visualization” feature in Autopsy to allow the identification and inclusion of social media artefacts from web browsers as discussed in Section 8.

7. DISCUSSION

This paper was focused on developing and demonstrating a framework that can be used for the recovery of artefacts from social media activity during a live triage investigation and Twitter has been used as a case study to demonstrate the feasibility of deploying SMURF as a live triage investigative tool, as shown by the results presented in this paper.

Communication was defined in Section 2, providing context to the type of artefacts that can be created and their recoverability. This was used as a means of defining what ‘communication content’ is in the context of this paper; thus creating a relationship between an action and the resultant artefact.

A baseline for defining the framework functionality based on perceived user activity and artefacts created was introduced as it was necessary to establish constraints for the purposes of testing the PoC.

Using the weighting matrix described in Section 6, this paper has highlighted some possible use cases for Twitter activities and

the type of artefacts that can be recovered and has shown that context is important when attributing actions to a user. For example, distinguishing use cases where:

- ⇒ a user logs in and interacts with others
- ⇒ a user has an account but doesn't login (at least not on the device being investigated)
- ⇒ a user doesn't have an account but frequents Twitter

Although the dataset used in the creation of the framework can be said to be a subset of real world user activity, this paper demonstrates that it is possible to recover a wider range of artefacts created as a result of social media activity. This was confirmed by the results from Autopsy showing that the same category of artefacts were recoverable in both Autopsy and SMURF.

While discussing context, when it comes to the availability or non-availability of communication content, this paper demonstrated the possibility of recovering enough information to allow inferences to be drawn about the communication (see Figure 8); thus, allowing the investigator to identify and prioritize areas for further analysis.

There are certain caveats to be aware of during the investigation of social media activity. For example, artefacts may exist as a result of background processes on the social media platform; a web browser's caching mechanism and/or web storage of content the user has not interacted with; hence the need for context and corroboration when assessing relevance and reliability.

It is important to highlight that evidence from live triage may not necessarily be court bound but may end up being used as supporting evidence; thus, it is important to adhere to the relevant jurisdictional guidelines for handling evidence in a live triage.

8. CONCLUSION AND FUTURE WORK

This paper has demonstrated a PoC for a Social Media User Relationship Framework (SMURF). It has demonstrated through the results recovered the possibility of using SMURF in a live triage, to obtain quick, actionable intelligence that can be used to guide or progress an investigation.

It has been shown that communication content can be inferred from other sources such as the page title. This is useful in a live triage where it may not be possible to conduct an extensive keyword search or file carving.

For attribution, a user account can be linked to specific searches enabling an investigator to build a timeline of contact and subsequent activity with a person of interest or a victim.

This paper has presented a PoC with the potential for improvement and the extensibility of its existing functions. A number of opportunities for future work have been identified. These include implementing an automated process for the categorisation of potential artefacts of interest for example the categories discussed in Section 6 and extending the current functionality for other triage investigations.

It is important to note that additional cache data such as image files were not recovered as part of the framework however, in the event that image files are required as part of a live triage investigation, further work would be required to implement the recovery of image and video files using a range of techniques including known file hashes and skin tone detection.

No machine learning (ML) or artificial intelligence (AI) techniques were implemented. This is an avenue for future work. With ML, the components can be trained to adapt to specific scenarios using pre-set

case information thus improving and extending the current functionality.

As mentioned in Section 6.2, SMURF (or elements of it) could be added as a plugin to Autopsy or other live triage tools. This will extend its current functionality and allow for the visualization of social media activity from browsers.

As this demonstration was carried out on Windows 10, there is the potential for testing SMURF on other operating systems and on a variety of web browsers.

REFERENCES

- 7safe. (2014). *The ACPO Good Practice Guide for Managers of e-Crime investigation* (Tech. Rep.). Retrieved from www.7safe.com
- ACPO. (2012, March). *Good Practice Guide for Digital Evidence*. Retrieved from <http://library.college.police.uk/docs/acpo/digital-evidence-2012.pdf> (Version: 5.0)
- Arshad, H., Jantan, A., & Omolara, E. (2019). Evidence collection and forensics on social networks: Research challenges and directions. *Digital Investigation*, 28, 126–138.
- Ballenthin, W. (2019). *python-evtx*. Retrieved from <https://github.com/williballenthin/python-evtx>
- Bancel, Jean-Rémy. (2015). *Chromagnon (SNSS Branch)*. GitHub. Retrieved from <https://github.com/JRBANCEL/Chromagnon/tree/SNSS> (Latest commit 2cbeeb1 on 28 Mar 2018)
- Bashir, M. S., & Khan, M. N. A. (2013). Triage in Live Digital Forensic Analysis. *The International Journal of Forensic Computer Science*, 1, 35–44. doi: 10.5769/J201301005
- Basis Technology. (2020). *Free Autopsy Training*. Retrieved from <https://www.autopsy.com/support/training/covid-19-free-autopsy-training/>
- Bello, M., & DiBlasio, N. (2013, sep). *Twitter: The new face of crime*. USA Today. Retrieved from <http://www.usatoday.com/story/news/nation/2013/09/29/twitter-crime-dark-side/2875745/>
- Benson, R. (2019). *Hindsight*. Retrieved from <https://github.com/obsidianforensics/hindsight>
- Blumenbach, T. (2015). *mozlz4a.py*. Retrieved from <https://gist.github.com/Tblue/62ff47bef7f894e92ed5>
- Cambridge University Press. (2019). *Cambridge Dictionary [Online]*. Retrieved from <http://dictionary.cambridge.org/>
- Cantrell, G., Dampier, D., Dandass, Y. S., Niu, N., & Bogen, C. (2012). Research toward a Partially-Automated, and Crime Specific Digital Triage Process Model. *Computer and Information Science*, 5(2). Retrieved from <http://www.ccsenet.org/cisURL:dx.doi.org/10.5539/cis.v5n2p29> doi: 10.5539/cis.v5n2p29
- Cantrell, G., & Dampier, D. A. (2012). Implementing the Automated Phases of the Partially-automated Digital Triage Process Model. *Journal of Digital Forensics, Security and Law*, 7(4). Retrieved from <https://commons.erau.edu/jdfsl/vol7/iss4/5/>
- Carrier, B. (2020). *Autopsy 4.15.0*. GitHub. Retrieved from <https://github.com/sleuthkit/autopsy/releases/>

- Casey, E. (2004, jan). Digital evidence and computer crime: Forensic Science, Computers and the Internet. *Elsevier Academic Press*, 215.
- Casey, E. (2011). Digital Evidence and Computer Crime, Forensic Science, Computers and the Internet. In (Third Edition ed., chap. 1: Foundations of Digital Forensics). Elsevier Inc.
- Casey, E. (2013). Triage in digital forensics. *Digital Investigation*, 10, 85–86.
- Cusack, B., & Son, J. (2012). Evidence Examination Tools for Social Networks. In *10th australian digital forensics conference* (pp. 33–40). SRI Security Research Institute, Edith Cowan University, Perth, Western Australia. doi: 10.4225/75/57b3afc1fb861
- Dance, F. E. X. (1967). *Towards a Theory of Human Communication (In Human Communication Theory: Original Essays)*. Holt, Rinehart and Winston, New York.
- David, A., Morris, S., & Appleby-Thomas, G. (2020). A Two-Stage Model for Social Network Investigations in Digital Forensics. *Journal of Digital Forensics, Security and Law*, 15(1). Retrieved from <https://commons.erau.edu/jdfsl/vol15/iss2/1>
- DFRWS. (2001). *A Road Map for Digital Forensic Research: DFRWS Technical Report* (Tech. Rep. No. DTR - T001-01). DFRWS: Digital Forensic Research Workshop.
- Garfinkel, S. L. (2013, feb). Digital media triage with bulk data analysis and bulk_extractor. *Computers & Security*, 32, 56–72. Retrieved from <https://www.sciencedirect.com/science/article/pii/S0167404812001472> doi: 10.1016/J.COSE.2012.09.011
- Gielen, M., & Bolzoni, D. (2014). *Prioritizing Computer Forensics Using Triage Techniques* (Tech. Rep.). Retrieved from https://essay.utwente.nl/65671/1/Gielen_MA_EWI.pdf
- Haroon, S., & Carter, H. (2010, mar). *Facebook security measures criticised after Ashleigh Hall murder*. The Guardian. Retrieved from <http://www.theguardian.com/uk/2010/mar/09/ukcrime-facebook>
- Hitchcock, B., Le-Khac, N.-A., & Scanlon, M. (2016). Tiered forensic methodology model for Digital Field Triage by non-digital evidence specialists. *Digital Investigation*, 16(Supplement), S75–S85. Retrieved from <http://dx.doi.org/10.1016/j.diin.2016.01.010> doi: 10.1016/j.diin.2016.01.010
- Hubert, K. (2014). *Evidence Collection From Social Media Sites*. SANS Institute Information Security Reading Room. Retrieved from <https://www.sans.org/reading-room/whitepapers/legal/evidence-collection-social-media-sites-35647>
- Jusas, V., Birvinskas, D., & Gahramanov, E. (2017, mar). Methods and Tools of Digital Triage in Forensic Context: Survey and Future Directions. *Multidisciplinary Digital Publishing Institute (MDPI)*, 9(4), 49. Retrieved from <http://www.mdpi.com/2073-8994/9/4/49> doi: 10.3390/sym9040049
- Littlejohn, S. W. (1992). *Theories of Human Communication*. Belmont, Calif.: Wadsworth Pub. Co.
- Marcella, A. J., & Menendez, D. (2007). *Cyber Forensics: A Field Manual for Collecting, Examining, and*

- Preserving Evidence of Computer Crime* (Second Edition ed.). NEW YORK: CRC PRESS - TAYLOR AND FRANCIS.
- McKemmish, R. (2008). When is Digital Evidence Forensically Sound? *Advances in Digital Forensics, IV*, 3–15.
- Montasari, R. (2016, jun). Formal Two Stage Triage Process Model (FTSTPM) for Digital Forensic Practice. *International Journal of Computer Science and Security (IJCSS)*, 10(2), 69–87. Retrieved from <https://pure.hud.ac.uk/en/publications/formal-two-stage-triage-process-model-ftstpm-for-digital-forensic>
- Moore, K. (2014, June). *Social media ‘at least half’ of calls passed to front-line police*. BBC News. Retrieved from <https://www.bbc.co.uk/news/uk-27949674>
- Mukasey, M. B., Sedgwick, J. L., & Hagy, D. W. (2008, April). *Electronic Crime Scene Investigation: A Guide for First Responders, Second Edition*. U.S. Department of Justice, Office of Justice Programs, National Institute of Justice. Retrieved from <https://www.ncjrs.gov/pdffiles1/nij/187736.pdf>
- Parsonage, H. (2009). *Computer Forensics Case Assessment and Triage*. (<http://computerforensics.parsonage.co.uk/triage/ComputerForensicsCaseAssessmentAndTriageDiscussionPaper.pdf>)
- Robertson-Steel, I. (2006). Evolution of triage systems. *Emergency medicine journal*, 23(2), 154–155. doi:10.1136/emj.2005.030270
- Rogers, M. K., Goldman, J., Mislán, R., Wedge, T., & Debrota, S. (2006). Computer Forensics Field Triage Process Model. *Journal of Digital Forensics, Security and Law*, 1(2). Retrieved from <https://commons.erau.edu/cgi/viewcontent.cgi?article=1004&context=jdfsl> doi:10.15394/jdfsl.2006.1004
- Roussev, V., & Quates, C. (2012). Content triage with similarity digests: The M57 case study. *Digital Investigation*, 9, S60–S68. doi:10.1016/j.diin.2012.05.012
- Russinovich, M. (2016). *PsTools Suite Windows Sysinternals — Microsoft Docs*. Retrieved from <https://docs.microsoft.com/en-us/sysinternals/downloads/pstools>
- Russinovich, M. (2018). *Process Monitor - Windows Sysinternals — Microsoft Docs*. Retrieved from <https://docs.microsoft.com/en-us/sysinternals/downloads/procmon>
- Select Committee on Communications. (2014, July). *CHAPTER 2: SOCIAL MEDIA AND THE LAW*. Retrieved from <https://publications.parliament.uk/pa/ld201415/ldselect/ldcomuni/37/3702.htm>
- Shaw, U., Das, D., & Mehdi, S. P. (2016). Social Network Forensics: Survey and Challenges. *International Journal of Computer Science and Information Security (IJCSIS)*, 14(11), 310–316.
- Sommer, P. (1999). Intrusion Detection Systems as Evidence. *Computer Networks*, 31(23–24), 2477–2487.
- TWGECSI. (2001). *Technical Working Group Electronic Crime Scene Investigation - Electronic Crime Scene Investigation: A Guide for First Responders*.
- Twitter Help Center. (2019). *New user FAQs*. Retrieved from <https://help.twitter.com/en/new-user-faq>

- Wiles, J., & Reyes, A. (2007). Incident Response: Live Forensics and Investigations. In (pp. 89–109). Syngress.
- WinPython. (2019). *winpython*. Retrieved from <https://github.com/winpython/winpython>
- Zeng, D., Chen, H., Lusch, R., & Li, S. (2010, Nov). Social Media Analytics and Intelligence. *IEEE Intelligent Systems*, 25(6), 13-16. doi: 10.1109/MIS.2010.151
- Zimmerman, E. (2019). *Prefetch Explorer Command Line - PECmd version 1.3.4.5*. Retrieved from <https://github.com/EricZimmerman/PECmd>

A.

A.1 Extracts from predefined_paths.py

```
def current_user_keys():
    # keys of interest in the current user hive
    # HKCU\Software\Microsoft\Windows\CurrentVersion
    # HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer
    # HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32
    # HKCU\Software\Microsoft\Windows\Shell\BagMRU
    # HKCU\Software\Clients

    comdlg_set = r"Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\CIDSizeMRU", \
        r"Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\
            OpenSavePidlMRU", \
        r"Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32\
            LastVisitedPidlMRU", \
        r"Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs", \
        r"Software\Microsoft\Windows\CurrentVersion\Explorer\RunMRU", \
        r"Software\Microsoft\Windows\CurrentVersion\Explorer\MountPoints2", \
        r"Software\Microsoft\Windows\CurrentVersion\Explorer\Taskband", \
        r"Software\Microsoft\Windows\CurrentVersion\Run", \
        r"Software\Microsoft\Windows\CurrentVersion\RunOnce", \
        r"Software\Microsoft\Windows\Shell\Bags\1\Desktop", \
        r"Software\Clients\StartMenuInternet", \
        r"Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist",

    return comdlg_set
```

A.2 Extracts from app_config.py

```
# predefined registry keys. These can be updated in 'predefined_hives.py'
prog_run = predefined_hives.software_keys()

def local_software():
    for path in prog_run:
        try:
            i = 0
            while True:
                get_keys = OpenKeyEx(HKEY_LOCAL_MACHINE, path, 0, KEY_READ | reg_view_flag)
                name, value, data_type = EnumValue(get_keys, i)
                [...]

                file_exists = pathlib.Path(sw_csv_file).is_file()

                with open(sw_csv_file, 'a+') as skv:
                    fields = ["path", "name", "value", "data_type"]
                    csvwriter = csv.DictWriter(skv, delimiter=',', lineterminator='\n',
                                                fieldnames=fields)

                    if not file_exists:
                        csvwriter.writeheader() # if file doesn't exist, write a header

                    skv.write("{0}, {1}, {2}, {3} ".format(path, name, value, data_type) +
                                '\n')

                i += 1
            [...]
        except:
```

A.3 Extracts from system_info.py

```
# Prefetch
pref_out_file = "output_files/"

# set %winroot% as the OS partition then join %winroot% to path the evtx logs
win_root = os.environ['WINDIR']
prefetch_dir = os.path.join(win_root, 'Prefetch\\')

def parse_pf():
    # if str(path).endswith(".pf"):
    [...]
    logging.info("Parsing {}".format(prefetch_dir))

    subprocess.call([r"PECmd\PECmd.exe", "-d", prefetch_dir, "--json", pref_out_file],
                    shell=True)
```

```
# System events
# set %winroot% as the OS partition may not be C: by default.
# join %winroot% to path the evtx logs
win_root = pathlib.Path(os.environ['WINDIR']).drive
evtx_dir = pathlib.Path(win_root).joinpath('\\Windows', 'System32', 'winevt', 'Logs')

def parse_logs():
    for each in evtx_dir.iterdir():
        i = str(each)

        e_logs = ["Application.evtx", "Security.evtx"]

        for log in e_logs:

            if i.endswith("Application.evtx"):
                evtx_file = i
                logging.info("Parsing {}: ".format(evtx_file))
                subprocess.run([r"python", r"third_party\bin\python_evtx\scripts\evtx_dump.py", evtx_file, ">>", app_out_file], shell=True)

            elif ...
            [...]
            else:
                continue

        subprocess.run([r"third_party\bin\sysinternals\psloglist64", "-accepteula", ">", process_loglist], shell=True)

    [...]
```

A.4 Extracts from session_info.py

```
# sessionstore
def sessionstore_info():
    try:
        import lz4.block as lz4
    except ImportError:
        import lz4

    for each_dir in profile_dir.iterdir():
        for i in each_dir.iterdir():
            i = str(i)

            if 'sessionstore' in i and 'lz4' in i:
                file_name = i

                get_data = open(file_name, "rb")
                read_bytes = get_data.read()
                get_sig = read_bytes[:8] # verify header
                if bytearray(get_sig).startswith(b"mozLz40\0"):
                    json_data = json.loads(lz4.decompress(read_bytes[8:]).decode("utf-8"))

                with open(json_data_sstore, 'a+') as jd:
                    jd.write(str(json_data))

                    for win in json_data.get("windows"):
                        for tab in win.get("tabs"):
                            i = int(tab.get("index")) - 1
                            urls = tab.get("entries")[i].get("url")
                            [...]
                            last_accessed = win.get("tabs")[i].get("lastAccessed")
                            tab_last_accessed = dt.datetime.utcnow().timestamp(
                                last_accessed//1000.0)

                    [...]
                [...]
```

```
# the path to profile folder is '\\AppData\\Local\\Google\\Chrome\\User Data\\Default'
# pathlib.Path.home() joins the user's home directory path to the profile path
chrome_dir = pathlib.Path.home().joinpath('AppData', 'Local', 'Google', 'Chrome',
                                           'User Data', 'Default')

user_profile_dir = pathlib.Path(chrome_dir).glob('*')

def chrome_data():
    # Check if chrome.exe is running and terminate process before parsing
    # Chrome places a lock on the browser files when it's running
    for proc in psutil.process_iter():
        try:
            # process_name = proc.name()
            # check if chrome.exe is running
            if 'chrome.exe' in proc.name():
                proc.kill()
        except (psutil.NoSuchProcess, psutil.AccessDenied, psutil.ZombieProcess):
            continue

    time.sleep(5)

    # Parse current session
    c_file = str(chrome_dir) + r"\Current Session"
    logging.info("Parsing: {}".format(c_file))
    subprocess.run(["python", r"chromagnonSession.py", c_file, ">", current_session],
                   shell=True)
    [...]
```

A.5 Extracts from browser_history.py

```
# Firefox History
# the path to profile folder is '\\AppData\\Roaming\\Mozilla\\Firefox\\Profiles\\'
# pathlib.Path.home() joins the user's home directory path to the profile path
home_dir = pathlib.Path.home().joinpath('AppData', 'Roaming', 'Mozilla', 'Firefox', 'Profiles')

profile_dir = pathlib.Path(home_dir)

def user_places():
    for each_dir in profile_dir.iterdir():
        for i in each_dir.iterdir():
            i = str(i)

            if 'sqlite' in i:
                file_name = i

                # establish a connection to the database
                conn = sq3.connect(file_name)
                cursor = conn.cursor()

                # query for places.sqlite
                sql = "SELECT datetime(visit_date/1000000, 'unixepoch') as visitDate, url,
                        title " \
                    "FROM moz_places, moz_historyvisits " \
                    "WHERE moz_places.id = moz_historyvisits.place_id " \
                    "ORDER BY visitDate "

[...]
```

```
with open(places_csv, 'a+', encoding='utf-8') as ud:
    fields = ["visit_datetime", "url", "title"]
    writer = csv.DictWriter(ud, delimiter=',', lineterminator='\n', fieldnames=fields)

    if not file_exists:
        # create file, write a header
        writer.writeheader()

        ud.write("{0}, {1}, {2}".format(each_record["visit_datetime"],
                                       each_record["url"],
                                       each_record["title"]) + '\n')

    with open(places_csv) as rd:
        readCSV = csv.reader(rd, delimiter=',')
        for row in readCSV:
            for item in row:
                ml = ['twitter', 'facebook', 'drive.google', 'docs.google']
                for j in ml:
                    if j in item:
                        patt = r'((?<=\/)[^:\/\s]*[?*\\w\\w]+)'
                        regexp = re.compile(patt, re.IGNORECASE)
                        matches = regexp.findall(item)

                        with open(parsed_url, 'a+') as pu:
                            pu.write("{0}, {1}".format(row[0], ', '.join(matches)) + '\n')
```

```

# Chrome History

# Parse history
h_file = str(chrome_dir) + r"\History"
logging.info("Parsing: {}".format(h_file))
subprocess.run(["python", r"chromagnonHistory.py", h_file, ">", chrome_hist_file],
               shell=True)

with open(chrome_hist_file, 'r') as ch:
    for line in ch:
        get_each_val = line.split(None, 3)
        cut_date_val = [i.split('.', 1)[0] for i in get_each_val[:-2]]

        ml = ['twitter', 'facebook', 'drive']
        for j in ml:
            if j in line:
                patt = r'((?<=\/)[^:\/\s]*[?*\w\w]+)'
                regexp = re.compile(patt, re.IGNORECASE)
                matches = regexp.findall(line)

                with open(chrome_parsed_url, 'a+') as pur:
                    pur.write("{} {}, {}".format(" ".join(cut_date_val), ' '.join(matches)) +
                             '\n')

```

A.6 Extracts from browser_cache.py

```

# the path to the Firefox cache2 profile folder is '\\AppData\\Local\\Mozilla\\Firefox\\
# Profiles\\'
# pathlib.Path.home() joins the user's home directory path to the profile path
cache_profile_dir = pathlib.Path.home().joinpath('AppData', 'Local', 'Mozilla', 'Firefox', '
Profiles')

cache_dir = pathlib.Path(cache_profile_dir)

# iterate through the cache profile directory
for each_dir in cache_dir.iterdir():
    for i in each_dir.iterdir():
        i = str(i)

        if i.endswith('cache2'):
            cache2_path = i

[...]

with open(cache_idx_out_file, 'a+') as cix:
    cix.write("Cache version: {}".format(cache_version) + '\n')
    cix.write("Index last written time: {}".format(unix_written_time) + '\n')
    [...]
    cix.write("Index_Hash is: {}".format(binascii.hexlify(index_hash).upper().decode("ascii"
    )) + '\n')

    cix.write(" " + '\n')

    count += 1
    more_records = file_size - cix.tell()

[...]

```

```
# write to text file
with open(cache_data_file, 'a+') as cdf:
    cdf.write("Parsing file: {0}".format(cache_entry_file) + '\n')
    cdf.write("Cache file is written in version: {0}".format(version) + '\n')
    cdf.write("Cache entry file size: {0}".format(file_size) + '\n')
    cdf.write("Start of cache metadata: {0}".format(metadata_start) + '\n')
    [...]
    cdf.write("Key size: {}".format(key_size) + '\n')
    cdf.write("Key (URL): {}".format(key) + '\n')
    cdf.write("Key hash (SHA1 hash of url): {}".format(key_hash) + '\n')
    cdf.write(" " + '\n')
[...]
```

```
# Google Chrome

[...]
# Parse cache
chrome_d = str(chrome_dir)
chrome_cache = str(chrome_dir) + r"\Cache"
print("Parsing: {}".format(chrome_cache))
logging.info("Parsing: {}".format(chrome_cache))
subprocess.run(["python", r"hindsight\hindsight.py", "-i", chrome_d, "-o", chrome_cache_file
                ,
                "-f", "jsonl"], shell=True)
```

A.7 Extracts from SMURF HTML Report template

```
# SMURF HTML Report template
<div class="panel-group" id="accordion">
  <div class="panel panel-default">
    <div class="panel-heading">
      <h4 class="panel-title">
        <a data-toggle="collapse" data-parent="#accordion" href="#collapse1">Device and System</a>
      </h4>
    </div>
    <div id="collapse1" class="panel-collapse collapse in">
      <div class="panel-body">
        <h5><strong>{{devices}} </strong></h5>
        <p><span style="font-size: xx-small; "><ul>
          {{tool_device}}
        </ul></span></p>
        <h5><strong>{{sy_info}} </strong></h5>
        <p><span style="font-size: xx-small; "><ul>
          {% for n in user_os_info %}
            <li>{{n}}</li>
          {% endfor %}
        </ul></span></p>
      </div>
    </div>
  </div>
</div>
```


D.2 Manuscript

David, Anne; Morris, Sarah; and Appleby-Thomas, Gareth (2020) “A Two-Stage Model for Social Network Investigations in Digital Forensics,” *Journal of Digital Forensics, Security and Law*: Vol. 15, Article 1.

DOI: <https://doi.org/10.15394/jdfsl.2020.1667>

Available at: <https://commons.erau.edu/jdfsl/vol15/iss2/1>

A TWO-STAGE MODEL FOR SOCIAL NETWORK INVESTIGATIONS IN DIGITAL FORENSICS

Anne David¹, Sarah Morris², Gareth Appleby-Thomas³

^{1,2}Centre for Electronic Warfare, Information, and Cyber, Cranfield University, Defence Academy Of The United Kingdom, Shrivenham, SN6 8LA, UK

³Centre for Defence Engineering

Cranfield University, Defence Academy Of The United Kingdom, Shrivenham, SN6 8LA, UK
{digitalinvestigation^{1,2,3}}@cranfield.ac.uk

This paper proposes a two-stage model for identifying and contextualizing features from artefacts created as a result of social networking activity. This technique can be useful in digital investigations and is based on understanding and the deconstruction of the processes that take place prior to, during and after user activity; this includes corroborating artefacts. Digital Investigations are becoming more complex due to factors such as, the volume of data to be examined; different data formats; a wide range of sources for digital evidence; the volatility of data and the limitations of some of the standard digital forensic tools. This paper highlights the need for an approach that enables digital investigators to prioritize social network artefacts to be further analysed; determine social connections in the context of an investigation e.g. a user's social relationships, how recovered artefacts came to be, and how they can successfully be used as evidence in court.

Keywords: digital evidence, digital forensics, social networking, relationship attribution

1. INTRODUCTION

The proliferation of social networking sites has improved ways in which users engage and form relationships with people of similar interests; going beyond the days of email communications to the use of social networking applications to share messages, photos and videos. It has also provided the opportunity for some users to perpetrate unlawful activities.

Social networking presents challenges to digital forensic investigations for example, content posted may not always be written to permanent storage media. In addition, communication content can be altered or deleted

after the fact. There is a need for digital forensic investigators to be able to recover such messages or other evidence which may be used to infer user activity and sufficiently attribute an action/actions to a user.

Evidence from social networking activity may be required in different types of criminal or corporate investigations. The type of evidence recovered helps the investigator obtain useful information that could:

- Guide the initial stages of an investigation e.g. determining if a based on the evidence recovered, suspect is worth investigating further.

- Generate new leads which may lead to the:
 - Identification of other persons, places or items of interest.
 - Identification of other potential sources of digital evidence to facilitate decision making.

1.1 Contribution

The key contribution of this paper is the proposal of a two-stage model for evidence recovery and investigations involving social networking activities. It aims to help investigators prioritize digital evidence and maximize efficiency where resources are limited by focusing on extracting meaningful information from social networking artefacts. It is focused on the prompt identification and interpretation of associated artefacts and is aimed at enabling the analyst to quickly determine whether to expand or narrow the scope of an investigation.

1.2 Paper Structure

The rest of this paper is structured as follows: related work is discussed in Section 2. Section 3 proposes a two-stage model for social networking investigation in digital forensics. Section 4 discusses the experimental and analysis methodologies for this work. The research results and the implementation of the proposed model is presented in Section 5. Finally, the conclusions and potential future work stemming from this research are presented in Section 6.

2. LITERATURE REVIEW

This section provides a background on related work in digital forensics and social networking investigations. It discusses the requirements for admissibility of digital evidence and the

evidential value of data generated as a result of user social networking activity.

2.1 Digital Evidence

Digital evidence can be described as any data that can be used to determine intent, culpability, how an event occurred, and the parties involved. It is useful in the investigation of a range of computer crimes and non-computer related crimes where evidence from a digital device may be used to link a suspect to an offence (Casey, 2011).

Casey defines digital evidence as “data stored or transmitted using a computer, that supports or refutes a theory on how an event occurred”. Digital evidence is crucial in digital investigation and thus must be acquired in a forensically sound manner (McKemmish, 2008) to ensure that its admissibility in a court of law.

ACPO (2012)’s definition of digital evidence encompasses a range of artefacts that can be found on digital devices for example system log files, application logs etc. Multiple devices with various artefacts, whilst ideal sources of digital evidence, present the challenge of “weeding out” information not directly relevant to the case. When time is of the essence, an investigator needs to be able to adequately identify devices that contain evidence pertinent to the case, and use the intelligence obtained from those devices to progress the investigation.

Although the processing of digital evidence varies across jurisdictions, there are a few requirements digital evidence needs to meet before it is deemed admissible in court (Casey, 2005, 2002; Murr, 2007; Sommer, 1999):

- Evidence submitted must be **relevant** to the case.
- The evidence must be **reliable**.
- The methods used to produce the evidence must be **repeatable** and should produce the same results when applied independently by a third party.

- The evidence must be **authentic** (genuine) and can be verified using hash values generated prior to and after imaging a device.
- The evidence must be **valid** and error free. In exceptional circumstances where evidence acquisition from an active device is required, the process must be accurately documented, and any alteration accounted for.
- The evidence must be **trustworthy** and believable beyond reasonable doubt.

In order to be admissible in court, evidence from social networking activity must satisfy these requirements.

2.2 Social Networking

Social networking has been defined as “the activity of sharing information and communicating with groups of people using the internet, especially through websites that are specially designed for this purpose” (Cambridge University Press, 2019). It enables users to connect with others and to form personal or business relationships.

In the context of digital evidence acquisition, social network activity provides a plethora of digital evidence to investigators. Artefacts from web browser history, cache, cookies etc. can be used to determine and infer a relationship between a user and a social network account or another user and may also be used to attribute an action to a user. This includes, but is not limited to, determining dates and times of access, usernames, session information etc.

In spite of the advantages presented by social networking applications (instant messaging, sharing personal events, micro blogging, personal or corporate marketing, advertising etc.), it has also been known to present a means for a small minority of users to engage in disagreeable or criminal activities (Bello & DiBlasio, 2013; Jonsson, 2011; Osborne, 2010; Richards, 2007; Rankin, 2010; Select Com-

mittee on Communications, 2014; BBC News, 2012; Moore, 2014; McGuire, 2019a)

Investigating a user’s (or suspect’s) social network activity may be required for several reasons such as the collection of evidence to be used in court for the prosecution of an offender or for use in disciplinary actions taken against employees who abuse corporate Acceptable Use Policy (AUP) (Taylor, Haggerty, Gresty, Almond, & Berry, 2014).

The Crown Prosecution Service (CPS) (2018) proceed to trial once they are satisfied with the evidence obtained during investigations involving social networking activity. However, there are no defined guidelines for digital forensic investigators with regards to prioritizing evidence collection and the management of artefacts related to social networking.

Taylor et al. (2014) suggests that although there are no specific guidelines for the forensic investigation of social networking applications, ACPO Guidelines can be used as a starting point for the investigation of offences committed through or with a social networking application. It can thus be inferred that the lack of defined guidelines often results in such investigations being broadly categorized under ‘web browser forensics’ (Cusack & Son, 2012) due to the nature of access on a computer (while comparable, access through mobile devices is not considered here as it is considered outside the scope of this paper). However, it is suggested that focus on features specific to social networks e.g. user IDs, profile IDs, etc. can also be used as a viable technique for evidence acquisition using methods tailored to web browser forensics.

Keyvanpour, Moradi, and Hasan-zadeh (2014) presents a three-phase framework for social network forensics however, the specifics and potential location of artefacts of interest and techniques for recovery were not discussed. Oh, Lee, and Lee (2011) proposed an integrated method for the collection and analysis

of web browser evidence where multiple web browsers have been used in the commission of an offence. This is based on the need to recover and utilize data created and stored on disk when a user accesses social networking sites using a web browser. It is important to note that due to the nature and flexibility of social networking applications, materials posted or shared can be later modified or deleted. In such situations, the service provider may be in a position to provide the evidence required to determine the author, when content was modified (or deleted) and reconstruct events. However, activities such as evidence retrieval from service providers is considered out of scope for this paper as most practitioners, for example in the UK, will not have access to the service provider hence the reliance on “dead disk” forensics which is more common and accessible.

Although at the time of writing there is a research gap in the use of social network artefacts as digital evidence (Powell & Haynes, 2019; Huber et al., 2011; Jang & Kwak, 2015; Zainudin, Merabti, & Llewellyn-Jones, 2011; Shaw, Das, & Mehdi, 2016; Arshad, Jantan, & Omolara, 2019; Taylor et al., 2014), there are a number of reported cases where evidence from social networking activity has successfully led

to prosecution (Wood, 2018; Agency, 2015; Press Association, 2014; Bowcott, Carter, & Clifton, 2011; Haroon & Carter, 2010; BBC, 2010).

In discussing evidence collection from social network activity, Arshad et al. (2019) grouped social network artefacts into four distinct classes, User, Activity, Network and Content:

- User: consists of user data such as profile information, name, email address, phone numbers etc.
- Activity: consists of a timeline of user actions logged by the service provider on the server side e.g. dates and time of activity, location information, source of post e.g. phone, tablet, third party app etc. These types of artefacts are created as a result of user actions on the social networking site for example, when a user posts a comment about the service at a restaurant, the service provider tags the comment with the date and time it’s posted (see Figure 1). The location may also be included if geolocation is enabled. The user is unable to directly modify these types of artefacts.

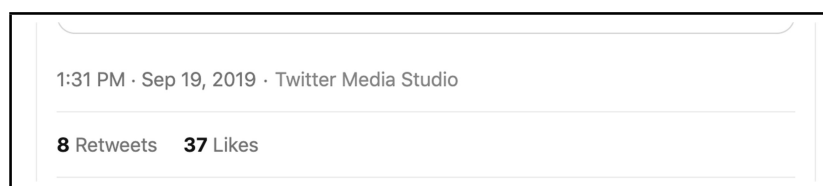


Figure 1: Activity time, date, source of tweet, number of retweets and likes illustrating server-side elements (source: <https://www.twitter.com>)

- Network: consists of personal social connections such as individuals or groups following or being followed by a user.
- Content: consists of materials published by a user such as photos, videos, tweets, retweets, shares etc.

Artefacts from each group can independently be used to infer user activity and when combined can be used to corroborate other related artefacts found on disk.

It is important to note that there is currently a knowledge gap with regards to formalizing the acquisition and analysis of social network artefacts. Some of the existing (traditional) digital forensic tools are not wholly designed for social networking investigations and the capability for targeted searches or the ability of the tool to interpret and present the evidence in a human readable format may be limited (Cusack & Son, 2012). For example, artefacts like Windows Registry Hives, Event logs, SQLite databases may need to be extracted and analyzed with a third-party tool. The objective of this paper is consequently to propose an approach that can be applied as a formal technique for social network investigations

2.3 Extracting Features from Social Networking Artefacts

Due to the proliferation of devices and the volume of data investigators need to process, it is often crucial to quickly identify content of interest prior to a detailed analysis of a seized device. Feature extraction is an approach that enables investigators to process vast quantities of data in an efficient manner (see Garfinkel (2006, 2013) for more on feature extraction).

A user's interaction with a digital device (computer) is a two-way process aptly explained by Locard's Exchange Principle (Chisum & Turvey, 2000) which states that every contact leaves a trace (Locard 1934, pp. 7-8 as cited in Chisum and Turvey (2007, pp. 23-24)). With regards to digital evidence, this principle can be adopted to explain the existence of artefacts created as a result of user activity. For example, creating a user account; installing an application such as a web

browser; or visiting social networking sites, all leave traces that can be used to infer what a user has done.

In the context of this paper, feature identification (and extraction) is described as the process of identifying and extracting artefacts containing key information about a user's social networking activities. Features in this context can be extracted from the absolute path of a given URL or other related artefacts such as HTML or JSON data, using pattern matching methods such as Regular Expressions (RegEx). The reoccurrence of a given feature can thus be attributed to a user's repeated access to a resource on a social networking site (Garfinkel, 2006).

2.3.1 Identify Features in URLs

Every website visited by a user has a URL which indicates where resources are located, and the protocol used in accessing those resources. RFC 1738 describes URL as a compact string representation for a resource available via the Internet (Berners-Lee, Masinter, & McCahill, 1994). It also describes the URL syntax as being made up of the following components:

`<scheme>:<scheme-specific-part>`

The `<scheme>` part of the syntax defines the protocol used e.g. `ftp`, `http`; while the `<scheme-specific-part>` varies and is dependent on the protocol used. For example, two typical protocols comprise of:

`ftp://user@host.domain/directory/filename`

or

`http://domain_name/path/query-searchpart-parameter`

`-fragment`

An example of a HTTP URL with three parts is shown in Figure 2. The query or searchpart of a URL may also be complex, having several parameters as seen in Figure 3. Some URLs may also specify subdomains or port numbers.

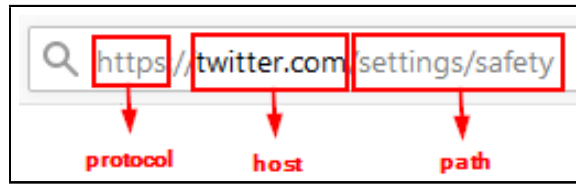


Figure 2: An example showing the parts of a HTTP URL



Figure 3: Example HTTP URL with multiple query or searchpart parameters

URLs can be generated in several ways for example, clicking on a link in an email or a web page; clicking a bookmark or a short-

cut; typing an address in the browser address bar, or using the autocomplete feature in the browser as shown in Figure 4.

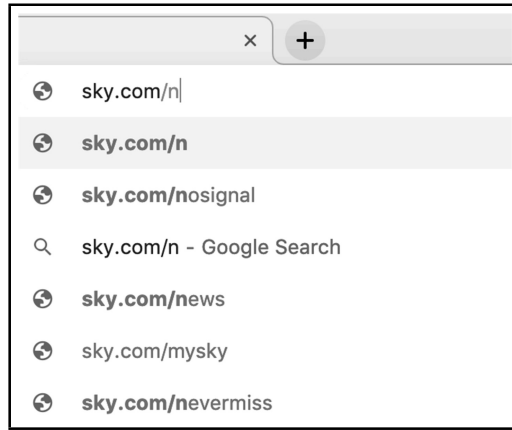


Figure 4: An example of the URL autocomplete/suggestion feature

URLs can provide a lot of information about what a user has been doing online – from pages accessed, to content searched for or shared. For this research, considering that URL structure is not browser dependent, identifying features in URLs involves decomposing the URL and decoding everything after the protocol part of the URL (i.e. <scheme-specific-part>) and in some instances, everything after the domain name part. This includes the path, parameters and their values and fragments.

For investigations involving social network activities, finding and recovering actionable intelligence/evidence will help narrow the scope and refocus an investigation, thus maximizing the use of the investigator’s time. Although there has been research in the forensic investigation of online activity and web browser artefacts, there is knowledge gap with regards to the deconstruction of individual URLs in relation to social networking activity. There is also limited research specifically focused on the modelling of social network forensic investigations processes and the extraction of

features from deconstructed URLs.

The research described in this paper highlights the need for a concise model for the investigation and analysis of such social media related artefacts. It also highlights the importance of features in understanding user intent and the attribution of actions to a user.

3. PROPOSED MODEL

Building on the identified need for a new approach to forensic analysis of social media data, there are two key areas to consider; understanding user activity and the types of artefacts which this can generate, as well as the links between the two.

3.1 Understanding “User Activity”

In general, there are three basic ways a user interacts with a computer:

1. Create/modify: This includes the creation of new files e.g. documents; modify-

ing existing files; installing applications, file upload or download.

2. Read/access: This includes accessing files with read, write, execute permissions. This type of interaction includes actions like launching a web browser (execute); generating an entry in the browser history (write); opening a folder or file (read).
3. Delete/remove files: This includes deliberate or unintentional removal of a file or application e.g. uninstall, delete.

In the context of this paper, user activity falls into three broad groups comprised of a mix of the basic interactions described above:

- Pre-browser activities:

- These set of activities create (or *modify*) artefacts that can be used to infer user activity. For example, powering on a device (computer) or a login would *create* an entry in the event log. Installing an application would *create* artefacts associated with setup and configuration. Launching the web browser would *read* (with execute permission), the executable file to start the browser; *create* a file, or an entry in the associated browser profile directory, which can be used to determine when a browser session was initiated.

- Browser activities:

- These set of activities *create* (or *modify*), read (*access*) artefacts that can be used in inferring user activity. For example, navigating to a website by typing a URL in the address bar; clicking on a website shortcut; selecting an autocomplete URL suggestion in browser

may *modify* the existing state of the browser history (e.g. the SQLite database) by creating new records based on the user’s activity.

- Activities such as accessing resources on a website, e.g. posts, uploads, downloads; would also *create/modify* entries in the session files, browser history, cache, etc. These activities also *read (access)* the browser cache to check that the associated content requested is in date and would *modify* the cache to update the content as required.

- Post-browser activities:

- These set of activities also perform *create* (or *modify*), *read (access, write* or *execute)* operations. For example, exiting/closing the web browser, may cause some of the contents in memory to be written to disk; logging off or powering off the device would *create* a corresponding entry in the event logs.
- In some instances, *delete (remove)* operations may occur where a browser is configured to “clear” content when the browser is closed. Delete operations as part of user activities are considered out of scope for this paper.

At each point during any form of interaction with a device (pre-browser, browser and post-browser), artefacts are created allowing inference of what occurred. In any investigation where there is more than one user account on the device, to prove an action was initiated by a user, it is important to identify and highlight artefacts (including corroborating evidence) that determine whether the user being investigated was at the keyboard of the

device in question within the timeframe of an incident.

3.2 Linking an Artefact to User Activity

Artefacts found on a digital device following the range of activities described in Section 3.1 can be broadly identified as system generated and user generated artefacts (Mabuto & Venter, 2012). This highlights the distinction between an artefact created by the system in the course of user activity and an artefact created as a result of direct input/interaction from a user as discussed below:

- System generated artefacts: these are artefacts created by the operating system (OS) or an application on the computer when core OS functions are performed by the OS or when a user performs tasks that require or trigger core OS services. These artefacts can be described as context artefacts because they can be used to support other artefacts recovered during an investigation and they also give context to the type of user activities that may have led to their creation. Examples include:
 - Event logs created during user login/logout activity (pre-browser; post-browser activities) could be used to infer a user had access and credentials for a device, and to connect a user to a timeframe of interest.
 - Setup or configuration artefacts created as local files or as records when an application or device driver is installed; including artefacts that are modified when a function is enabled/disabled by a user (pre-browser activity) could be used to provide context to the existence of an application (e.g. a web browser).
- User generated artefacts: these are artefacts created as a result of a user's direct interaction with an application. These artefacts can be described as case-specific artefacts and form the basis of inferences made about user behavior. They often create associated system generated artefacts which can be used to contextualize the activity. In the context of this paper, these refer to artefacts created during pre-browser, browser and post-browser activities. Examples include:
 - Installing a web browser (pre-browser activity) would create artefacts such as browser profile directories where the user's browser activities are stored. These artefacts can be attributed to user actions such as running an executable file to install the web browser. This can be linked to the system generated artefacts which are created as a result of this type of activity.
 - Accessing a website (browser activity) involves direct input by the user through typing in a URL or clicking on hyperlinks as discussed in Section 3.1. Artefacts indicating visits to websites can be attributed to a user launching a web browser and providing the input (URL or click) required to access the website(s).
 - File downloads from a visited website is another example of an artefact that could be attributed to user activity. The sequence of activity and user clicks on hyperlinks, as recorded in the browser history, and leading up to the file download may

The artefact (e.g. a Prefetch file, Registry entry) can thus be attributed to user activity (i.e. installing the web browser).

be used to corroborate the existence of a local copy.

- User activity may also create artefacts inferring the deletion/ removal of files; and the uninstallation of applications and drivers. These activities are also deemed out of scope for this paper.

3.2.1 Sources of Determination for Investigators

There are several ways an investigator can build a picture or reconstruct an event and attribute actions to a user based on the system and user generated artefacts recovered. These include but are not limited to using:

- Local Files: Created, modified, accessed dates and times can be determined when files resident on the device are analyzed.
- URLs: By deconstructing the URL, it is possible to determine what sites have been visited and what the user had typed in the address bar or search box e.g.: `search?q=statlerwaldorf\&src=typd`
- System setup or configuration logs: This infers when an application was installed; a file was created, number of times run or accessed, last time an application was run, or a file was accessed. It may also contain the path to Event logs.
- System Event logs: Logging is a way for the OS to record information about system activities that occur. This includes date and time of event; hostname/computer name of device involved; username

of who was logged in to the machine when the event occurred; the program that triggered the event etc. In Windows, an identification number is assigned to each event type (Ultimate IT Security, 2014).

Attribution in the context of this paper requires an approach that links a user (or user account) to the web activity being investigated. The two-stage model proposed in this paper can be used to achieve this. This model is intended to produce case-specific and context artefacts in a social networking investigation:

- i Case-specific artefacts: these can be described as artefacts that can be directly attributed to an action/activity of interest. For example, this feature “A” from the URL “Y” infers that the user clicked on the “X” tab on Twitter.
- ii Context artefacts: these can be described as artefacts that provide an explanation of how another artefact occurred. For example, clicking on a tab or link on a web page generates a new URL however, HTTP headers or the JSON artefacts for the browser will indicate if a link was clicked as well as if there is a referrer URL. Context artefacts may also include artefacts that are expected to occur as a result of a user’s activity on a social networking site. For example, the existence of photos from a social media profile in the cache could be attributed to a visit to that profile. This is discussed further in Section 5.1.2.

3.3 Two-Stage Model

Details of the proposed model are presented Figure 5.

Stage 1: URL feature extraction

The first stage of this model involves the identification and recovery of URLs from disk and the extraction of features from the URLs. The

URL in this instance is the main/core source of features for online activity. For example, the social network site visited, or the actions performed by the user (search, follow). It is

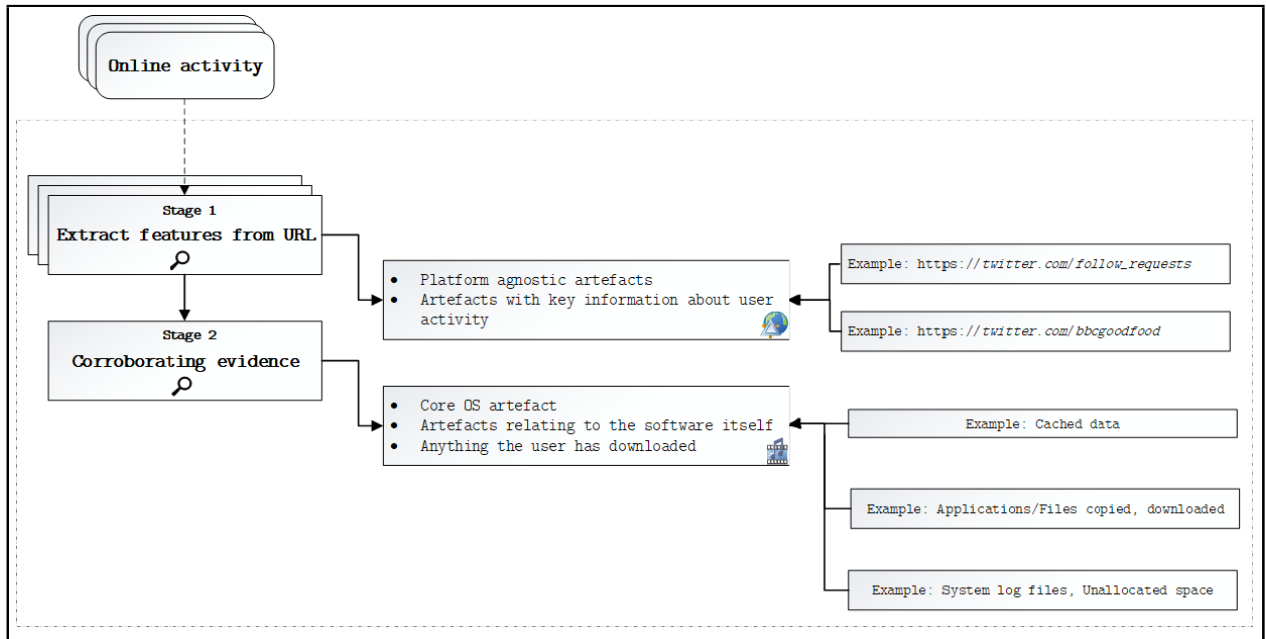


Figure 5: A schematic illustration of the proposed two-stage model for the investigations of social network activity

important to note that URLs are not platform dependent, so this approach can be applied to any platform (i.e. OS or browser).

Features are extracted using a combination of RegEx and the sqlite3 module in Python. Artefacts recovered are stored in CSV files containing the dates and times of activity, the full URL, and extracted feature(s) which can be used to infer user activity or allude to the user’s intent.

Stage 2: Corroborating evidence Corroborating artefacts validate each piece of evidence found during an investigation. In the context of this paper, corroborating artefacts provide both confirmation and supplementary information about the artefacts recovered during the URL feature extraction stage.

This stage of the model involves the identification and recovery of artefacts that validate what a URL feature indicates. These types of artefacts provide context to the features extracted from a URL. For example, the HTTP header information in the cache may show a URL that contains “/settings/account”

was created as a result of clicking on the “account” link in the page “settings” causing the browser to respond by updating the URL and rendering the requested content. In addition to information derived from HTTP headers found in the browser cache (or unallocated space), metadata from the web page HTML could be useful in understanding the user’s interaction with the social networking site.

This stage also involves the recovery of core OS artefacts that backup what has been inferred of the user’s activity. For example, downloaded files associated with the recovered URLs; a local copy of uploaded data (associated with a “www.domain-name/upload” URL); artefacts indicating that a downloaded application was installed and run “X” number of times including the physical path of the application; artefacts verifying application paths.

The proposed model is useful for both the recovery of actionable intelligence and for focusing and ensuring a structured investigation. Having a “URL feature extraction” stage takes

the bulk of URL artefacts and extracts meaningful information from them. This is useful because the digital forensic investigator needs a clear understanding of the URL structure in order to extract usable information from it. When artefacts from the “URL” stage have been extracted, corroborating (supplementary) artefacts are used to contextualize events and help digital evidence meet the requirement to be beyond reasonable doubt.

This paper presents work that improves on existing research on the forensic investigation of social network activities. It provides context by identifying and highlighting the importance of artefacts that corroborate or supplement the extracted feature(s). This includes data in URLs which ordinarily may be missed due to the volume of information returned by conventional digital forensics tools; data from HTTP headers and general browser artefacts.

4. RESEARCH METHODOLOGY

This section presents the data generation and analysis methodology for this paper.

4.1 Data Generation

The experiments for this paper were conducted in virtual environments running Windows 7 and Windows 10 respectively with Firefox browser installed via an executable in a network shared folder. Firefox version 61 was installed with rolling updates up to version 69.

The purpose of these experiments was to simulate real life activity on a social networking site and to create a feasible model for investigating such activities. The need to ensure the repeatability of the experiments made it necessary to use a virtual environment.

During the experimental phase, Fiddler (Telerik, 2018) was used to capture HTTP requests in order to understand how individual parts of a URL can be deconstructed; what

was sent to the web server and the response returned to the client (web browser); what was cached eventually irrespective of “no-cache” options in the header etc.

Data generation for this paper involved creating a local user account in the Windows virtual machine (VM) and creating a user account on Twitter using Firefox. Normal user activity was simulated by conducting a variety of the activities listed below over the course of the experiment:

- Power on the VM and log in
- Launch the web browser
- Login to the test user account on Twitter
- Searching for users to follow
- Sending tweets
- Reply and retweet
- Viewing and sending Direct Messages (DMs)
- Send follow requests to other test user accounts
- Viewing and accepting follow requests
- Updating the test user account privacy settings
- Continuous scrolling and viewing the test user account timeline

During the experiments, all activity was logged in a contemporaneous note as a means of verifying the user activity against the results found during the analysis.

4.2 Data Analysis

The analysis was conducted in a Windows 10 desktop environment using existing digital forensics tools. This was done to identify and understand artefacts of interest in a social networking investigation and to help with the implementation of the proposed model. The data analysis covered core OS artefacts and user generated artefacts, and a multi-tool analysis technique was employed. The following tools were used:

- General examination tool:
 - WinHex (X-Ways Software Technol-

ogy, AG, 2018): this was used to view the virtual disk contents and for the extraction of artefacts to be analyzed with third-party tools (listed below). It was also used for simultaneous keyword search. WinHex simultaneous search function allows a list of search terms (one per line) to be searched at once. It is important to note that a keyword search was run across the whole disk image and WinHex would have only picked up hits (data) that was visible to it; this limitation means that it would not have captured the contents of compound or encrypted files.

- Tools used for individual artefact analysis:
 - Registry Decoder (Case & Marziale, n.d.): this was used to analyze the registry hives.
 - RegRipper (Carvey, 2018): this was used to validate the results from Registry Decoder.
 - DB Browser for SQLite (DB4S Project, n.d.): used for the analysis of user browser SQLite databases in the user’s Firefox profile.
 - MZCacheView (NirSoft, 2018b): used for the analysis of the Firefox cache artefacts.
 - FullEventLogView (NirSoft, 2018a): used for the analysis of Windows event logs.
 - Prefetch Forensics (Woan, 2013): used for the analysis of Prefetch artefacts.
 - Python 3 (Python.org, 2019): python scripts were used to convert

the sessionstore files to JSON format and to extract features from the recovered URLs.

5. RESULTS AND DISCUSSION

This section discusses the experimental results, following the proposed two-stage model approach. These are categorized into URL feature artefacts and corroborating artefacts.

5.1 Implementation

Artefacts of interest in this research include user and system generated artefacts inferring user activity on Twitter. This includes URLs generated as a result of Twitter activity (login, searching for followers, viewing followers, tweeting etc.) and system generated artefacts that give context to other artefacts and a user’s activity.

5.1.1 URL Feature Artefacts

As discussed in previous sections, features from URLs can be used to give context to or infer user action. URL artefacts were recovered from the user’s Firefox Profile using DB Browser for SQLite. It is important to differentiate between the History location and the Cache location as both folders share the same name. In this section, when the Firefox Profile folder is mentioned, it refers to the folder containing the browser history. The Cache is discussed in Section 5.1.2.3.

The Firefox profile can be found in: `%APPDATA%\Mozilla\Firefox\Profiles;` where `%APPDATA%` is the variable name for `%SYSTEMROOT%\Users\<username>\AppData\Roaming`.

The profile folder contains a subdirectory with a *.default* extension (e.g. *oeds8ys7.default*) which contains the SQLite database files (shown in Figure 6), session information files, other files and directories used by Firefox.

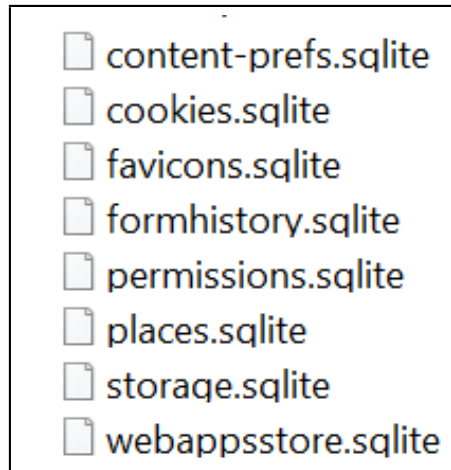


Figure 6: SQLite database files in the Firefox Profile folder

5.1.1.1 History

The browser history is written to places.sqlite and was analyzed using DB Browser for SQLite. The query was focused on URLs indicative of accessing Twitter. The returned URLs include login, timeline, search, profile views, and follow/follower activities.

Extracts of the results are shown in Figure 7 (although the experimental user is fictitious, certain parts of the URL have been modified to remove identifying information). The extracts show standard Twitter activity URLs as displayed in the browser address bar while navigating to different pages and as recorded by Fiddler during the experiments.

1	SELECT datetime(visit_date/1000000, "unixepoch"), url, title, rev_host, from_visit, visit_type, visit_count
2	FROM moz_places, moz_historyvisits
3	WHERE visit_date != ""
4	AND moz_places.url LIKE "%twitter%"
5	ORDER BY visit_date
6	

	datetime(visit_date/1000000, "unixepoch")	url	title	rev_host	from_visit	visit_type	visit_count
1	2018-12-01 18:18:12	http://twitter.com/	NULL	moc.rettiwt.	0	1	1
2	2018-12-01 18:18:12	https://twitter.com/	Twitter	moc.rettiwt.	0	1	6
3	2018-12-01 18:18:12	https://twitter.com/login	Login on Twitter	moc.rettiwt.	0	1	1
4	2018-12-01 18:18:12	https://twitter.com/i/notifications	Twitter / Notifications	moc.rettiwt.	0	1	1
5	2018-12-01 18:18:12	https://twitter.com/settings/account	Twitter / Settings	moc.rettiwt.	0	1	1
6	2018-12-01 18:18:12	https://twitter.com/settings/safety	Twitter / Settings	moc.rettiwt.	0	1	1
7	2018-12-01 18:18:12	https://twitter.com/notifications/followers	People following @N... (d) Twitter	moc.rettiwt.	0	1	3
8	2018-12-01 18:18:12	https://twitter.com/follower_requests	@N... (d) Twitter	moc.rettiwt.	0	1	1
9	2018-12-01 18:18:12	https://twitter.com/S... 3	S... h (@S... 3) Twitter	moc.rettiwt.	0	1	1

Result: 240 rows returned in 96ms
 At line 1:
 SELECT datetime(visit_date/1000000, "unixepoch"), url, title, rev_host, from_visit, visit_type, visit_count
 FROM moz_places, moz_historyvisits
 WHERE visit_date != ""
 AND moz_places.url LIKE "%twitter%"
 ORDER BY visit_date

Figure 7: URLs showing user activity (recovered from places.sqlite)

The results from DB Browser for SQLite were exported and saved in .csv format ready

for URL deconstruction.

Using the contemporaneous notes, the recovered URLs were grouped based on the user activities carried out during the experiments.

Tables 1 - 3 show examples of artefacts indicative of profile/timeline view, clicking the “followers” hyperlink on the user’s timeline and viewing follower requests, respectively.

Table 1: Twitter timeline URL and tab title

URL	Tab Title
<code>https://twitter.com/NxxxxxxxOxxx</code> (User’s home/timeline)	<code>Oxxx Nxxxxxxx (@Nxxxxxxxxxd) Twitter</code> (Tab title contains user’s full name and (@user’s handle))

Table 2: URL generated when the “followers” link is clicked

URL	Tab Title
<code>https://twitter.com/NxxxxxxxOxxx/followers</code> (Created when “followers” is clicked)	<code>People following Oxxx Nxxxxxxx (@Nxxxxxxxxxd)</code> (Tab title)

Table 3: URL generated when the “followers” link is clicked to view follow requests

URL	Tab Title
<code>https://twitter.com/NxxxxxxxOxxx/follower_requests</code> (“pending follower requests” is clicked.)	<code>Oxxx Nxxxxxxx (@Nxxxxxxxxxd) Twitter</code> (Tab title)

During the experiments, the account security and privacy settings were updated. This includes making the account private etc. The

user clicked on settings and was directed to the account page from where security and privacy settings can be modified. Tables 4 - 5 show URLs from this activity:

Table 4: URL indicative of account settings modification

URL
https://twitter.com/settings/account
<i>(This URL takes the user to the 'Account page' from where the user account settings can be modified.)</i>

Table 5: URL indicative of privacy settings modification

URL
https://twitter.com/settings/safety
<i>(When the user clicks on 'Privacy and safety' within settings, this URL is created. This page allows the user to set tweets as private, disable location tagging etc.)</i>

To attribute these artefacts to actions performed, it was necessary to review the HTTP request and response headers captured by Fiddler. This was done to determine if an unrelated activity could have caused the creation of these artefacts.

The next step was to break down the URLs

into manageable components, the csv and re modules in Python 3 were used as seen in the example code extract below. The features including the date and time of access were written to a csv file using a version of the code illustrated below:

```

1. import csv
2. import re
3.
4. # open the csv file
5. with open("history.csv") as hist:
6.     # read the csv file
7.     readCSV = csv.reader(hist, delimiter=',')
8.     for row in csv file:
9.         for item in row:
10.            if 'twitter' in item:
11.                pattern = your_regex_pattern
12.                regex_search = re.compile(pattern, re.IGNORECASE)
13.                matches = regexp.findall(item)
14.
15.            # write regex match to a csv file
16.            with open("matched_patterns.csv", 'a') as mp:
17.                mp.write("{0}, {1}".format(row[0], ','.join(matches)) + '\n')

```

The examples below are some of the saved feature matches; they are intended to help an analyst make sense of how an event happened. For example, the user accessed Twitter on date and time, then the user navigated to this

part of the page; the user searched for this other user, viewed their profile and status etc.

When sorted by the date and time of activity, it can be used to recreate a probable timeline of activity.

```

2016-01-17 21H:09M:49S, twitter.com, MissPiggy?ref_src=twsrc%5Egoogle
2018-12-01 18H:18M:12S, twitter.com, login
2018-12-01 18H:18M:13S, twitter.com, i, notifications
2018-12-01 18H:21M:14S, twitter.com, settings, account
2018-12-01 18H:21M:14S, twitter.com, settings, safety

```

```

2018-12-01 18H:21M:52S, twitter.com, Nxxxxxxxxxd, followers
2018-12-01 18H:21M:52S, twitter.com, follower_requests
2018-12-01 18H:21M:52S, twitter.com, Sxxxxxxh3
2018-12-01 18H:21M:52S, twitter.com, Nxxxxxxxxxd
2019-11-08 11H:46M:15S, twitter.com, Pxxxxxxxxd
2019-11-11 15H:14M:45S, twitter.com, DxxxxxKxxxx, with_replies
2019-12-24 11H:53M:39S, twitter.com, jxxxxhxxxxx, status, 544385844081987584

```

5.1.2 Corroborating Artefacts

Corroborating artefacts as discussed in Section 3.3 are artefacts that provide context to the URL artefacts recovered using the two-stage model proposed in this paper. This section discusses corroborating artefacts as they relate to the URL artefacts discussed in the preceding section.

5.1.2.1 Other SQLite Database Files

Cookies.sqlite records cookies set during a browsing session. It provided corroborating information for the features that were recovered from URLs in Section 5.1.1.1. Figure 8 shows a cookie set for the path “settings/safety”.

baseDomain	name	value	host	path
twitter.com	dnt	1	.twitter.com	/settings/safety/
twitter.com	fm	0	.twitter.com	/

Figure 8: Cookies set in cookies.sqlite indicates the setting/safety page was accessed on twitter.com

The cookie information can be used in conjunction with other artefacts such as login credentials shown in Figure 9, to determine and link the user account to the browsing session where the activity of interest occurred.

fieldname	value	datetime(lastUsed/1000000, "unixepoch")
session[username_or_email]	o[REDACTED]8@gmail.com	2018-12-01 18:35:01

Figure 9: Login credentials from formhistory.sqlite shows the last time the recorded email address was used

Formhistory.sqlite records entries made in form fields in Firefox. These are stored in *key:value* pairs where the fieldname is the key, and the entry typed into a text field is the value. Information recovered shows the login username for the user’s Twitter account and the last time the username was used. This can be used to infer user login activity with regards to an activity and time frame of interest.

Webappsstore.sqlite stores data for websites in *key:value* pairs. In this instance, the value of the *__typeahead__:userHash* key con-

tained information on the user’s Twitter account including that of over 500 other Twitter users. This amount of information can be overwhelming however, using features such as *@user_handle*, *user_id*, *profile_id*, extracted from the URLs, it can be filtered down to a manageable size.

Artefacts from webappsstore such as the values of *“followed_by”* and *“following”* within *“social_context”* as seen in Figure 10, can be used to infer a user’s social connection (relationship) to other users.

5.1.2.2 Session Information

Artefacts supporting follow activity identified during the URL extraction stage were recovered from Firefox sessionstore. Sessionstore (as at the time of writing) is stored in a compressed file format (MOZLZ4/JSONLZ4) in the browser profile folder and is used by Firefox to manage the ability to restore currently open windows and tabs in the event of a crash or forced restart. It can also be used to open previous tabs on startup following a clean exit. The following files are also used to store session data:

```
%APPDATA%\Mozilla\Firefox\Profiles\  
<profile-folder>\sessionstore  
-backups\  
• previous.jsonlz4
```

JSON artefacts such as the extracts shown in Figure 12 are useful in identifying the session a URL is part of, and any associated

- recovery.baklz4
- upgrade.jsonlz4-[timestamp] (session state before a browser version upgrade)

The information of interest in sessionstore include URL, page title, referrer URL; time a tab was last accessed or closed; the time a window was last accessed or closed; session start/last updated time; cookies associated with the session.

The sessionstore file was decompressed from LZ4 to JSON format, using the lz4 module in Python 3. When analyzed, it provided information on the user session where the account settings were accessed thus corroborating the activities inferred by the URLs (Section 5.1.1) as shown in Figures 11 - 14.

(referrer) URL.

This information can be used when reconstructing how a user arrived at a specific part of the website.

As observed during the experiments, when another user's profile is viewed, the URL displayed in the address bar contains the username or identifier for the visited profile.

Thus, the results can be used to infer that when a recovered URL contains a username

or identifier other than that of the logged in user, it indicates a visit to the other user's profile.

Figure 13 shows a visit to the profile of another user (URL) and the tweet sent by the experimental user.

In order to establish timelines, it is important to review the session within which the activity of interest occurred as shown in Fig-

ure 14. This would corroborate visit times indicated in the browser history and would also identify other activities that may have occurred within the timeframe of interest.

5.1.2.3 Cache

The Firefox cache can be found in: `%SYSTEMROOT%\Users\<username>\AppData\Local\Mozilla\Firefox\Profile`. The profile directory contains a subdirectory with a `.default` extension and an identical name to that of the history profile folder (see Section 5.1.1).

During the analysis of the cache, artefacts recovered were indicative of a user's direct interaction with other users for example, profile banners and profile photo URLs and images. These URLs validated the contents recovered from `webappsstore.sqlite` (Figure 10) and `sessionstore`. Extracts are shown below in Figures 15 and 16:

It was observed that some of the *http://pbs.twimg.com* URLs were for profiles the experimental user did not interact with. This behavior is somewhat expected, depending on the browser in use, as page elements are stored locally pending a refresh when the cached content is out of date as discussed in Section 3.1.

Just as observed with the webappsstore artefacts, the URLs can be grouped into a separate category after the features extracted in Sec-

tion 5.1.1.1 have been used to filter out the URLs of interest indicating social networking activity.

It is important to note that some of the URLs recovered from the cache were as a result of background processes on Twitter. For example, hashflags and hashtags.

Hashflags can be described as custom emojis that accompany a hashtag (e.g. **#StarWars** 🚫) and are used by Twitter to promote events. For example:

`https://abs.twimg.com/hashflags/Amazon_Holiday_2018/Amazon_Holiday_2018.png`
`https://abs.twimg.com/hashflags/WB_LegoMovie_Emmet/WB_LegoMovie_Emmet.png`

Hashflag URLs may be cached even though they have not been viewed or used by the user.

An example of a hashflag URL recovered from the cache is shown below:

7109699712	68 74 74 70 73 3A 2F 2F	61 62 73 2E 74 77 69 6D	<code>https://abs.twim</code>
7109699728	67 2E 63 6F 6D 2F 68 61	73 68 66 6C 61 67 73 2F	<code>g.com/hashflags/</code>
7109699744	57 42 5F 4C 65 67 6F 4D	6F 76 69 65 5F 45 6D 6D	<code>WB_LegoMovie_Emm</code>
7109699760	65 74 2F 57 42 5F 4C 65	67 6F 4D 6F 76 69 65 5F	<code>et/WB_LegoMovie_</code>
7109699776	45 6D 6D 65 74 2E 70 6E	67 00 E5 00 00 00 00 00	<code>Emmet.png à</code>

Hashtag URLs from the “Trends for you” frame on the user’s Twitter homepage were also recovered. The user also did not inter-

act with this part of Twitter or use hashtags during the experiment. Examples of hashtag URLs recovered from the cache are:

`https://twitter.com/hashtag/Disney?src=hash`
`https://twitter.com/hashtag/RoaldDahl?src=hash`
`https://twitter.com/hashtag/StarTrek?src=hash`
`https://twitter.com/hashtag/WrathOfKhan?src=hash`

An extract of the hashtag URLs recovered from the cache is shown below:

7096393600		68 74 74 70 73 3A 2F 2F	<code>https://</code>
7096393616	74 77 69 74 74 65 72 2E	63 6F 6D 2F 68 61 73 68	<code>twitter.com/hash</code>
7096393632	74 61 67 2F 57 72 61 74	68 4F 66 4B 68 61 6E 3F	<code>tag/WrathOfKhan?</code>
7096393648	73 72 63 3D 68 61 73 68	00	<code>src=hash</code>

Some of the recovered URLs were from suggested/promoted tweets advertised on the user’s timeline. It is important to note that these ads and sponsored content can be found

on disk even when a user hasn’t clicked on them. In the context of this paper, they were a result of continuous scrolling on the user timeline. These tweets can be identified by the example features highlighted below:

```
https://twitter.com/i/cards/tfw/v1/1056932085485658113?
advertiser_name=NespressoUK&Ireland&cardname=unified_card&
is_following_advertiser=false&forward=true&impression_id=358a
4182a9a96b66&edge=true&lang=en&card_height=271&scribe_context=
"client":"web","page":"home","section":"home","component":"tweet"&
bearer_token=AAAAAAAAAAAAAAAAAPYXBAAAAAACLXUNdekMxqa8h/4
OK4moUkGsoc=TYfbDKbT3jJPCEVnMYqilB28NHfOPqkca3qaAxGfsyKCs0wR
bw# xdm_e=https://twitter.com&xdm_c=default4701&xdm_p=1
```

xdm is cross-domain messaging and the expected value for *xdm_e* is the base URL of the host. As shown in the URL above, the base URL is *https://twitter.com*. To determine whether a user is following an advertiser, the value for “**is_following_advertiser**” would be “true”.

Figure 17 shows an extract of HTTP Request and Response headers captured through

Fiddler, which was used to monitor Twitter traffic during the experiments. Fiddler was used to determine what was expected to be written to disk, what eventually was written to disk, and what was not written to disk. The ad URLs captured by Fiddler corroborate the Twitter advertising URL recovered from the user’s Firefox cache. They show that the background processes on Twitter can be written to disk.

Due to the volume of cached data, and although they may not be used for user relationship attribution, it is important to identify and highlight social networking artefacts that are unrelated to direct user interaction. This would help an investigator/analyst focus on artefacts of immediate interest. For example, grouping sponsored content into different categories, separating them from normal user accounts.

5.1.2.4 Registry and Event Logs

The artefacts presented in this section come from a range of sources (system and user generated) as discussed in Section 3.2. They aim to answer questions such as the number of user accounts on the system; the logged in

user; dates and times of activity; applications installed or accessed; paths to files and applications etc.

Where there are multiple user accounts on a device, it is important to identify registered account(s) of interest, credentials, permissions and dates and time of access. These can be identified and recovered by analyzing the SAM registry hive and reviewing Windows Event logs.

The hive files (Table 6) were extracted using WinHex and information about the user account such as when the account was created, the username, the account type, application (browser) installation, and access were identified and recovered.

Table 6: Registry hives extracted with WinHex

Hive	Location
SAM SECURITY SOFTWARE SYSTEM	%SYSTEMROOT%\System32\config
NTUSER.DAT	%SYSTEMROOT%\Users\<username>
UsrClass.dat	%SYSTEMROOT%\Users\<username>\AppData\Microsoft\Windows

The SAM hive provided information about the user accounts, date and time created and last login time. The last login date and time

can be used to corroborate session information recovered from sessionstore:

```

Username      : 0xxx Nxxxxxxx [1000]
Full Name     :
User Comment  :
Account Type  : Default Admin User
Account Created : Sun Sep 25 14:44:15 2016 Z
Name          :
Last Login Date : Sat Dec 1 18:12:26 2018 Z
Pwd Reset Date : Sun Sep 25 14:44:15 2016 Z
Pwd Fail Date  : Never
Login Count    : 8

```

Extracts from the SYSTEM hive indicate the path of the Firefox installer in the net-

work shared folder (see Section 4.1), and an indication that Firefox was executed:

```

1533971040|REG||M... AppCompatCache - Z:\shared_installer_files\Firefox\Setup 61.0.2.exe

1533670994|REG||M... [Program Execution] AppCompatCache - C:\Program Files\Mozilla Firefox\
firefox.exe [Executed]

```

This type of information recovered from the Registry when cross referenced with Event

logs (see Figure 19) verifies user account information and may be used to connect online

identities matching the OS account username e.g. registered social network credentials.

It can also be used to attribute specific browser (social network activity) sessions to a user based on login/logoff times correlated using the “*LogonId*” as seen in Figures 18 and 19.

Other artefacts of interest were recovered

from NTUSER.DAT, indicating when the browser (Firefox) was installed, the system default browser as shown in Table 7, where it was launched from and the number of times the browser was launched as shown in Table 8. For example, starting Firefox from the desktop or taskbar shortcut. This artefact could help an investigator/analyst corroborate session and social network activity.

Table 7: URL indicative of account settings modification

Firefox is set as the default browser	
StartMenuInternet [Sat Aug 18 18:11:33 2018 (UTC)]	
NOTE: default Internet Browser client key (default) -> Firefox-308046B0AF4A39CB	

Table 8: URL indicative of account settings modification

Browser execution and Run count		
Datetime stamp	Path	Run count
Sat Dec 1 2018 18:19:15	9E3995AB-1F9C-4F13-B827-48B24B6C7174\TaskBar\Firefox.lnk	1

Event IDs 4720, 4624 and 4647 mean ‘A user account was created’, ‘An account was successfully logged on’ and ‘User initiated logoff’ respectively (Ultimate IT Security, 2014). There are other description fields in the Win-

dows event logs that can provide additional information and context (e.g. Account Security ID, Domain etc.) but they are out of scope for this paper. Figures 18, 19 and 20 show extracts from the event logs indicating account creation and activity.

The artefacts presented in this section provide additional means of contextualizing the URL artefacts discussed in Section 5.1.1. These artefacts can be used to corroborate the inference that the browser activity of interest as discussed in Section 5.1.2.2, is within the time frame of the user’s last logon and logoff on the system.

5.1.2.5 Prefetch

Prefetch is used by Windows for memory management, speeding up the boot process and application start up process. Prefetch files can be found in %SYSTEMROOT%\Prefetch and have a .pf extension.

Prefetch artefacts can help an investigator/analyst determine when an application was installed, when it was last run and the number of times the application has been run.

In the context of this research, Prefetch provides information related to Firefox thus, corroborating other artefacts recovered previously (see Tables 7 and 8). Figure 21 is a

Prefetch extract showing when Firefox was installed, the last time it was run and the number of times it has been run since installation.

5.1.2.6 Keyword Search

As the last task of the second stage of the model, a keyword search was used to recover artefacts resident in other parts of the disk such as unallocated space, slack space and pagefile. The simultaneous search feature in WinHex was used to search across a variety of character encodings. A keyword search is useful in the identification and recovery of outlier artefacts that may be resident in unstructured part of a disk where they are not readily visible or accessible when viewed in a digital forensics tool.

It is important to use search terms or strings that would reduce the number of false posi-

tives returned. This may involve using some of the features extracted from the URLs or keys from JSON data discussed in Sections 5.1.1 and 5.1.2.2. Examples of the search terms used include but is not limited to:

- “followed_by”:
- “following”:
- “follower_requests”
- “profile_image_url”
- “is_following_advertiser”
- “@twitter_handle”

Figure 22 shows an example of the results returned by the keyword search. It contains the tweet sent by the user.

Keyword searches also highlight artefacts that may be of interest. For instance, a Google Analytics (GA) URL with access to account

settings (“settings/safety”) inferred in its payload data was recovered.

6934584320	68 74 74 70	73 3A 2F 2F 77 77 77 2E	https://www.
6934584336	67 6F 6F 67 6C 65 2D 61	6E 61 6C 79 74 69 63 73	google-analytics
6934584352	2E 63 6F 6D 2F 63 6F 6C	6C 65 63 74 3F 76 3D 31	.com/collect?v=1
6934584368	26 5F 76 3D 6A 37 32 26	61 69 70 3D 31 26 61 3D	&_v=j72&aip=1&a=
6934584384	32 39 34 37 38 31 38 35	33 26 74 3D 70 61 67 65	294781853&t=page
6934584400	76 69 65 77 26 5F 73 3D	32 26 64 6C 3D 68 74 74	view&s=2&dl=htt
6934584416	70 73 25 33 41 25 32 46	25 32 46 74 77 69 74 74	ps%3A%2F%2Ftwitt
6934584432	65 72 2E 63 6F 6D 25 32	46 73 65 74 74 69 6E 67	er.com%2Fsetting
6934584448	73 25 32 46 73 61 66 65	74 79 26 64 72 3D 68 74	s%2Fsafety&dr=ht
6934584464	74 70 73 25 33 41 25 32	46 25 32 46 74 77 69 74	tps%3A%2F%2Ftwit
6934584480	74 65 72 2E 63 6F 6D 25	32 46 73 65 74 74 69 6E	ter.com%2Fsettin
6934584496	67 73 25 32 46 73 61 66	65 74 79 26 64 70 3D 25	gs%2Fsafety&dp=%
6934584512	32 46 75 73 65 72 25 32	46 68 6F 6D 65 25 32 46	2Fuser%2Fhome%2F
6934584528	68 6F 6D 65 26 75 6C 3D	65 6E 2D 67 62 26 64 65	home&ul=en-gb&de
6934584544	3D 55 54 46 2D 38 26 64	74 3D 54 77 69 74 74 65	=UTF-8&dt=Twitte
6934584560	72 26 73 64 3D 32 34 2D	62 69 74 26 73 72 3D 31	r&sd=24-bit&sr=1
6934584576	30 32 34 78 37 36 38 26	76 70 3D 31 30 30 37 78	024x768&vp=1007x
6934584592	36 35 34 26 6A 65 3D 30	26 5F 75 3D 53 41 43 41	654&je=0&_u=SACA
6934584608	41 51 41 42 7E 26 6A 69	64 3D 26 67 6A 69 64 3D	AQAB~&jid=&gjid=
6934584624	26 63 69 64 3D 38 33 37	39 39 38 39 36 35 2E 31	&cid=837998965.1
6934584640	35 34 33 36 38 38 35 31	35 26 74 69 64 3D 55 41	543688515&tid=UA
6934584656	2D 33 30 37 37 35 2D 36	26 5F 67 69 64 3D 31 33	-30775-6&_gid=13
6934584672	37 32 33 37 30 37 31 36	2E 31 35 34 33 36 38 38	72370716.1543688
6934584688	35 31 35 26 7A 3D 34 37	33 37 33 33 30 30 35 00	515&z=473733005

Figure 23 is an example of a GA URL with user login activity in the payload data.

Table 9 breaks down the login URL and describes the parameters. “safety/security” etc. on Twitter. When investigating social network activity, GA URLs if

Other GA URLs results from the keyword search infer user access to “settings/account”, found on disk, may be useful in understanding user activity.

Table 9: Google Analytics URL parameter breakdown (source: Google Developers (2018))

Parameter	Value	Description
t	pageview	This is the ‘Hit’ type. Permitted values for this parameter are ‘pageview’, ‘screenview’, ‘event’, ‘transaction’, ‘item’, ‘social’, ‘exception’, ‘timing’.
_s	1	Hit sequence. The value increments by 1 with each pageview hit.
dl	https://twitter.com/	Document location URL: This parameter sends a resource (or document) location.
dr	https://twitter.com/login	Document referrer: the format for the value for this parameter is a URL (and specifies the referral source of traffic).
dp	/user/home/home	Document path (i.e. resource path) specifies the ‘path’ portion of the URL.
ul	en-gb	User language.
de	UTF-8	This specifies the character set used in encoding the page / resource (Twitter).
dt	Twitter	Document title. In this instance, “Twitter” is the page title.

Figure 24 shows the parameters from the

GA URL that allude to the cookies set during the session.

Recovering and correctly interpreting artefacts such as the ones discussed in this section will enable the investigator/analyst to explain the important aspects of the recovered URL features in the context of a user's social relationships and activity. For example, explaining:

- i a user's connection to a social networking site e.g. account set up and credentials
- ii a user's social relationships e.g. if the user is following or being followed by another user
- iii whether Twitter IDs found are as a result of direct contact, sponsored content or background processes

5.2 Section Summary

Section 5.1 discussed the experimental results, and how the proposed two-stage model can be implemented. This section addresses the relevance of artefacts recovered and the attribution of artefacts to a user.

5.2.1 Relevance of Recovered Artefacts

There are a variety of crimes that can be committed on social media (Osborne, 2010; BBC News, 2012; McGuire, 2019a, 2019b) these include activities such as, but not limited to cyberstalking, cyberbullying, inappropriate contact with another individual (including minors), fraud (digital currency or crypto-scams, identity theft), impersonation, social engineering and, dissemination of malware.

There are also other instances where social media is used to facilitate physical crime (BBC, 2010; Bowcott et al., 2011; Press Association, 2014; McGuire, 2019b). For example,

burglaries, kidnapping, murder, criminal damage, drug, human or firearms trafficking.

The artefacts discussed in this paper can provide an investigator with information and context when making a determination about possible criminal activity. For example, an investigation into crimes committed on social media would seek to verify that the suspect had initiated contact with this victim. Artefacts inferring profile visits as discussed in Section 5.1.1 can add value to an investigation as it can be used to attribute a visit to the victim's profile to the suspect, and corroborating artefacts (Section 5.1.2) can be used to contextualize the suspect's activities.

Different artefacts provide insight into the possible actions of a user that created them. When taken in isolation, they do not provide the full picture of events however, corroborating each artefact in context, will give the investigator an idea of the sequence of events and the ability to exclude artefacts that are not relevant to the case. Social media artefacts as discussed in this paper can thus be used to reconstruct events, show the intent of the user, and determine user associations/relationships during an investigation.

5.2.2 Attributing Artefacts to a User

When considering the attribution of artefacts to a user, it is important to establish that:

- i a link exists between a user (suspect) and a social media account
- ii links exist between the user (suspect) and the activity performed by the social media account.

To determine the connection between a user and a social media account, artefacts from formhistory (Section 5.1.2.1) may be used to

determine if there are login credentials for the social media account. Also, browser history and session information artefacts may be used to establish a pattern of logins and the home page for the account of interest.

Once a determination about account ownership has been made, the activity of interest associated with the social media account can be linked to the user. This can be achieved by viewing the activity in context, using the URL and corroborating artefacts as well as other types of evidence (alibi, statements) and then determining whether the user was at the keyboard at the time of an incident.

Depending on the type of activity being investigated, it is possible to use the data obtained through the two-stage model to attribute possible actions to a suspect. An example would be using images recovered from the browser cache, in conjunction with other supporting artefacts such as session information and browser history to attribute drug trafficking activity to a social media account. Using additional artefacts such as the account login credentials recovered from formhistory, strings of text advertising drugs obtained through keyword search etc., a determination can be made linking the suspect to the account and thus to the activity.

6. CONCLUSION AND FUTURE WORK

This paper has proposed a two-staged model for investigating social network activity. It has shown that a user's social networking activity can be inferred based on a range of artefacts and that it is possible based on these artefacts recovered, to identify and prioritize evidence that is pertinent to a case.

Although the syntax (see Section 2.3.1) is constant, some parts of URLs may be subject to change in structure due to improvements or changes implemented by the service provider. Such changes may include the implementation

of shortened URLs or changing the location of resources on the website e.g. moving user photos from /home to /home/profile.

This model is currently focused on manual analysis with the help of digital forensics tools. Further work is required to improve this model by automating each stage of the evidence recovery process and testing its generalizability and applicability across various browsers and operating systems.

An automated framework stemming from this research is currently under development and will be used as a proof of concept to test the recoverability of artefacts and user relationship attribution.

REFERENCES

- ACPO. (2012). *Good Practice Guide for Digital Evidence*. Retrieved from <http://library.college.police.uk/docs/acpo/digital-evidence-2012.pdf> (Version: 5.0)
- Agency. (2015). *Five internet trolls a day convicted in UK as figures show ten-fold increase*. Retrieved from <https://www.telegraph.co.uk/news/uknews/law-and-order/11627180/Five-internet-trolls-a-day-convicted-in-UK-as-figures-show-ten-fold-increase.html>
- Arshad, H., Jantan, A., & Omolara, E. (2019). Evidence collection and forensics on social networks: Research challenges and directions. *Digital Investigation*, 28, 126–138.
- BBC. (2010). *BBC News - Facebook murderer to serve at least 35 years* (No. 6/27/2010). Retrieved from <http://news.bbc.co.uk/1/hi/england/wear/8555221.stm>
- BBC News. (2012). *Huge rise in social media 'crimes'*. Retrieved from <https://www.bbc.co.uk/news/uk-20851797>

- Bello, M., & DiBlasio, N. (2013). *Twitter: The new face of crime*. USA Today. Retrieved from <http://www.usatoday.com/story/news/nation/2013/09/29/twitter-crime-dark-side/2875745/>
- Berners-Lee, T., Masinter, L., & McCahill, M. (1994). *Uniform Resource Locators (URL) - RFC 1738*. Retrieved from <http://www.ietf.org/rfc/rfc1738.txt>
- Bowcott, O., Carter, H., & Clifton, H. (2011). *Facebook riot calls earn men four-year jail terms amid sentencing outcry*. Retrieved from <https://www.theguardian.com/uk/2011/aug/16/facebook-riot-calls-men-jailed>
- Cambridge University Press. (2019). *Cambridge Dictionary [Online]*. Retrieved from <http://dictionary.cambridge.org>
- Carvey, H. (2018). *RegRipper*. Retrieved from <https://github.com/keydet89/RegRipper2.8>
- Case, A., & Marziale, L. (n.d.). *RegistryDecoder*. Retrieved from <http://www.infosecisland.com/blogview/17867-Open-Source-Registry-Decoder-11-Tool-Released.html>
- Casey, E. (2002). Error, Uncertainty, and Loss in Digital Evidence. *International Journal of Digital Evidence*, 1(2).
- Casey, E. (2005). *Computer Crime and Digital Evidence: Forensic Science, Computers and the Internet*. In *Encyclopedia of Forensic and Legal Medicine*. Oxford: Elsevier. doi: <https://doi.org/10.1016/B0-12-369399-3/00062-8>
- Casey, E. (2011). Digital Evidence and Computer Crime, Forensic Science, Computers and the Internet. In (Third Edition ed., chap. 1: Foundations of Digital Forensics). Elsevier Inc.
- Chisum, W. J., & Turvey, B. E. (2000). Evidence Dynamics: Locard's Exchange Principle & Crime Reconstruction. *Journal of Behavioural Profiling*, 1(1).
- Chisum, W. J., & Turvey, B. E. (2007). *A History of Crime Reconstruction*. In *Crime Reconstruction*. Elsevier.
- Crown Prosecution Service (CPS). (2018). *Guidelines on prosecuting cases involving communications sent via social media*. Retrieved from <https://www.cps.gov.uk/legal-guidance/social-media-guidelines-prosecuting-cases-involving-communications-sent-social-media>
- Cusack, B., & Son, J. (2012). Evidence Examination Tools for Social Networks. In *10th Australian digital forensics conference* (pp. 33–40). SRI Security Research Institute, Edith Cowan University, Perth, Western Australia. doi: <https://doi.org/10.4225/75/57b3afc1fb861>
- DB4S Project. (n.d.). *DB Browser for SQLite*. Retrieved 2019-02-24, from <http://sqlitebrowser.org>
- Garfinkel, S. L. (2006). Forensic Feature Extraction and Cross-Drive Analysis. *Digital Investigation*, 3S, 71–81. Retrieved from <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.581.9553&rep=rep1&type=pdf>
- Garfinkel, S. L. (2013). Digital media triage with bulk data analysis and bulk_extractor. *Computers & Security*, 32, 56–72. Retrieved from <https://www.sciencedirect.com/science/article/pii/S0167404812001472> doi: <https://doi.org/10.1016/J.COSE.2012.09.011>
- Google Developers. (2018). *Measurement*

- Protocol Parameter Reference — Analytics Measurement Protocol — Google Developers*. Retrieved 2019-03-04, from <https://developers.google.com/analytics/devguides/collection/protocol/v1/parameters>
- Haroon, S., & Carter, H. (2010). *Facebook security measures criticised after Ashleigh Hall murder*. The Guardian. Retrieved from <http://www.theguardian.com/uk/2010/mar/09/ukcrime-facebook>
- Huber, M., Mulazzani, M., Leithner, M., Schrittwieser, S., Wondracek, G., & Weippl, E. (2011). Social Snapshots: Digital Forensics for Online Social Networks. In *Proceedings of the 27th annual computer security applications conference* (pp. 113–122). New York, NY, USA: ACM. doi: <https://doi.org/10.1145/2076732.2076748>
- Jang, Y. J., & Kwak, J. (2015). Digital forensics investigation methodology applicable for social network services. *Springer Series in Multimedia Tools Appl*, 74, 5029–5040. Retrieved from <https://link.springer.com/content/pdf/10.1007%2Fs11042-014-2061-8.pdf> doi: <https://doi.org/10.1007/s11042-014-2061-8>
- Jonsson, P. (2011). 'Flash robs': *How Twitter is being twisted for criminal gain [VIDEO]*. The Christian Science Monitor. Retrieved from <http://www.csmonitor.com/USA/2011/0803/Flash-robs-How-Twitter-is-being-twisted-for-criminal-gain-VIDEO>
- Keyvanpour, M., Moradi, M., & Hasanzadeh, F. (2014, 01). Digital forensics 2.0: A review on social networks forensics. *Studies in Computational Intelligence*, 555, 17–46. doi: <https://doi.org/10.1007/978-3-319-05885-6-2>
- Mabuto, E. K., & Venter, H. S. (2012). User-generated digital forensic evidence in graphic design applications. In *Proceedings title: 2012 international conference on cyber security, cyber warfare and digital forensic (cybersec)* (pp. 195–200). IEEE. Retrieved from <http://ieeexplore.ieee.org/document/6246107/> doi: <https://doi.org/10.1109/CyberSec.2012.6246107>
- McGuire, M. (2019a). *Into The Web of Profit: Social Media Platforms and the Cybercrime Economy*. Bromium. Retrieved from <https://www.bromium.com/wp-content/uploads/2019/02/Bromium-Web-of-Profit-Social-Platforms-Infographic.pdf>
- McGuire, M. (2019b). *Social Media Platforms and The Cybercrime Economy: The next chapter of Into The Web of Profit*. Bromium.
- McKemmish, R. (2008). When is Digital Evidence Forensically Sound? *Advances in Digital Forensics, IV*, 3–15.
- Moore, K. (2014). *Social media 'at least half' of calls passed to front-line police*. BBC News. Retrieved from <https://www.bbc.co.uk/news/uk-27949674>
- Murr, M. (2007). *The admissibility vs. weight of digital evidence — Forensic Computing*. Retrieved 2019-04-14, from <https://forensicblog.org/the-admissibility-vs-weight-of-digital-evidence/>
- NirSoft. (2018a). *FullEventLogView*. Retrieved from https://www.nirsoft.net/utils/full_event_log_view.html
- NirSoft. (2018b). *MZCacheView*. Retrieved from https://www.nirsoft.net/utils/mozilla_cache_viewer.html
- Oh, J., Lee, S., & Lee, S. (2011). Advanced

- evidence collection and analysis of web browser activity. *Digital Investigation*, 8, Supplem(0), S62 – S70. Retrieved from <http://www.sciencedirect.com/science/article/pii/S1742287611000326>
doi: <https://doi.org/10.1016/j.diin.2011.05.008>
- Osborne, B. (2010). *Twitter sees more active users, but also attracts more criminal activity*. Geek Website. Retrieved from <http://www.geek.com/news/twitter-sees-more-active-users-but-also-attracts-more-criminal-activity-1130461/>
- Powell, A., & Haynes, C. (2019). Social Media Data in Digital Forensics Investigations. *Digital Forensic Education*, 281–303.
- Press Association. (2014). *Peter Nunn jailed for abusive tweets to MP Stella Creasy*. Retrieved from <https://www.theguardian.com/uk-news/2014/sep/29/peter-nunn-jailed-abusive-tweets-mp-stella-creasy>
- Rankin, B. (2010). *Send in the 'Twitter squad': Police forces may need dedicated to cope with rising social media crime*. Mirror News. Retrieved from <http://www.mirror.co.uk/news/technology-science/technology/rocketing-crime-complaints-involving-social-1507527>
- Richards, J. (2007). *Sex offenders can use social sites, say police - Times Online* (Vol. 2010).
- Select Committee on Communications. (2014). *CHAPTER 2: SOCIAL MEDIA AND THE LAW*. Retrieved from <https://publications.parliament.uk/pa/ld201415/ldselect/ldcomuni/37/3702.htm>
- Shaw, U., Das, D., & Mehdi, S. P. (2016). Social Network Forensics: Survey and Challenges. *International Journal of Computer Science and Information Security (IJCSIS)*, 14(11), 310–316.
- Sommer, P. (1999). Intrusion Detection Systems as Evidence. *Computer Networks*, 31(23–24), 2477–2487.
- Taylor, M., Haggerty, J., Gresty, D., Almond, P., & Berry, T. (2014). Forensic investigation of social networking applications. *Digital Investigation*, 11, 9–16.
- Telerik. (2018). *Fiddler - Free Web Debugging Proxy - Telerik*. Retrieved from <https://www.telerik.com/fiddler>
- Ultimate IT Security. (2014). *Windows Security Log Encyclopedia*. Retrieved 2019-01-18, from <https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/default.aspx>
- Woan, M. (2013). *PrefetchForensics*. GitHub. Retrieved from <https://github.com/woanware/woanware.github.io/blob/master/downloads/PrefetchForensics.v.1.0.4.zip>
- Wood, C. (2018). *WhatsApp photo drug dealer caught by 'groundbreaking' work*. BBC News. Retrieved from <https://www.bbc.co.uk/news/uk-wales-43711477>
- X-Ways Software Technology, AG. (2018). *WinHex*. WinHex: Computer Forensics & Data Recovery Software, Hex Editor & Disk Editor. Retrieved from <https://www.x-ways.net/winhex/>
- Zainudin, N. M., Merabti, M., & Llewellyn-Jones, D. (2011). Online social networks as supporting evidence: A digital forensic investigation model and its application design. In *2011 international conference on research and innovation in information systems* (pp. 1–6). doi: <https://doi.org/>

10.1109/ICRIIS.2011.6125728

Key Node	Value
1	
cacheKey	0
children	
docIdentifier	17
docshellUUID	{c0952ff0-1dbb-459f-af6e-fc8a9649bef9}
ID	34
persist	true
presState	
0	
principalToInherit_base64	vQZuXxRvRHkDMxv9BbHkAAAAAAAAAAAAAAAAAAAAAAEYAAAA4bW96LW5...
resultPrincipalURI	null
structuredCloneState	AgAAAAA8f8AAAAACAD//wUAAIAEAP//dGI0bGUAAAAAACAABAD//1R3aXR...
structuredCloneVersion	8
title	Twitter / Settings
triggeringPrincipal_base64	ZT4OTT7kRfqycpCC8AeuAAAAAAAAAAAAAAAAAAAAAAEYB3pRy0IA0EdOTm...
url	https://twitter.com/settings/account

Figure 11: JSON (sessionstore) artefacts showing session where user account settings were accessed

Key Node	Value
7	
cacheKey	0
children	
docIdentifier	72
docshellUUID	{c0952ff0-1dbb-459f-af6e-fc8a9649bef9}
ID	72
originalURI	https://twitter.com/N[redacted]/followers
persist	true
principalToInherit_base64	ZT4OTT7kRfqycpCC8AeuAAAAAAAAAAAAAAAAAAAAAAEYB3pRy0IA0EdOTm...
referrer	https://twitter.com/
referrerPolicy	1
resultPrincipalURI	null
structuredCloneState	AgAAAAA8f8AAAAACAD//wUAAIAEAP//dGI0bGUAAAAA4AACABAD//1Bb3Bs...
structuredCloneVersion	8
title	People following O[redacted]s (@N[redacted]) Twitter
triggeringPrincipal_base64	ZT4OTT7kRfqycpCC8AeuAAAAAAAAAAAAAAAAAAAAAAEYB3pRy0IA0EdOTm...
url	https://twitter.com/N[redacted]/followers

Figure 12: JSON (sessionstore) artefacts showing referrer URL through which the user reached “followers”

Key Node	Value
attributes	
entries	
formdata	
id	
device_country_code	
url	https://twitter.com/S[redacted]3
xpath	
//xhtml:div[@id='global-tweet-dialog-dialog']/xhtml:div[2]/xhtml:div...	
//xhtml:div[@id='global-tweet-dialog-dialog']/xhtml:div[2]/xhtml:div...	
//xhtml:div[@id='global-tweet-dialog-dialog']/xhtml:div[2]/xhtml:div...	
//xhtml:div[@id='global-tweet-dialog-dialog']/xhtml:div[2]/xhtml:div...	@S[redacted]3 Hey, gimme a call tomorrow
//xhtml:div[@id='global-tweet-dialog-dialog']/xhtml:div[2]/xhtml:div...	@S[redacted]3 Hey, gimme a call tomorrow

Figure 13: JSON (sessionstore) artefacts indicate the user visited another user’s profile and sent a tweet

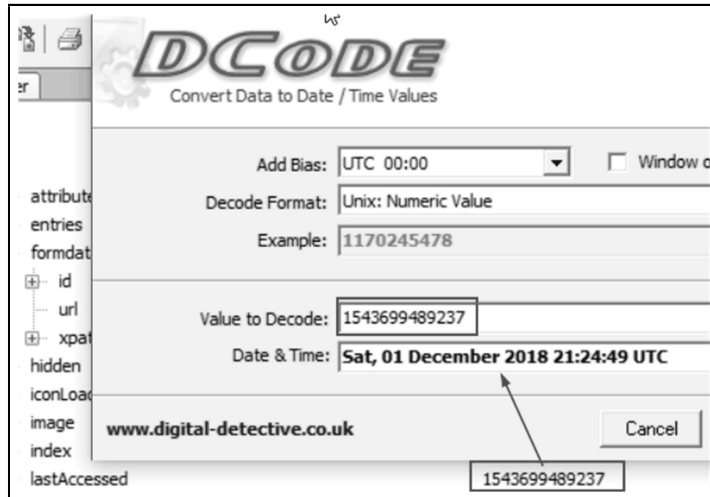


Figure 14: JSON (sessionstore) artefacts showing the last accessed date and time for a session. Timestamp decoded by DCode

```
"name": "Oxxx Nxxxxxx",
"screen_name": "NxxxxxxOxxx",
"profile_image_url": "http://pbs.twimg.com/profile_images/103xxxxxxxxxxxxx616/9nkteNKb_normal.jpg",
"profile_image_url_https": "https://pbs.twimg.com/profile_images/103xxxxxxxxxxxxx616/9nkteNKb_normal.jpg",
```

Figure 15: Extract of user profile data from webappsstore.sqlite

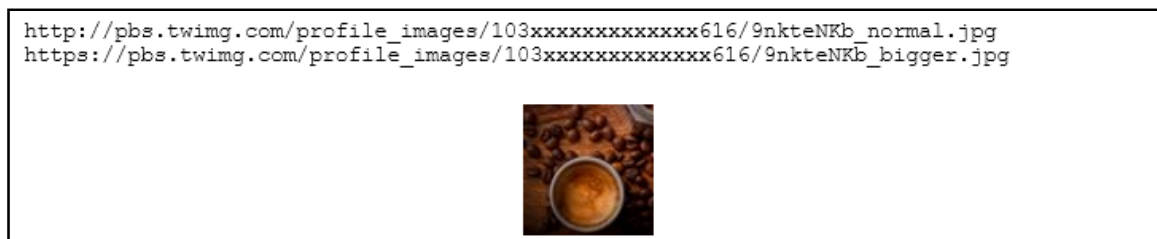


Figure 16: Extract of user profile image from the cache

Name	Value
advertiser_name	Nespresso UK&Ireland
cardname	unified_card
is_following_advertiser	false
forward	true
impression_id	358a4182a9a96b66
edge	true
lang	en
card_height	271
scribe_context	{"client":"web","page":"home","section":"home","component":"tweet"}
bearer_token	AAAAAAAAAAAAAAAAAAAAAPYXBAAAAAAACLXUNDeKmxqa8h%2F40K4moUkGsoc%3DTYfbDKbT3jJPCEVnMYqilB2sNHfOPqkca3qaAxGfsyKCs0wRbw

Transformer	Headers	TextView	SyntaxView	ImageView	HexView	WebView	Auth	Caching	Cookies	Raw	JSON	XML
-------------	---------	----------	------------	-----------	---------	---------	------	---------	---------	-----	------	-----

JSON

```

card
  bearer_token=AAAAAAAAAAAAAAAAAAAAAPYXBAAAAAAACLXUNDeKmxqa8h%2F40K4moUkGsoc%3DTYfbDKbT3jJPCEVnMYqilB2sNHfOPqkca3qaAxGfsyKCs0wRbw
  card_name=unified_card
  card_uri=card://1036931798771331532
  card_url=https://twitter.com
  forward=true
  is_author=False
  is_caps_enabled=True
  is_emojiify_enabled=True
  is_logged_in=True
  is_static_view=False
  language=en
  scribe_context=%7B%22client%22%3A%22web%22%2C%22page%22%3A%22home%22%2C%22section%22%3A%22home%22%2C%22component%22%3A%22tweet%22%7D
  token=0649d3b91dac6f31e830949e7b03e4f67bbd63c
  tweet_id=103693203549363113
  viewer_id=1031133833213034496
  scribeData
    card_name=unified_card
    card_url=https://twitter.com
  strings
    twitter_cldr=False

```

Figure 17: Extracts from Fiddler showing the HTTP Request and Response for the cached sponsored content

```

<EventID>4720</EventID>-
<TimeCreated SystemTime="2016-09-25T14:44:15.715800100Z"/>-
<Channel>Security</Channel>-
<EventData>-
  <Data Name="TargetUserName">Oxxx Nxxxxxxx</Data>-
  <Data Name="SamAccountName">Oxxx Nxxxxxxx</Data>-
  <Data Name="LogonHours">%1797</Data>-
</EventData>

```

Figure 18: Account creation dates, time, username (extract from Security Event log)

```

<EventID>4624</EventID>-
<TimeCreated SystemTime="2018-12-01T18:12:26.149464600Z"/>-
<Channel>Security</Channel>-
<EventData>-
  <Data Name="TargetUserName">Oxxx Nxxxxxxx</Data>-
  <Data Name="TargetLogonId">0x0000000000015a3c</Data>-
  <Data Name="LogonType">2</Data>-
  <Data Name="ProcessName">C:\Windows\System32\winlogon.exe</Data>-
  <Data Name="IpAddress">127.0.0.1</Data>-
  <Data Name="IpPort">0</Data>-
</EventData>-</Event>-

```

Figure 19: Login date, time, type and username (extract from Security Event log)

```

<EventID>4647</EventID>-
<TimeCreated SystemTime="2018-12-01T21:31:39.057611700Z"/>-
<Channel>Security</Channel>-
<EventData>-
  <Data Name="TargetUserName">Oxxx Nxxxxxxx</Data>-
  <Data Name="TargetLogonId">0x0000000000015a3c</Data>-
</EventData>-</Event>-

```

Figure 20: Logoff date, time, and username (extract from Security Event log)

File Name	Created Date/Time	Modified Date/Time	Date Last Run	Num Times Run	Path Hash	Calc Hash	Physical Path
FIREFOX SETUP 61.0.2.EXE-EF039B4B.pf	18 August 2018 (Sat) 18:11:12	18 August 2018 (Sat) 18:11:12	18 August 2018 (Sat) 18:11:02	1	EF039B4B		
FIREFOX.EXE-E60C0AA7.pf	01 December 2018 (Sat) 18:18:08	01 December 2018 (Sat) 18:21:04	01 December 2018 (Sat) 18:21:04	10	E60C0AA7	E60C0AA7	\DEVICE\HARDISKVOLUME1\PROGRAM FILES\MOZILLA\FIREFOX\FIREFOX.EXE

Figure 21: Firefox last run date and run count

073E18C00	5D 22 3A 22 40 53						33 20	]": "@S= 3
073E18C10	40 65 79 2C 20	67 69 6D	6D 65 20 61 20 63 61 6C					Hey, Gimme a cal
073E18C20	6C 20 74 6F 6D 6F 72 72	6F 77 22 2C AA 00 54 54						l tomorrow", a TT

Figure 22: Tweet fragment recovered from unallocated space

```
https://www.google-analytics.com/collect?v=1&_v=j72&aip=1&a=1803425654
&t=pageview
&s=1
&dl=https%3A%2F%2Ftwitter.com%2F
&dr=https%3A%2F%2Ftwitter.com%2Flogin
&dp=%2Fuser%2Fhome%2Fhome
&ul=en-gb
&de=UTF-8
&dt=Twitter
&sd=24-bit
&sr=1024x768
&vp=1007x617
&je=0
&_u=QACAAQAB~
&jid=
&gjid=
&cid=837998965.1543688515
&tid=UA-30775-1ca8eb7406
&_gid=1372370716.1543688515
&z=888718953
```

Figure 23: Twitter login activity captured in a Google Analytics URL

baseDomain	name	value	host	path
twitter.com	dnt	1	.twitter.com	/settings/safety/
twitter.com	fm	0	.twitter.com	/
twitter.com	_gat	1	.twitter.com	/
twitter.com	_ga	GA1.2.837998965.1543688515	.twitter.com	/
twitter.com	_gid	GA1.2.1372370716.1543688515	.twitter.com	/
twitter.com	personalization_id	"v1_hu[REDACTED]MyA=="	.twitter.com	/
twitter.com	guest_id	v1%3A154368[REDACTED]B9493	.twitter.com	/
twitter.com	ct0	61ddd5fe191ebcee41d753de50f1499d	.twitter.com	/
twitter.com	eu_cn	1	.twitter.com	/
twitter.com	ads_prefs	"HBISAAA="	.twitter.com	/
twitter.com	kdt	MvecUT[REDACTED]500q1mQ372	.twitter.com	/
twitter.com	remember_checked_on	1	.twitter.com	/
twitter.com	twid	"u=1031[REDACTED]496"	.twitter.com	/
twitter.com	auth_token	5a[REDACTED]	.twitter.com	/
twitter.com	csrf_same_site_set	1	.twitter.com	/
twitter.com	csrf_same_site	1	.twitter.com	/
twitter.com	dnt	1	.twitter.com	/

Figure 24: Cookies.sqlite shows cookie info found in a Google Analytics URL

D.3

David, A. and Hargreaves, C. (2013). Detecting Tor Vidalia Bundle installation.
DOI: 10.13140/2.1.4362.0642.

Detecting Tor Vidalia Bundle installation

Anne David¹, Christopher Hargreaves²

Centre for Forensic Computing and Security, Cranfield University, Shrivenham
{a.david1,c.j.hargreaves2}@cranfield.ac.uk

Abstract. Tor is an open network run by volunteers worldwide. It ensures a user's anonymity by relaying communication around a distributed network of nodes thus protecting users from the network adversary (traffic analysis) and the local adversary (digital forensic investigators). Tor Vidalia Bundle is a graphical user interface (GUI) that helps a user control access to the Tor network. The way Tor interacts with Firefox via the Torbutton means that web browser artefacts are prevented from being written to disk thus posing a challenge to digital investigations. This paper investigates the possibility of detecting Tor Vidalia Bundle installation on a user's computer. Determining whether a user may be communicating through Tor can help an investigator focus on other areas if web browser artefacts are not available on disk. It can also help create a timeline of when Tor was installed and when it was last used so user behaviour may be inferred.

Keywords. Tor, anti-forensics, digital forensics.

1 Introduction

There is growing concern about user privacy and security while communicating online [1], [2] hence the use of technologies that address issues like location disclosure, and identity disclosure [3]. An example is Tor (The Onion Routing) Network which allows users to communicate anonymously on the Internet. This is achieved through transmitting user communication through other nodes on the network. Tor can be considered a challenge to digital investigations based on its design requirements and the information on the project website [4], [5]. Vidalia is one of Tor's services and is combined with other services to enhance user safety and security on the Internet.

This paper investigates the artefacts that are present as a result of Tor Vidalia Bundle installation. describes the Tor Vidalia Bundle in relation to digital investigations; it presents methods of identifying and analysing artefacts of evidential value that may allow the presence of Tor Vidalia Bundle to be inferred. This paper makes the following contributions: identification and documentation of some of the artefacts that exist as a result of the installation of Tor Vidalia Bundle; provides a discussion of the evidential value of these artefacts and also shows the possibility of making inferences about user actions and establishing a timeframe for the observed actions.

This paper is made up of six sections: an introduction; a brief background of Tor and reasons for its use; an experimental methodology section; results section; evaluation and future work; and the conclusion.

2 Background and related work

Tor Vidalia Bundle is a service provided by Tor and consists of a graphical controller for Tor and it has a range of options that enables a user to configure Tor. Tor Vidalia Bundle is made up of four components – Vidalia, Polipo, Tor and the Torbutton. The Torbutton is a browser add-on for Firefox and was designed based on adversary goals and capabilities [4]. It is expected to meet security and privacy requirements such as disk avoidance and unlinkability. Disk avoidance means that nothing from a Tor initiated browser activity is written to disk while unlinkability ensures that traffic analysis is hindered. In addition to these requirements, the Torbutton meets undiscoverability (the presence of Tor cannot be detected on the network) and location neutrality. From an investigative perspective, these can be described as anti-forensic measures and may make the recovery and analysis of artefacts from web based communication on the hard disk level highly unlikely.

Existing research on Tor, mainly from a network perspective [6] [7], is focused on identifying vulnerabilities and improving the network. There is also limited research on browser extensions such as the Torbutton. Anti-forensics research has produced techniques and tools for detecting encryption [8], malware [9], etc. but there is a gap in knowledge in relation to Tor research as there are no tools for the detection of Tor Vidalia Bundle. In the digital forensics domain, there are gaps in research on the identification of artefacts from browser extensions. From a disk analysis perspective, in the absence of web browser artefacts on a user's hard disk, establishing the presence of Tor would be the first step in narrowing down the likely locations of residual artefacts e.g. unallocated space.

3 Methodology

3.1 Aim

The aim of this research is to identify the artefacts that indicate that Tor may be in use on a system, specifically that the Tor Vidalia Bundle is installed.

3.2 Data generation and analysis

The experimental method was used in this paper. A virtual machine was created and Windows 7 was installed. Procmon was also installed to monitor associated processes running in the background. Tor Vidalia Bundle (Vidalia 0.2.15, Tor 0.2.2.35, Polipo 1.0.4.1, and Torbutton 1.4.5) was downloaded and installed as client only. This prompted the installation of Firefox 9 in order to use Torbutton browser plugin. After installing Tor Vidalia Bundle (TVB), the Vidalia Control Panel auto launched and a

connection to the Tor network was established. Firefox was also auto launched; the browser default settings and Torbutton preferences were explored, the connection to Tor was subsequently terminated then the browser and Vidalia Control Panel were shut down.

The data analysis involved making a copy of the virtual machine (.vmdk file). Using a multi-tool technique, a keyword search for Vidalia, Polipo, Tor, and Torbutton on the disk (.vmdk) was conducted using X-Ways Forensics, an analysis of the Procmon logs was conducted, user directories and files were analysed using X-Ways Forensics, FTK imager, and SQLite3Explorer. Associated prefetch files were analysed using PrefetchForensics from woanware, while the Windows 7 Registry was analysed using AccessData Registry Viewer.

4 Results

This section discusses the results from the data analysis conducted after the installation of Tor Vidalia Bundle. This section is broken down into location specific results including Prefetch, registry, and other files and locations.

4.1 Prefetch

Prefetching is a system used by Microsoft Windows to speed up application start up process. Information on applications accessed by a user can be saved in a number of files in the prefetch folder [10] which can be found by default in %System-Root%\Prefetch e.g. C:\Windows\Prefetch.

A Windows 7 prefetch file consists of a header field (size 0xF0) which contains information such as: file signature “SCCA”, the prefetch file name and size, application run count, last accessed date and time; a list of files and directories accessed by the application. For more on Windows prefetch file structure, see [11], [12].

As seen in Table 1, the prefetch files of interest were those created for Firefox setup.exe, Firefox.exe, Polipo.exe, Tor.exe, Vidalia-Bundle and Vidalia.exe. The dates, times and run count of these files correspond with recorded installation time.

Table 1. Prefetch files recovered from C:\Windows\Prefetch and analysed using PrefetchForensics

Filename	Created Time	Modified Time	Size	Process Path	Run Count	Last Run Time
FIREFOX%20SETUP%2020.0[1].EXE-1FC07D63.pf	04/04/2013 12:58	04/04/2013 12:58	108,830	\DEVICE\HARDDISKVOL_UME1\USERS\USER1\APPDATA\LOCAL\MICROSOFT\WINDOWS\TEMPORARY INTERNET FILES\CONTENT.IE5\74Z ZKR86\FIREFOX%20SETUP%2020.0[1].EXE	1	04/04/2013 12:58
POLIPO.EXE-F3F9017C.pf	04/04/2013 13:09	04/04/2013 13:09	18,954	\DEVICE\HARDDISKVOL_UME1\VIDALIA BUNDLE\POLIPO\POLIPO.EXE	1	04/04/2013 13:09

TOR.EXE-CBF6CD08.pf	04/04/2013 13:09	04/04/2013 13:11	43,148	\\DEVICE\\HARDDISKVOL UME1\\VIDALIA BUNDLE\\TOR\\TOR.EXE	2	04/04/2013 13:11
VIDALIA-BUNDLE-0.2.2.35-0.2.1-AA17B669.pf	04/04/2013 12:55	04/04/2013 13:00	91,760	\\DEVICE\\HARDDISKVOL UME1\\USERS\\USER1\\DO CUMENTS\\VIDALIA- BUNDLE-0.2.2.35-0.2.15- 1.EXE	2	04/04/2013 13:00
VIDALIA.EXE-5791F07A.pf	04/04/2013 13:09	04/04/2013 13:09	61,610	\\DEVICE\\HARDDISKVOL UME1\\VIDALIA BUNDLE\\VIDALIA\\VIDA LIA.EXE	1	04/04/2013 13:09

4.2 Other related files and their locations

As mentioned in Section 3.2, during installation, Vidalia Bundle prompts a user to install Firefox. This required aborting the Vidalia installation process in order to install Firefox.

The user has the choice of installing full, base or custom components for Tor Vidalia Bundle. These components are installed in the default destination folder however a user can specify the destination. The installation folder contains subfolders and files required by Vidalia to function properly. The components of a full install are described briefly in Table 2.

Table 2. Vidalia Bundle components and default locations

File	Description/Additional information	Default Location
Vidalia Bundle	Contains \\Polipo, \\Vidalia, \\Tor, \\Torbutton, uninstall.exe	\\Program Files\\Vidalia Bundle
Polipo	A small web caching proxy that increases performance of web browsing through Tor. Subfolder also contains Change log files, config files, Polipo website shortcut, uninstall.exe	\\Program Files\\Vidalia Bundle\\Polipo
Tor.exe	This executable is a system for using the Internet anonymously, helping a user anonymize web browsing and publishing, instant messaging, IRC and more. Subfolder also contains tor-resolve, Tor website shortcut and a documents folder with Tor specifications	\\Program Files\\Vidalia Bundle\\Tor
Vidalia.exe	This is a GUI that helps a user control, monitor and configure Tor. Subfolder contains DLL files, license files, changelog files, and Vidalia website shortcut	\\Program Files\\Vidalia Bundle\\Vidalia
Torbutton	This is a Firefox extension that allows a user to quickly enable or disable anonymous web browsing. In order to use the Torbutton, Firefox must be pre-installed on a user's PC. Subfolder contains a license file and torbutton-1.4.5.xpi	\\Program Files\\Vidalia Bundle\\Vidalia\\Torbutton

In addition to these files and directories, a text fragment of the Firefox prompt was recovered from unallocated space that relates to the installation of Tor Vidalia Bun-

dle. This artefact may not be recoverable in all cases as it was in unallocated space and prone to be overwritten. However, a keyword search for Vidalia Bundle Setup, Tor, Torbutton or Polipo should recover this artefact if it is present.

There are also potentially useful files which can be found in the Vidalia, Tor and Mozilla subfolders in \Users\username\AppData\Roaming and are discussed in below:

\Vidalia

This subfolder contains files that can be viewed and edited in a text editor: geoip-cache (contains a list of IP addresses for Tor nodes); torrc (contains Tor configuration instructions and can be modified by a user to specify preferred exit nodes) and vidalia.conf (Vidalia configuration file).

\Tor

This subfolder contains files that can be viewed in a text editor: cached-certs (contains directory keys and certificates); cached-consensus (contains network status, client and server versions); cached-descriptors, cache-descriptors.new (Tor download info – date, time, source, default router, platform, and public keys); geoip (list of IP addresses with geographical location); state (editing this file is not recommended. It contains a list of entry nodes, circuit build times and dates).

\Mozilla\Firefox\Profile\~.default

This subfolder contains files used by Firefox. However the files related to the installation of Vidalia Bundle are extensions.sqlite, permissions.sqlite and prefs.js:

Extensions.sqlite: this file is located by default in the user's Firefox profile folder \Users\username\AppData\Roaming\Mozilla\Firefox\Profile\~.default\extensions.sqlite. When analysed with SQLite3Explorer.exe, it listed the Torbutton as a browser extension with the GUID - {e0204bd5-9d31-402b-a99d-a6aa8ffebdca}. It also included the source of the extension and an .xpi file (located in the extensions subfolder in the \extensions\{e0204bd5-9d31-402b-a99d-a6aa8ffebdca}.xpi).

Permissions.sqlite: this is a permission database for add-ons installation, cookies, and image loading in Firefox [13]. It makes use of values 0-2 in Firefox. 0 = default permission granted, 1 = permission granted, 2 = permission denied. When analysed with SQLite3Explorer.exe, permissions.sqlite showed a value of 1 for "torproject.org" granting permission for the use of strict transport security (enforcing https only connections) [13].

Prefs.js: this is a plain text file used by Firefox to store user browser preferences and is located by default in a user's Firefox profile folder: \Users\username\AppData\Roaming\Mozilla\Firefox\Profile\~.default\prefs.js. According to [4], in order to meet disk avoidance requirements, the Torbutton modifies normal browser functionality by resetting preference values which are then written to

the prefs.js file. Table 3 shows the following preference values have been modified (see [14] for their default values):

Table 3. Modified Firefox preferences as seen in prefs.js after installing Tor Vidalia Bundle

Preference	Value	Description
<i>dom.storage.enabled</i>	false	prevents the storage of persistent data on disk
<i>browser.cache.memory.enable</i>	false	prevents the caching of decoded images and secure pages in memory
<i>browser.cache.disk.enable</i>	false	prevents the disk from caching content
<i>browser.cache.offline.enable</i>	false	prevents the browser from downloading resources for the offline cache [15]
<i>network.http.use-cache</i>	false	Prevents the caching of http or https files
<i>general.open_location.last_url</i>	false	prevents the retrieval of the most recently opened URL [15]
<i>places.history.enabled</i>	false	prevents the browser from retaining history [15]
<i>browser.formfill.enable</i>	false	prevents the browser from storing or retrieving form field entries
<i>signon.rememberSignons</i>	false	disables the password manager
<i>browser.download.manager.retention</i>	0	indicates how long downloads are retained after completion 0 indicates downloads should be removed on completion [15]
<i>network.cookie.lifetimePolicy</i>	2	causes all cookies to expire at the end of the session [15]

4.3 Registry

The Windows registry is a database that contains important information about system hardware, installed programs, settings and user profiles on a Windows PC [10]. The registry is referred to and is updated when changes such as the installation of a new program or addition of a user account are made.

Several registry keys related to the install were identified and the Local Machine Software hive and NTUSER.DAT were extracted from \Windows\System32\config and \Users\username respectively. These artefacts are discussed in order of installation, local user settings, and program launch activity. These artefacts include paths to where Vidalia and its components were installed, modification and associated uninstall details as shown in Table 4.

Table 4. Registry entries relating to the installation of Tor Vidalia Bundle.

Hive	Key
HKCU (NTUSER.DAT)	HKCU\Software\Polipo
Description: The key contains a string value Install_Dir and the path to where Polipo is installed: "C:\Program Files\Vidalia Bundle"	
HKLM\Software	Key: HKLM\Software\Microsoft\Windows\CurrentVersion\Uninstall\ Subkeys: Polipo, Tor, Vidalia

Description: This key contained two string values; DisplayName, and UninstallString, and two REG_DWORD entries NoModify, and NoRepair. The data for these entries are program name and version, the path to the uninstall executable - "C:\Program Files\Vidalia Bundle\Uninstall.exe" and the value of 1 (disabled) for NoModify and NoRepair.	
HKCU(NTUSER.DAT)	Key: HKCU\Software\Microsoft\Windows\CurrentVersion\Run\Vidalia
Description: This key contains a string value for Vidalia and the program launch path "C:\Program Files\Vidalia Bundle\Vidalia\vidalia.exe".	
HKCU(NTUSER.DAT)	Key: HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist\ {CEBFF5CD-ACE2-4F4F-9178-9926F41749EA}\Count:
Description: The UserAssist key also contains path to the accessed program. The entries for Vidalia-bundle had a launch count of 2 which is consistent with aborting and restarting the program while Tor.exe had a launch count of 1, both with consistent last accessed dates and time	

5 Evaluation and further work

This paper has shown that the presence of Tor can be detected using prefetch, registry and a selection of other files on disk. However, there are limitations to this research. First experiments were conducted with only one version of Tor Vidalia Bundle, and this is only one way of accessing Tor. The paper has shown some of the files that are installed however a hash set of these files has not been presented in this paper. Also the paper has considered only the detection of the installation. Further work will involve attempting to recover evidence from browsing whilst using Tor.

6 Conclusion

User privacy and security on the Internet has been described as a cause for concern causing users to employ privacy protection measures such as anonymizing technologies like Tor. In order to connect to Tor, the user is required to install software such as the Vidalia Bundle which has added properties that modify browser behavior and in some cases, prevent disk writes.

This paper has shown that it is possible to determine that Tor Vidalia Bundle is on a machine. This paper has suggested locations that are worth investigating and the type of artefacts that can be recovered. It can be seen that it is possible to recover text strings related to the set up GUI and also decode artefacts from Windows registry hives and Prefetch.

References

1. M. A. Caloyannides, *Privacy Protection and Computer Forensics*. Norwood, MA: Artech House, 2004.
2. J. Claessens, B. Preneel, and J. Vandewalle, *Solutions for Anonymous Communication on the Internet*, 1999.
3. A. Pfitzmann and M. Hansen, *Anonymity, Unlinkability, Undetectability*,

Unobservability, Pseudonymity, and Identity Management - A Consolidated Proposal for Terminology, 2008.

4. M. Perry. (2011, Apr.) Tor Project. [Online]. <https://www.torproject.org/torbutton/en/design/#id2702702>
5. Tor Project. What is Tor? [Online]. <https://www.torproject.org/>
6. A. Chaabane, P. Manils, and M.A. Kaafar, "Digging into Anonymous Traffic: A Deep Analysis of the Tor Anonymizing Network," in *Network and System Security (NSS), 2010 4th International Conference on*, September 2010, pp. 167-174.
7. Roger Dingledine, Nick Mathewson, and Paul Syverson, "Tor: The Second-Generation Onion Router," in *Proceedings of the 13th USENIX Security Symposium*, August 2004. [Online]. <https://svn.torproject.org/svn/projects/design-paper/tor-design.html>
8. Eoghan Casey, Geoff Fellows, Matthew Geiger, and Gerasimos Stellatos, "The growing impact of full disk encryption on digital forensics," *Digital Investigation*, vol. 8, no. 2, pp. 129-134, 2011.
9. Dan Haagman and Byrne Ghavalas, "Trojan defence: A forensic view".
10. Microsoft Windows. (2013) [Online]. <http://windows.microsoft.com/en-gb/windows7/>
11. M. A. Wade. (2010, Dec.) Digital Forensic Investigator (DFI) News. [Online]. Decoding Prefetch Files for Forensic Purposes: Part 1
12. Mark Woan. woanware. [Online]. <http://www.woanware.co.uk/forensics/prefetchforensics.html>
13. mozillaZine. (2011, Feb.) [Online]. <http://kb.mozillazine.org/>
14. Mozilla Developer Network. (2011, Nov.) Mozilla networking preferences. [Online]. https://developer.mozilla.org/en-US/docs/Mozilla/Preferences/Mozilla_networking_preferences
15. Mozilla Developer Network. (2011) [Online]. <https://developer.mozilla.org/>

D.4.

D.4

David, A. and Hargreaves, C. (2012). An Analysis of Hotmail Artefacts in Firefox 9.
DOI:10.13140/RG.2.1.4750.0326.

An Analysis of Hotmail Artefacts in Firefox 9

Anne David¹, Christopher Hargreaves²

Centre for Forensic Computing
Department of Engineering and Applied Science
Cranfield University,
Shrivenham
SN6 8LA
United Kingdom

¹a.david@cranfield.ac.uk

²c.j.hargreaves@cranfield.ac.uk

Abstract

Webmail is a convenient way of accessing emails via a web browser on any computer connected to the Internet and it has gained popularity amongst Internet users. Many webmail service providers offer a free email service where users can set up an email account online by supplying their personal details and choosing a preferred username.

Email artefacts such as usernames, aliases, message subject and body may be useful in a digital investigation and thus require recovery and analysis. Unlike client based email software where a user's messages are stored locally on the hard disk, webmail messages are stored remotely on the webmail provider's servers, potentially making it difficult for digital investigators to obtain relevant artefacts. However, since webmail is accessed through a browser and browsers leave their own artefacts, it may be possible to recover artefacts that may be useful in investigations.

This paper discusses certain artefacts that can be left on a user's hard disk as a result of using Hotmail. For instance, artefacts that could be used to infer when an email account was created and the details supplied at set up; details of exchanged emails such as who a user sent an email to, when the email was sent and whether it was replied to; full or partial contents of the email; details of contacts that had been added, edited, deleted or restored by the account user.

The experiments are carried out on Hotmail using Firefox 9 and involve the analysis of the various file formats used by Firefox as well as their evidential value. The research also involves a multi-tool analysis technique which is necessary due to the differences in the format of artefacts recovered and to ensure the accurate interpretation of data. A hex editor, SQLite analysis tool, standalone JSON viewer, and a cache analysis tool are some of the tools identified as useful and are discussed in this paper.

1.0 Introduction

Webmail has gained popularity amongst users as a convenient way of accessing emails through a web browser on any computer connected to the Internet [1]. A range of service providers (e.g. Yahoo, Lycos and Gmail) offer free webmail to users.

The identification and analysis of webmail artefacts is useful in digital investigations because evidence of criminal activity (e.g. the Anderson-Enron case [2]), or innocence of an individual [3] could be found in such artefacts. The forensic analysis of webmail artefacts has been described as a challenge for investigators because messages are stored remotely by the service provider thereby making it less accessible to digital investigators [1], [4]. However, as webmail is accessed through a web browser, and browsers leave their own artefacts, some webmail artefacts may be recoverable.

This paper discusses the types and evidential value of artefacts that may be recovered from a user's hard disk as a result of Hotmail use. This paper makes the following contributions: it identifies the artefacts that exist as a result of Hotmail use in Firefox 9; it documents the artefacts and discusses their volatility. It also shows the possibility of making inferences about user actions and establishing a timeframe for the observed actions.

2.0 Background

2.1 Overview

This section provides a background to Email analysis and Hotmail, briefly discussing related research.

2.2 Existing work on Email analysis

Email has been described as “one of the most important vehicles for criminal activity” [5] and artefacts from emails have also been recognised as important and often crucial to digital investigations as they can contain a wealth of information that can advance a case [6]. Email architecture and the forensic analysis of client and web based email artefacts has been researched by [7–9] amongst others.

Jones et al. [10] researched the reconstruction of client based email activity and analysed the different formats used to store the emails. This was achieved using commercial and open source tools and the conclusion was that the recovery and analysis of such artefacts is possible however, some storage formats may not be recognised by certain tools.

Al-Zarouni [11] discussed the issues associated with tracing header information in emails i.e. determining message source and destination by detecting fake email addresses and whether a message was sent via a browser or client software.

Webmail was identified as harder to trace for a number of reasons including a user's ability to provide false details when creating an account [7]. Al-Zarouni [11] came to the conclusion that email investigations are complex and in a similar view to [8] suggests techniques such as server side investigation as means of determining the source of a message. Although some focus has been from a client based email perspective [7], [12], a number of researchers have shown interest in webmail [13–15].

One example of a webmail service is Hotmail which is from Microsoft. It is one of the popular webmail services and has several features such as contact listing, emailing, instant messaging, and managing user profile as illustrated in Figure 1.

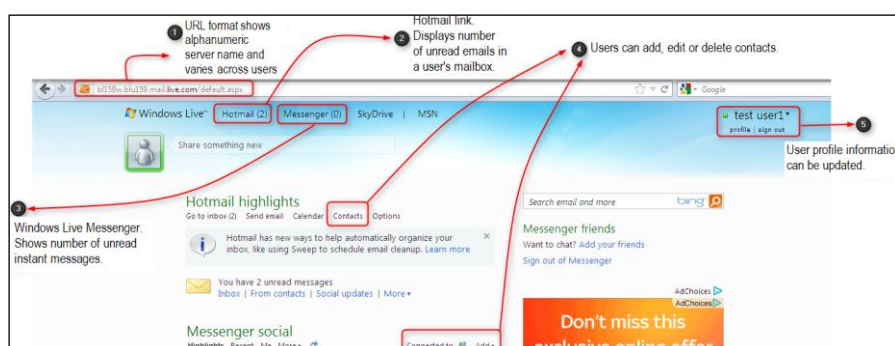


Figure 1—An Illustration of Hotmail home page showing links to the main features.

Webmail has been identified as a relevant aspect of digital investigations and at the same time a challenge to investigators in terms of availability and recovery of artefacts. Eleutério [16] carried out experiments to determine the relationship between web browsers and the webmail artefacts that can be recovered from a user's hard disk and concluded that it is dependent on the browser type and chosen webmail service provider. Eleutério [16] provided tabulated summaries of methods and the type of artefacts recovered from different web browsers including Firefox but did not provide full details of location and format of artefacts investigators need to look out for e.g. JSON or SQLite format artefacts.

2.5 Summary

This section has shown various approaches taken by researchers investigating email evidence and its relevance to digital investigations. The related work shows that there has been more research into client based email. This highlights the need for more research into webmail artefacts as factors like browser version updates or webmail service provider updates could affect the way in which webmail artefacts are left on disk.

3.0 Methodology

3.1 Aim and Scope

As discussed in the previous section, there is limited publication on specific artefacts left by Hotmail. Because artefacts are browser based, to report specifics, focus needs to be on a particular browser. This paper is focused on Hotmail artefacts generated in Firefox 9 web browser on Windows 7.

3.2 Overall methodology

The research method used for this paper was experimental. The experiments conducted involved the generation of data i.e. creation of user email accounts, sending and receiving emails, adding, editing and deleting contacts. Experiments also involved logging all actions performed and an analysis of the disk images.

3.3 Experimental methodology

3.3.1 Data Generation

Two sets of virtual machines (VMs) were created for hypotheses formulation and validation. Windows 7 and Firefox 9 were installed and Hotmail accounts were created on each VM for 4 test users.

Emails were exchanged between the users and the URLs and message content were recorded prior to sending a message to recipients using the To, CC and BCC. The second part of the exchange involved reading, replying, deleting messages and viewing the deleted messages in the trash folder. In addition to the message content and URL being recorded: actions such as switching on the VM, launching the browser, user login to email account, email addresses, usernames/aliases, date and time of exchange, message subject and the disk image were also noted. Contacts were added to the user's address book when prompted. An existing contact was viewed, edited, deleted and restored.

3.3.2 Data Analysis

For each step described in Section 3.3.1, the analysis involved a keyword search of disk, analysing the SQLite databases, Firefox cache analysis, and analysing JSON/js files as discussed below:

- *Keyword search:*
Selected keywords recorded at the experimental stage (email addresses, usernames/aliases, etc.) were searched for across the disk image using X-Ways Forensics. Keyword search returned results from live and deleted files and provided an insight as to other parts of the disk e.g. unallocated space, where artefacts are located. There are limitations associated with keyword searching for instance, data not stored as plain ASCII or Unicode on disk (e.g. compressed files in the Firefox cache) were not found.

- *SQLite databases:*

SQLite databases in the Firefox profile folder were analysed for Hotmail artefacts using sqlite3Explorer (from: <http://www.singular.gr/sqlite/>). The complexity of the SQLite file format means that raw data from the keyword search alone does not provide easy to interpret structured results. Analysing the databases separately provides a more structured interpretation of the data e.g. human readable dates and times. Queries were constructed based on the information logged during the experiments, to help filter the records and the following databases were searched: places.sqlite and formhistory.sqlite.

Places.sqlite is used by Firefox to store information such as browsing history, downloads and bookmarks. Places.sqlite is located by default in the user's Firefox profile in Windows 7: **\Users\username\AppData\Roaming\Mozilla\Firefox\Profiles\###.default** (where ### is a string of alphanumeric characters). Places.sqlite contains a number of tables however, the moz_places and moz_historyvisits tables are of particular interest as they contain visited URLs with dates and times of visit [17], and session ID which makes it possible to group actions that occurred within the same session [18].

Formhistory.sqlite holds a set of name/value pairs and manages the autocomplete service in Firefox [19]. It stores all text entries made in forms on web pages with the exception of password fields [20], including the date and time the entry was first used and last used. Formhistory.sqlite is located in the user's Firefox profile in Windows 7: **\Users\username\AppData\Roaming\Mozilla\Firefox\Profiles\###.default** and it contains the moz_formhistory table which stores this information so that the values and field names held correspond to specific fields they were submitted to. Thus when a web page is revisited, the user is shown options of previously entered values e.g. when completing the 'To, CC or BCC' fields on an email compose page. In some cases, several entries may exist for a field such as username but with different values [19]. Formhistory.sqlite was analysed for artefacts relating to form entries such as user login, adding contacts to address books, and entering email addresses in the recipient fields of a compose page.

- *sessionstore.js:*

Sessionstore.js stores a user's browser session data including any open windows and tabs, page views (minimised, maximised), form data, session cookies, closed tabs/windows, start and "last update" times. Sessionstore.js is located in the user profile folder **\Users\username\AppData\Roaming\Mozilla\Firefox\Profiles\###.default** and enables session restoration in the event of an unexpected browser shutdown e.g. crashes,

browser or add-on installation and updates [21]. When a session is restored, the windows and tabs are placed in their original position i.e. specific scroll positions the user has on each page will be displayed. However, if the browser is shutdown properly, Firefox deletes the sessionstore.js file and a new file is created for the new session. Firefox also maintains a sessionstore.bak file as a backup for browser sessions [21]. This is a renamed sessionstore.js file and is also in JSON format with name/value pairs in an array [22]. An excerpt is shown below:

```
{
  "windows": [
    {
      "tabs": [
        {
          "entries": [
            {
              "url": "",
              "title": "",
              "ID": "",
              "docshellID": "",
              "docIdentifier": "",
              "scroll": "X,Y",
              "index": "",
              "hidden": true/false,
              "attributes": {
                "image": ""
              },
              "selected": 1,
              "closedTabs": [],
              "width": 994,
              "height": 984,
              "screenX": 4,
              "screenY": 4,
              "sizemode": "normal",
              "cookies": [
                {
                  "host": "adobe.com",
                  "value": "true",
                  "path": "/"
                },
                {
                  "name": "s_cc",
                  "host": "xyz.com",
                  "value": "",
                  "path": "/",
                  "name": ""
                },
                {
                  "title": ""
                }
              ],
              "selectedWindow": 0,
              "closedWindow": [],
              "session": {
                "state": "stopped",
                "lastUpdate": 1321625114482,
                "startTime": 1321624839684
              }
            }
          ]
        }
      ]
    }
  ]
}
```

It is important to note that it is possible to recover complete or fragmented sessionstore.js files by initiating a search across a hard disk using {"windows":{"tabs":{"entries":{"url": as a header and "startTime": as an approximation of the footer allowing the data to be extracted from a disk image.

Sessionstore.js was analysed using multiple tools. The fragments were extracted using X-Ways and analysed using JsonViewer (a standalone JSON Viewer available from <http://jsonviewer.codeplex.com/>). The aim was to facilitate the deconstruction and interpretation of what the different values mean. In addition to the manual extraction and examination of sessionstore.js artefacts, Internet Evidence Finder (IEF4 v4) was used to validate the results. However, this only provided the URLs and did not give a detailed view of additional elements such as form data and session cookies, unlike JsonViewer and X-Ways.

- *Cache:*

In order to enhance efficiency, Firefox uses the cache to temporarily store objects from websites visited by a user. This allows the browser to fetch such items locally when the website is revisited rather than re-downloading these resources [5]. However, if the cached object is out of date, the browser downloads it again.

In Windows 7, the Firefox cache is located in **\Users\username\AppData\Local\Mozilla\Firefox\Profiles\###.default\Cache**. Exploring the Cache folder shows that it contains four files: a **_CACHE_MAP_** file (index for cache data and metadata); three cache block files: **_CACHE_001_**, **_CACHE_002_**, **_CACHE_003_**; sixteen subdirectories (named 0-9, A-F) which contain other subdirectories that are used to store

objects that are larger than the three cache block files [23–25]. It is important to note that some cached objects are gzipped (the web server can send compressed resources to the browser). Analysing the Firefox cache was achieved using Digital Detective's HstEx v3.8 and NetAnalysis v1.54 was used for reconstructing the cached data.

- *Additional files:*

Besides the above mentioned files, artefacts may also be found in unallocated space and in some system files e.g. pagefile.sys and volume shadow copies. This is often as a result of files being cleared or deleted by an application (such as Firefox) but are still resident on disk until they are overwritten. For example the use of SQLite Write Ahead Logs (WAL) [26],[27]. Search hits returned from unallocated space or system files (pagefile.sys etc.) were extracted and analysed using relevant tools e.g. sqlite3Explorer, JsonViewer.

4.0 Results and Discussion

4.1 Overview

Artefacts of interest in this research include those generated as a result of account creation, sending, receiving and deleting messages, and managing contacts in a user's address book. This section discusses the experimental results for each of these.

4.2 Account creation

As described in Section 3.3, email accounts were created for users on VMs with Windows 7 and Firefox 9 installed. Following on from a keyword search of the VM disk images; a number of artefacts were identified that indicate the creation of a Hotmail account.

4.2.1 Account 'sign up' results

Locating email account creation artefacts involved searching for the recorded email addresses/aliases, user passwords and answers to the security questions. These keywords returned several hits which were further analysed to determine their relevance to this paper. Artefacts were identified in places.sqlite and sessionstore.js. The URL identified for account creation in this case is https://signup.live.com/signup.aspx?wreply=http:%2F%2Fmail.live.com&id=64855&mkt=en-GB&form=WWLMCNA&publ=SIGNINHM&crea=HTML_CIMS017788_Window s+%2b+WL_EN-GB_0X0_134324&lic=1# as shown in Figure 2.

datetime(visit_date/100	url	title	session
2012-01-30 17:01:42	https://signup.live.com/signup.aspx?wreply=http:%2F%2Fmail.live.com&id=64855&mkt=en-GB	Sign up - Windows Live	2
2012-01-30 17:01:43	https://signup.live.com/signup.aspx?wreply=http:%2F%2Fmail.live.com&id=64855&mkt=en-GB	Sign up - Windows Live	2
2012-01-30 17:01:43	https://signup.live.com/signup.aspx?wreply=http:%2F%2Fmail.live.com&id=64855&mkt=en-GB	Sign up - Windows Live	2
2012-01-30 17:02:23	https://signup.live.com/signup.aspx?wreply=http:%2F%2Fmail.live.com&id=64855&mkt=en-GB	Sign up - Windows Live	8
2012-01-30 17:02:25	https://signup.live.com/signup.aspx?wreply=http:%2F%2Fmail.live.com&id=64855&mkt=en-GB	Sign up - Windows Live	8
2012-01-30 17:44:28	https://signup.live.com/signup.aspx?wreply=http:%2F%2Fmail.live.com&id=64855&mkt=en-GB	Sign up - Windows Live	12
2012-01-30 17:44:29	https://signup.live.com/signup.aspx?wreply=http:%2F%2Fmail.live.com&id=64855&mkt=en-GB	Sign up - Windows Live	12
2012-01-30 17:44:29	https://signup.live.com/signup.aspx?wreply=http:%2F%2Fmail.live.com&id=64855&mkt=en-GB	Sign up - Windows Live	12

Figure 2 - Account creation in moz_places filtered by data, time and session ID

More experiments were conducted and a common base URL: <https://signup.live.com/signup.aspx> was identified. To recover additional data, the results in moz_places and moz_historyvisits were sorted by date and time where the URL includes “signup.live.com”. Using the places.sqlite specification [18], the filtered results were grouped by session ID which can be used to determine the last time and date a particular page was visited, thus increasing the chances of identifying visits from the same session. The results from places.sqlite did not provide the details of the user who had signed up thus a keyword search was carried out on disk for “signup.live.com” and hits were returned in sessionstore.js.

The start and end time within the recovered sessionstore.js file in Figure 3 provided a time frame in UNIX millisecond value, for the session during which the Hotmail account was created. The time was interpreted using DCode to obtain the annotated values and was compared to the dates and times in places.sqlite in order to establish a timeline.

The artefacts recovered from sessionstore.js were in JSON file format in a deleted copy of sessionstore.js, in unallocated space and was analysed using JsonViewer to make the information more presentable and manageable. This made it possible to view the individual elements of the file and extract additional session information such as formdata which provided information regarding how and when an account was created as seen in Figure 3.

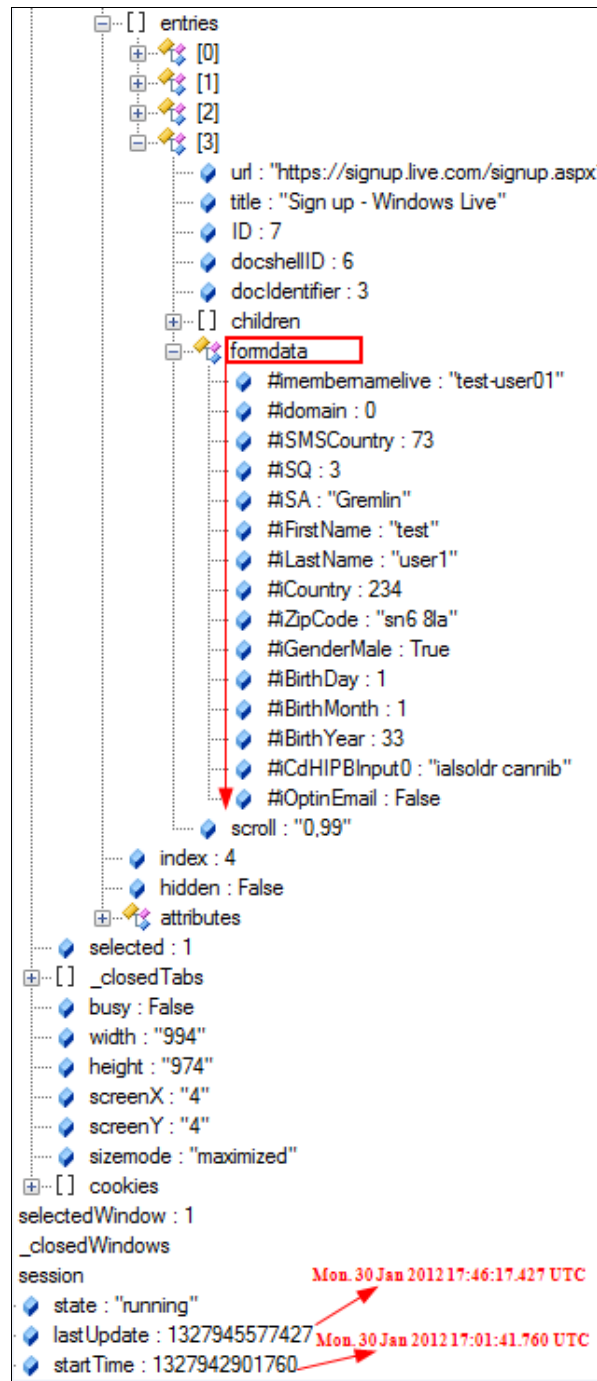


Figure 3 - Screenshot of account set up info when viewed in JsonViewer. As can be seen, the user's sign up details are available in plain text.

As the artefact format (JSON) has been determined and is plain text, it was determined that an automated search across the disk using the known fields which are unique to Hotmail such as *#imembernamelive*, *iBirthYear* etc., will determine if any fragments are on disk (a keyword search using *#iSQ* and *#iSA* returned false positives). The *sessionstore.js* file also contained 26 cookies set for “live.com” hosts which on examination appear to be session cookies as there was no record of cookies set for hosts like “signup.live.com” within *cookies.sqlite*.

While account creation artefacts can be useful, it is important to know that they may not always be available as they may be overwritten with time. More research is required to determine the factors that can affect the length of time this sort of artefact is available before it is overwritten. In addition to availability, the details supplied at account set up may be fictitious and thus unreliable.

4.2.1 Account ‘sign up’ results

In addition to recovering artefacts from account creation, *places.sqlite* was analysed for account access artefacts and the login URL was identified as <https://login.live.com/login.srf?wa=wsignin1.0&rpsnv=11&ct=1329068205&rver=6.1.6206.0&wp=MBI&wreply=http:%2F%2Fmail.live.com%2Fdefault.aspx&lc=2057&id=64855&mkt=en-gb&cbcxt=mai&sncs=1>, prompting a keyword search for “login.live.com” results were returned from fragments of *sessionstore.js* with no session time details. It is important to note that login details can be written to *formhistory.sqlite* as discussed in Section 3.3.2 and an analysis of *formhistory* provided the user email address which was used see Figure 4. The *places.sqlite* URL was compared with the *formhistory.sqlite* entry and a timeline was determined (although the *formhistory* entry was slightly after the *places* entry).

fieldname	value	datetime(firstUsed/1000000)	datetime(lastUsed/1000)	timesUsed	guid
login	test-user01@hotmail.co.uk	2012-02-12 17:37:33	2012-03-07 15:28:25	2	9uwVKDANRIOND9Wg

Figure 4 - Snapshot of user login details from *formhistory.sqlite*

4.3 Mailbox management

This section discusses the results from composing, sending, reading and deleting messages.

4.3.1 Composing messages

A keyword search was carried out using the usernames, subject and content of the emails recorded during the experiment. Several artefacts were found in different areas on disk and in different file formats (SQLite, JSON, and HTML). Fragments of the URLs recorded when composing messages during the experiment were searched for on disk and hits were returned as seen in Figure 5.

0B2FA06E0	7F 81 6C 0C 00 7B 1B 45	01 01 01 01 01 06 25 68	l { E %h
0B2FA06F0	74 74 70 3A 2F 2F 73 6E	31 32 37 77 2E 73 6E 74	ttp://sn127w.snt
0B2FA0700	31 32 37 2E 6D 61 69 6C	2E 6C 69 76 65 2E 63 6F	127.mail.live.co
0B2FA0710	6D 2F 6D 61 69 6C 2F 48	69 70 53 75 63 63 65 73	m/mail/HipSucces
0B2FA0720	73 2E 61 73 70 78 48 6F	74 6D 61 69 6C 6D 6F 63	s.aspxHotmailmoc
0B2FA0730	2E 65 76 69 6C 2E 6C 69	61 6D 2E 37 32 31 74 6E	.evil.liam.721tn
0B2FA0740	73 2E 77 37 32 31 6E 73	2E 01 00 00 0A 62 00 04	s.w721ns. b
0B2FA0750	BA A6 1F 01 E5 30 6D 44	7A 70 54 64 50 50 62 33	q ä0mDzpTdPPb3
0B2FA0760	70 6A 81 18 81 6B 0D 00	81 2B 1B 45 01 01 01 01	pj k + E
0B2FA0770	01 06 25 68 74 74 70 3A	2F 2F 73 6E 31 32 37 77	%http://sn127w
0B2FA0780	2E 73 6E 74 31 32 37 2E	6D 61 69 6C 2E 6C 69 76	.snt127.mail liv
0B2FA0790	65 2E 63 6F 6D 2F 6D 61	69 6C 2F 48 69 70 4C 69	e.com/mail/HipLi
0B2FA07A0	67 68 74 2E 61 73 70 78	3F 48 69 70 4F 70 3D 43	ght.aspx?HipOp=C
0B2FA07B0	6F 6D 70 6F 73 65 26 6E	3D 35 33 38 34 39 32 30	compose&n=5384920
0B2FA07C0	38 32 48 6F 74 6D 61 69	6C 6D 6F 63 2E 65 76 69	82Hotmailmoc.evi

Data is consistent with the moz_places table e.g. URL, page title, reverse host URL.

Figure 5 - Fragment of data consistent with moz_places showing message compose URL

An analysis of places.sqlite returned */HipLight.aspx?HipOp=Compose* as one of the URLs associated with sending messages from a Hotmail account. Account verification is required by Hotmail when a message is sent from an email account for the first time thus the URL fragment */HipLight.aspx?HipOp=Compose* is indicative of a recently created and unverified account.

Hotmail automatically redirects the user to a verification page before a message is sent. Once the account is verified, a */HipSuccess.aspx* URL is seen in the address bar and the user is prompted to send the message. See Figure 5 for screenshots of user Hotmail account prior to and after verification.

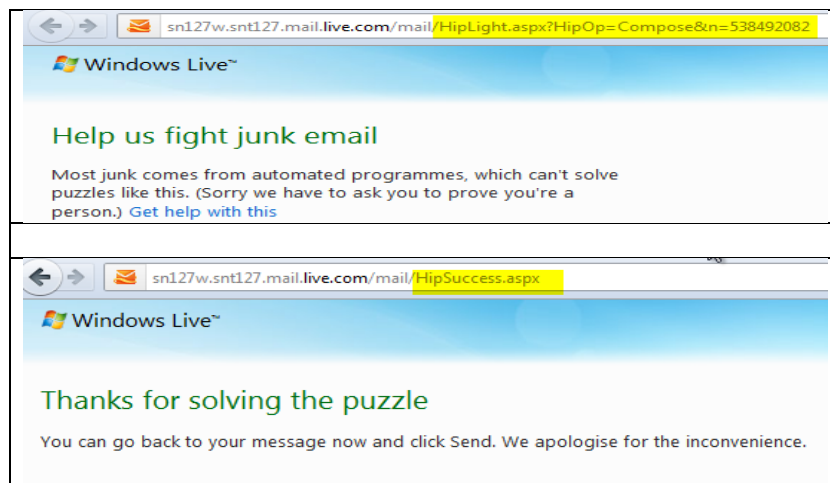
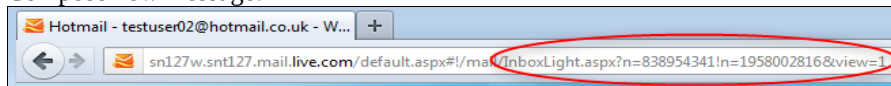


Figure 6 - Account verification request.

Places.sqlite returned both URLs and a third which is consistent with verified accounts. When an account has been used to send several emails, there is a change in the format of the URL. An */InboxLight.aspx?n=* format is seen. This is followed by a string of numbers that changes depending on the action carried out by the user e.g.:

Compose new message:



Reply an email:

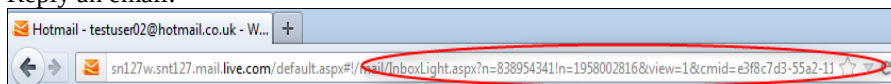


Figure 7 - Changes in URL format depending on past user behaviour.

The page title showing the user email address was available from places.sqlite but actual content of the compose page was not found. Searching sessionstore.js produced other results that were compared to places.sqlite.

4.3.2 Sending, replying and deleting messages

Messages were read, replied, and deleted. Filters were created in places.sqlite to return URLs containing terms such as “SendMessage, EditMessage, ReadMessage, fid=” which had been recorded previously and this returned results and were sorted by date and time. It was observed that when messages are replied, the sender’s email address was embedded in the URL. Two formats were observed and the second appears to be a condensed version of the first as seen in Figure 8.

Format 1:

/P.mvc#!/mail/EditMessageLight.aspx?Action=Reply&AllowUnsafe=True&**ReadMessageId=9e4590ef-c9c0-11e1-97ce-001e0bcc1a3a**&FolderID=00000000-0000-0000-0000-000000000001&Aux=2150%7C0%7C8CF2BE488C9E580%7C%7C0%7C1%7C0%7C0%7C6%7C3%2C11&**SenderEmail=test-user01%40hotmail.co.uk**&n=578919616

Format 2:

/P.mvc#!/mail/InboxLight.aspx?n=193555677&view=1&**cmid=9e4590ef-c9c0-11e1-97ce-001e0bcc1a3a**&**csem=test-user01%40hotmail.co.uk**&cdid=&cfid=1&cau=1&cmad=2154%7C0%7C8CF2BE48935C6B0%7C%7C0%7C1%7C0%7C0%7C6%7C3%2C11&cacc=1

Figure 8 - Recovered URL formats for replied messages (bold highlighting added for emphasis). Note the %40 substituted for the @ symbol.

Searching for the subject, content of exchanged messages and additional terms “SenderEmail, cmid=, csem=” across disk and returned results in sessionstore.js, formhistory.sqlite, and pagefile.sys. A HTML fragment was returned from

pagefile.sys on test user2's disk and contained the reply of a message. As seen in Figure 6, the fragment provided information such as date and time, origin, destination, message content. It also highlights other search terms e.g. fOriginalMessageId, fSubject, fMessageBody.

```
-----41184676334
Content-Disposition: form-data; name="ToolbarActionItem" SendMessage
-----41184676334
Content-Disposition: form-data; name="fOriginalMessageId"
e3f8c7d3-55a2-11e1-9b27-001e0bccddde

-----41184676334
Content-Disposition: form-data; name="fMsgSentState"1
-----41184676334
Content-Disposition: form-data; name="fFrom" testuser02@hotmail.co.uk

-----41184676334
Content-Disposition: form-data; name="cpselectedAutoCompleteTo"
[;test-user01%26%2364%3Bhotmail.co.uk;false;false;0]

-----41184676334
Content-Disposition: form-data; name="fTo" "" <test-user01@hotmail.co.uk>;
-----41184676334
Content-Disposition: form-data; name="cpselectedAutoCompleteCc"

-----41184676334
Content-Disposition: form-data; name="fCc"

-----41184676334
Content-Disposition: form-data; name="cpselectedAutoCompleteBcc"

-----41184676334
Content-Disposition: form-data; name="fBcc"

-----41184676334
Content-Disposition: form-data; name="fSubject" RE: first test email

-----41184676334
Content-Disposition: form-data; name="fMessageBody"

<br>This is my own test message<br>
From: test-user01@hotmail.co.uk<br>
To: testuser02@hotmail.co.uk<br>
Subject: first test email<br>
Date: Sun, 12 Feb 2012 17:56:33 +0000<br><br>

</style>
<div dir="ltr">
This is a test message. <br>
</div></div>
```

Figure 9 – Message fragment recovered from pagefile.sys (highlighted in bold for emphasis)

A sent message fragment was also recovered from sessionstore.js on test user3's hard disk (see Figure 10).

```
{ "url": "http://gfx3.hotmail.com/mail/16.2.7030.0523/ts0a.js",  
  "ID": 109, "docshellID": 101,  
  "referrer": "http://sn139w.snt139.mail.live.com/handlers/resourcespreload.mvc  
?bicild=&view=Hotmail.Compose",  
  "docIdentifier": 109, "scroll": "0,0"},  
  "scroll": "0,0"},  
  "formdata": { "#isFirstPL": "", "//xhtml:div[@id='RichTextEditor']/xhtml:div/xht  
ml:textarea[@name='fMessageBody']": "\nThis is three lines of a test  
message.<br><br>3f65bb87b7a8aaea0ad529cb7c85717c<br><br>22b597f64a  
2484189f4e3b9a3e0bc73a<br><br><br>" }
```

Figure 10 – Fragment recovered from sessionstore.js (line breaks inserted) showing message content.

Artefacts recovered from sessionstore.js and formhistory.sqlite contained the names email addresses of user contacts and corresponds with addition after a message was sent to recipients for the first time (an excerpt is shown below). This is discussed further in Section 4.4. In order to infer user behaviour, artefacts from places.sqlite, sessionstore.js, pagefile.sys and formhistory.sqlite were combined.

Using NetAnalysis, the Cache was examined for Hotmail artefacts. The search returned several CSS, HTML, wyciwyg (what you cache is what you get) entries [28] but no message content. To understand the absence of cached messages, Firefox 9 was launched and used to access Hotmail while monitoring the cache. It was observed that Firefox 9 uses three types of cache (memory, disk and offline cache) to manage user browsing activity.

While logged into the user Hotmail account, the HTTP response headers in the memory cache were captured, examined and seen to contain Cache-Control: no-cache' in the header (see Figure 12). It was determined that Hotmail resources have 'no-cache', no-store, must-revalidate in the response header i.e. the server instructs the browser not to cache the page content locally [15]. When the browser was shut down and restarted, the memory cache was flushed however, the no-cache content was not written to the disk cache. It is possible however for such artefacts to be resident in sessionstore.js or pagefile.sys (as discussed earlier).

```
Client: HTTP  
request-method: GET  
response-head: HTTP/1.1 200 OK  
Cache-Control: no-cache, no-store, max-age=0, must-revalidate,  
no-transform  
Content-Type: text/html; charset=utf-8
```

Figure 11 - Snippet of Hotmail response header in memory cache.

Messages were deleted and viewed in the Hotmail trash folder during the experiment and artefacts found in live and deleted copies of sessionstore.js; live places.sqlite (and deleted places.sqlite-wal) contained the URL (with the message id) of the deleted message. Establishing whether a recovered URL indicates a sent, deleted or draft message was achieved by repeated visits to the different folders and recording the changes to the value after the folder ID (“fid=”) where 1=inbox, 2=deleted, 3=sent, 4=draft, 5=junk. Recovering the URL of a message alone may not be useful if the messageId is unknown to the investigator but may require identifying the message folder and contacting the service provider in order to retrieve the contents from the server. A search of the cache using NetAnalysis did not return any relevant results.

4.4 Contact management

This section discusses the results from adding, editing, deleting and restoring contacts in a user’s address book within Hotmail.

4.4.1 Adding contacts

Contacts were added to the user address book after a prompt by Hotmail after a message was sent. The fragment seen in Figure 12 is from sessionstore.js and shows formdata containing the names entered by the user.

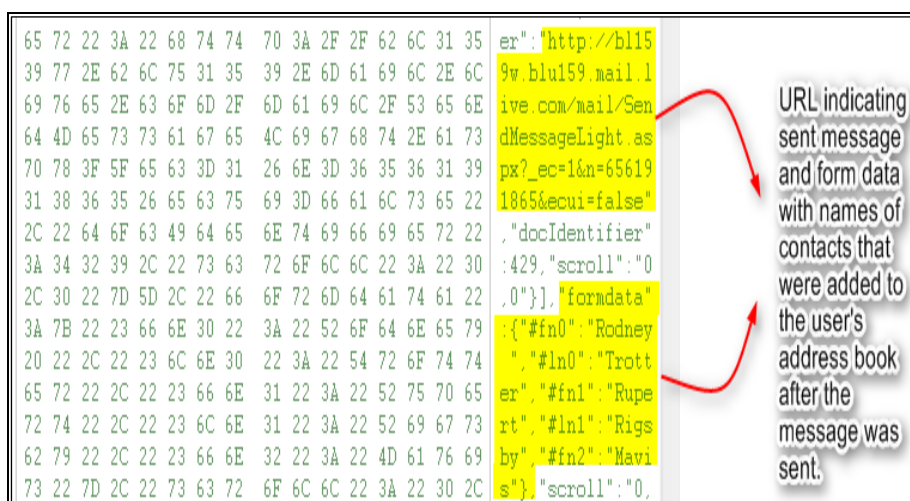


Figure 12 - Sessionstore.js fragment with contact names as formdata values.

4.4.1 Editing, deleting and restoring contacts

When a message is sent to recipients, the email address only is written to formhistory.sqlite however, when a user is prompted to add a contact to the address book, more information can be stored e.g. names, nicknames. As seen in Figure 13, the first entry in the To, Cc and Bcc fields are email addresses while the second has first and last names associated with it.

fieldname	value	datetime(firstUsed/10)	datetime(lastUsed/10)	timesUse	guid
login	testuser02@hotmail.co.uk	2012-03-07 12:04:16	2012-03-07 12:04:29	2	HPbTLQqaRHKNbHPS
fTo	"" <test-user01@hotmail.co.uk>;	2012-03-07 12:13:07	2012-03-07 12:13:07	1	u/8GHlwTTZ2J+FRt
fn0	test	2012-03-07 12:18:27	2012-03-07 12:18:27	1	ZTNHuuFNQm0b7d
ln0	user1	2012-03-07 12:18:27	2012-03-07 12:18:27	1	eoPrR1UYRKHsbFo
fTo	"test user1" <test-user01@hotmail.co.uk>;	2012-03-07 13:32:13	2012-03-07 13:38:10	2	NwpBuLxTSEm0ba2
fCc	"" <nuprigsby@gmail.com>;""	2012-03-07 13:38:10	2012-03-07 13:38:10	1	2U7ckwMMRyyh63ci
fBcc	"" <rodrtter@gmail.com>;	2012-03-07 13:38:10	2012-03-07 13:38:10	1	p6pJDJr5iCY5gb
fn0	Rupert	2012-03-07 13:39:17	2012-03-07 13:39:17	1	Mr5QNI4RAecEpU0
ln0	Rigsby	2012-03-07 13:39:17	2012-03-07 13:39:17	1	f1VR2sqeReWxmsbQ
fn2	Rodney	2012-03-07 13:39:17	2012-03-07 13:39:17	1	JniUjby8RzOfcPi0
ln2	Trotter	2012-03-07 13:39:17	2012-03-07 13:39:17	1	U1Pcx50TOKRyolq
fNameVal	Mavis	2012-03-07 13:49:36	2012-03-07 13:49:36	1	wkcZC1OLRMilhm4R
fNameVal	Winkle	2012-03-07 13:49:36	2012-03-07 13:49:36	1	2sb8aJ1xTiKAuaFA
nicknameVa	Mav	2012-03-07 13:49:36	2012-03-07 13:49:36	1	93p0iz3ERJu9wH7Z
pEmailVal	maviswinkle@yahoo.co.uk	2012-03-07 15:18:21	2012-03-07 15:18:21	1	Eo13BuxwQ3SF6KEq

Figure 13 - Contacts added to user address book as seen in formhistory.sqlite

The fieldname column in formhistory.sqlite shows where the entries in the value column were made. The entries in the table include the user email address used to login to the email account, names and email addresses of recipients. In the fieldname column, when more than one recipient was added to the 'To, Cc or Bcc' fields, they were indexed by appending numbers 0–n (n = maximum number of entries in each field). The 'firstUsed' date is when the entry was first made while 'lastUsed' is the most recent use date. These dates and times correspond with the time logged during the experiments.

For contact editing and deletion, keywords searched for in places.sqlite were: "contact, profile, cid=, edit, delete, and restore". URLs recovered from places.sqlite contained a 'ContactId' which is a string of hexadecimal characters assigned to a contact once it has been added to a Hotmail account. Places.sqlite also contained the page title e.g. test user2's details, Edit details" as seen in Figure 14.

2012-03-07 13:49:38	https://profile.live.com/P.mvc#/cid-148eed47e7501b7c/details/?contactId=703ea366-0000-0000-0000-000000000000&ru=	Mavis Winkle's Details - Windows Live
2012-03-07 13:49:38	https://profile.live.com/P.mvc#/cid-148eed47e7501b7c/details/edit/?contactId=703ea366-0000-0000-0000-000000000000	Edit details - Windows Live
2012-03-07 13:49:38	https://snr127.mail.live.com/default.aspx?ru=contacts	
2012-03-07 13:49:38	https://snr127.mail.live.com/default.aspx?ru=contacts#/mail/ContactMainLight.aspx?n=1697900813	Contacts - Windows Live
2012-03-07 13:49:38	https://snr127.mail.live.com/default.aspx?ru=contacts#/mail/ContactMainLight.aspx?n=2044443346	Contacts - Windows Live
2012-03-07 13:49:38	https://profile.live.com/cid-148eed47e7501b7c/contacts/restore/?ru=https%3a%2f%2fsnr127.mail.live.com%2fmail%2fContact	
2012-03-07 13:49:38	https://profile.live.com/P.mvc#/cid-148eed47e7501b7c/contacts/restore/?ru=https%3a%2f%2fsnr127.mail.live.com%2fmail	Restore deleted contacts - Windows Live
2012-03-07 13:49:38	https://profile.live.com/P.mvc#/cid-148eed47e7501b7c/	test user2's Profile - Windows Live

Figure 14 - Date and time correlation of modified contact in places.sqlite

For restored contacts, the 'ContactId' was present with the page title. However, this information may be unreliable as the ContactId changes when a contact is deleted and restored. However, after sorting the results by date and time, it was possible to match the entries in formhistory to the URLs, page titles, date and time in places.sqlite and these were used to deduce which contacts had been edited or restored (the formhistory entry was slightly ahead of the places entry).

4.5 Summary

This section has shown that artefacts relating to Hotmail activity in Firefox 9 can be recovered from a user's hard disk. Artefacts indicating account setup were recovered from pagefile.sys and sessionstore.js and were identified as volatile as they degrade with time. This section has shown that sessionstore.js is a very useful file as it contains a wealth of information (e.g. URLs, formdata, date and time) that can be extracted and compared or combined with other artefacts on disk and used to reconstruct an event or infer user behaviour. This section has also shown that keyword searching a disk is necessary as it can help capture fragmented artefacts which may not be recoverable by tools which search for artefacts using file signatures. Finally, this section has presented key terms that can be useful when searching for Hotmail artefacts on disk.

5.0 Evaluation

This paper analysed artefacts resident on a hard disk as a result of Hotmail use in Firefox 9 on Windows 7. Other web browsers, webmail service providers and operating systems can be used to conduct similar experiments and are likely to produce varying results as demonstrated by [16]. Due to the rapid version updates, this paper did not cover artefacts that can be recovered from other versions of Firefox (v 10+). In addition, other system files and locations e.g. hyberfil.sys, and the Windows Registry were not examined as a detailed study of their interactions with the web browser is yet to be undertaken but these files may contain artefacts.

Artefacts created as a result of user behaviour such as creating and verifying a Hotmail account, managing messages and contacts were analysed. In particular the volatility of artefacts from account creation was identified however, the length of time it lasts or the factors affecting its availability were not established and would

require further work. Artefacts relating to email exchange and contact management were examined however an in-depth deconstruction of parts of the URL e.g. `messageId` and `ContactId` was not carried out due to unavailability of specifications or naming conventions from Hotmail. Although this paper has shown the possibility of recovering Hotmail artefacts, there is currently insufficient information that can be used to develop a dedicated tool for Hotmail analysis.

Other activities such as using the integrated instant messenger or the document storage and file sharing feature (SkyDrive) within Hotmail have not been explored as it is out of the scope of this paper however, the methodology described in this paper can be applied to these. Finally, the use of multiple tools was necessitated by the difference in the file formats used by Firefox 9 and also to enable the reliable interpretation of recovered artefacts.

6.0 Conclusions & Future Work

This paper presented the results obtained from analysing Hotmail artefacts in Firefox 9. It can be seen that although the recovery of webmail artefacts has been described as a challenge for investigators, it is possible to recover artefacts that can be used to infer user behaviour by exploiting the data left by a browser. Several methods of recovering these artefacts such as keyword searching, analysing SQLite databases and analysing live and deleted `sessionstore.js` were discussed.

This paper identified artefacts of interest related to Hotmail use and discussed methods of recovering and interpreting these artefacts. It has shown that although these artefacts can be written to disk in various ways, timing of recovery is important as they may be overwritten. This paper has shown the possibility of automating the search for artefacts by incorporating the identified key terms identified in the artefacts into scripts in order to scan disks for fragments of artefacts. This is particularly useful in cases where they have been partially overwritten and the file headers/footers are unavailable.

Recovering Hotmail artefacts is an area that needs to be researched further. As mentioned in Section 5.0, the specification for Hotmail URLs is yet to be determined (these may give a better understanding of the artefacts and may establish links between recipients). In addition, artefacts from other integrated features of Hotmail are yet to be identified and interpreted.

Artefacts from Hotmail on other web browsers and operating systems also requires further research although a similar study was conducted by Eleutério [16]. Nevertheless, this paper has shown that there is the potential to build social networks using URLs from `places.sqlite` and user contact information from `formhistory.sqlite`.

References

- [1] D. Schatzmann, W. Mühlbauer, T. Spyropoulos, and X. Dimitropoulos, "Digging into HTTPS," *Proceedings of the 10th annual conference on Internet measurement - IMC '10*, pp. 322–326, Jan. 2010.
- [2] S. Mason, *E-mail and the Internet at Work: A Concise Guide to Legal Issues*. 2004.
- [3] C. Steel, *Windows Forensics: The Field Guide for Corporate Computer Investigations*. .
- [4] M. Schwartz, "Client and Web - Email analysis," pp. 1–9, Feb. 2006.
- [5] E. Casey, "Digital evidence and computer crime: Forensic Science, Computers and the Internet.," *Elsevier Academic Press*, Jan. 2004.
- [6] W. G. Kruse and J. G. Heiser, *Computer forensics: incident response essentials*. 2002.
- [7] F. Cohen, "Bulk Email Forensics," *Advances in Digital Forensics V*, vol. 306, pp. 1–17.
- [8] T. M. Banday, "Techniques and Tools for Forensic Investigation of Email," *IJNSA*, vol. 6, pp. 1–15.
- [9] M. T. Banday, F. A. Mir, J. A. Qadri, and N. A. Shah, "Analyzing Internet e-mail date-spoofing," *Digital Investigation*, vol. 7, no. 3–4, pp. 145–153, Jan. 2011.
- [10] K. Jones, R. Bejtlich, and C. W. Rose, *Real Digital Forensics: Computer Security and Incidence Response*. 2005.
- [11] M. Al-Zarouni, "Tracing E-mail Headers," pp. 1–11, Nov. 2004.
- [12] A. Persaud and Y. Guan, "A Framework for Email Investigations," *Advances in Digital Forensics 2005*, pp. 1–12.
- [13] J. Fan, X. Wu, S. Zhu, and Y. Xia, "Research and Implementation of Web Mail Forensics System," *2011 International Conference on Management and Service Science*, pp. 1–4, Sep. 2011.
- [14] B. Taylor, "Sender reputation in Large Webmail Service," *CEAS 2006 - Third Conference on - Email and Anti-Spam*, pp. 1–6, Jan. 2006.
- [15] S. Kuang and B. Boucher, "Email evidence: Now you see it, now you don't!," *Forensic Focus*, Aug. 2011.
- [16] P. M. Eleutério and J. D. Eleutério, "Webmail evidence recovery: a comparison among the most used Web browsers and webmail services," *ICoFCS 2011*, pp. 182–189, Jan. 2011.
- [17] D. Morill, "Understanding Firefox and SQLite Tables for Computer Forensics Analysis," *InfoSec Institutes Resources*, Jan. 2011.
- [18] MDN, "Querying Places," *Mozilla Developer Network*. [Online]. Available: https://developer.mozilla.org/en/Querying_Places. [Accessed: 22-Jan-2012].
- [19] MDN, "nsIFormHistory," *Mozilla Developer Network*. [Online]. Available: https://developer.mozilla.org/en/XPCOM_Interface_Reference/nsIFormHistory2. [Accessed: 22-Jan-2012].
- [20] E. Fontarensky, I. Martin, M. Picod, and J. Bursztein, "Doing forensics in the cloud age - OWADE: beyond files recovery forensic," *Black Hat USA 2011*, pp. 1–23, Jan. 2011.

- [21] MozillaKnowledgeBase, "Session Restore," *mozillaZine*. [Online]. Available: http://kb.mozillazine.org/Session_Restore. [Accessed: 22-Jan-2012].
- [22] H. Parsonage, "Web Browser Session Restore Forensics - A valuable record of a user's internet activity for computer forensic examinations."
- [23] J. Ritchie, "FfFormat - Description of the format of Firefox cache files," *firefox-cache-forensics wiki*, Apr. 2012.
- [24] J. Ritchie, "FfCacheRead - Description of how to read and extract data from the Firefox Cache.," *firefox-cache-forensics wiki*, Apr. 2012.
- [25] J. Ritchie, "Firefox Cache Format and Extraction," *Forensic Focus*, Apr. 2012.
- [26] A. Caithness, "The Forensic Implications of SQLite's Write Ahead Log," *Digital Investigation - Digital evidence: if it's there, we'll find it*, Jun. 2012.
- [27] M. T. Pereira, "Forensic analysis of the Firefox 3 Internet history and recovery of deleted SQLite records," *Digital Investigation*, vol. 5, no. 3–4, pp. 1–11, Apr. 2009.
- [28] P. Andrews and C. Wilson, "Firefox wyciwyg Cache Entries," *NetAnalysis User Guide*, 2012. [Online]. Available: <http://kb.digital-detective.co.uk/display/NetAnalysis1/Firefox+wyciwyg+Cache+Entries>. [Accessed: 22-Feb-2012].

Appendix E: Larger Representation of Modules

E.1

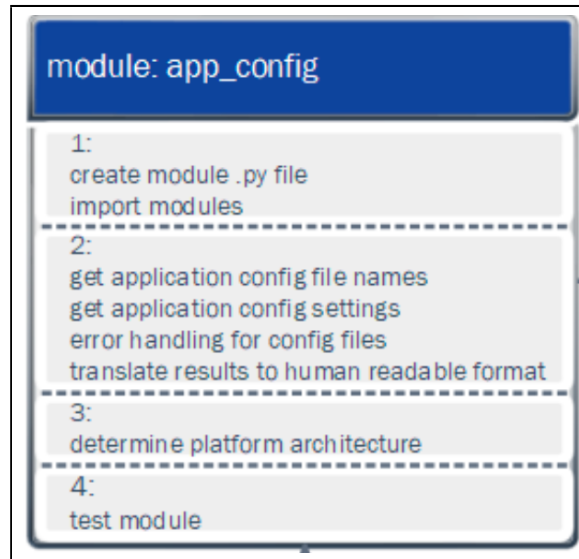


Figure E.1: app_config

E.2

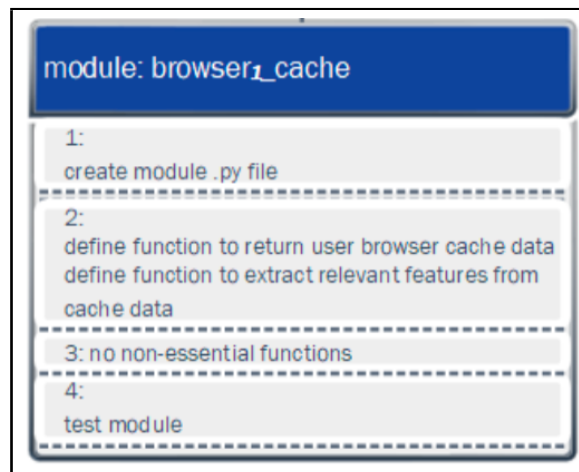


Figure E.2: browser₁_cache

E.3.

E.3

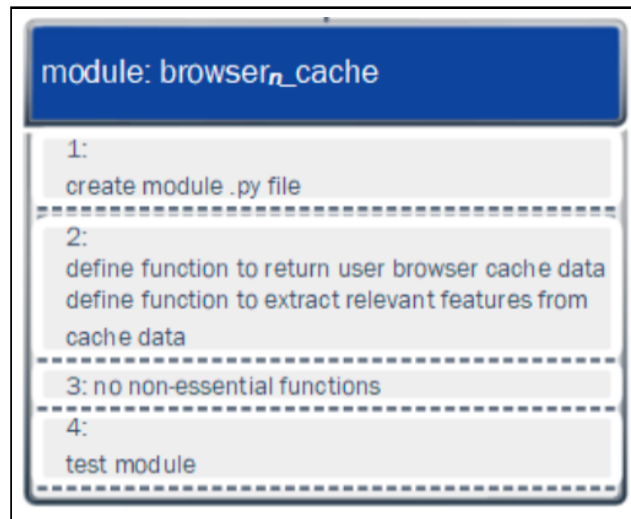


Figure E.3: browser_n_cache

E.4

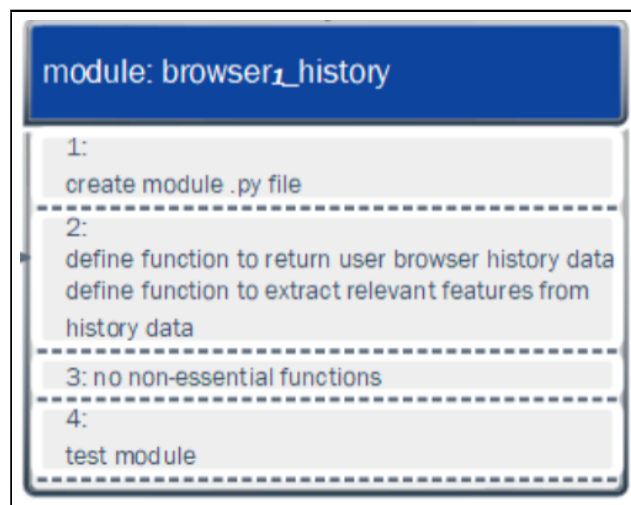


Figure E.4: browser₁_hist

E.5

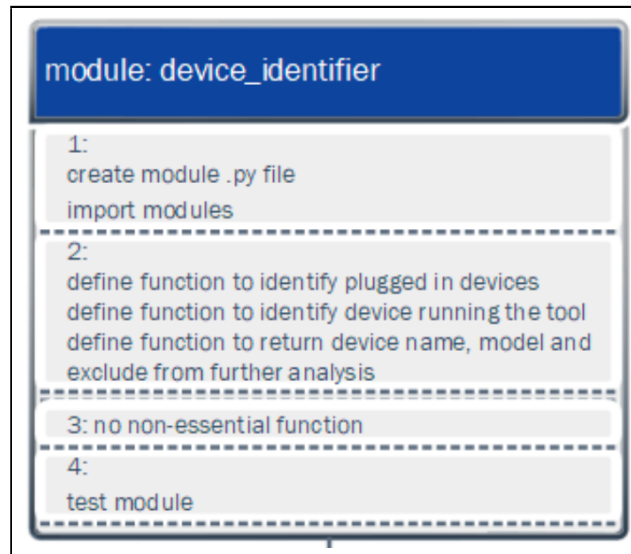


Figure E.5: device_id

E.6

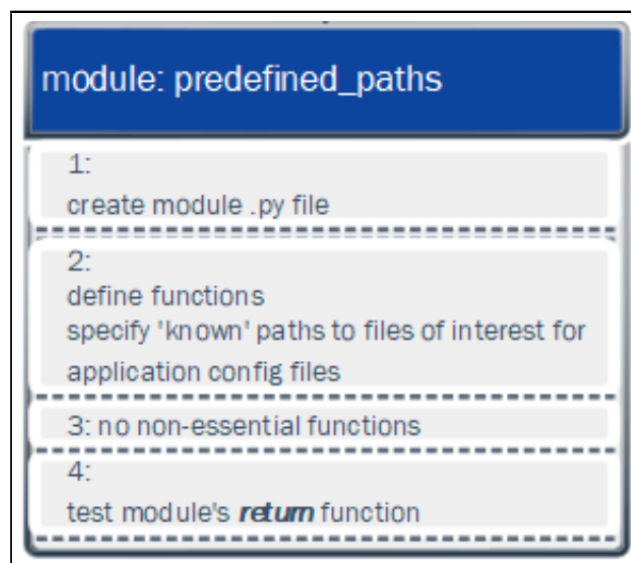


Figure E.6: predefined_paths

E.7.

E.7

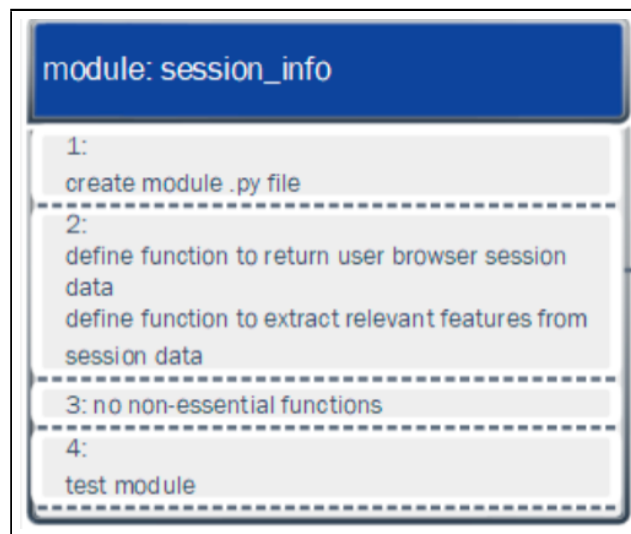


Figure E.7: session_info

E.8

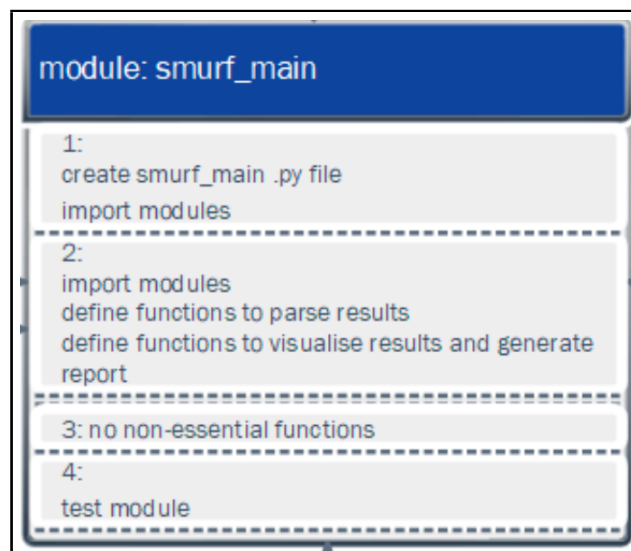
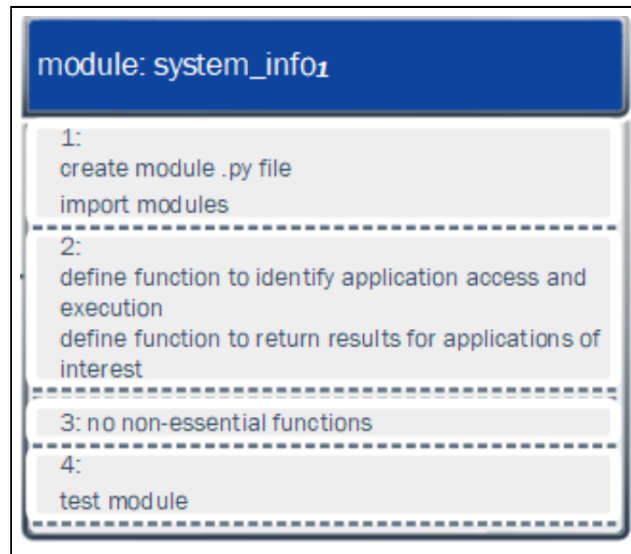
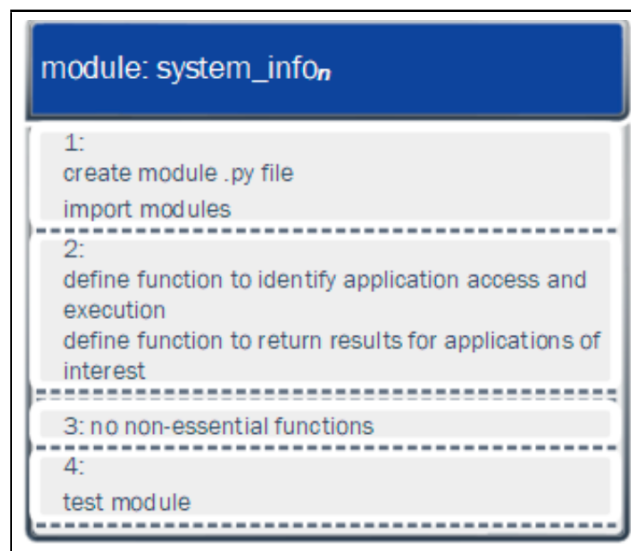


Figure E.8: smurf_main

E.9Figure E.9: system_info¹**E.10**Figure E.10: system_infoⁿ

Appendix F: Ethics Approval

F.1

Note:

The research title changed but the scope did not.



5 October 2019

Dear Miss David ,

Reference: CURES/9355/2019

Title: Feature extraction and matching: a reliable method for contextualising web apps artefacts

Thank you for your application to the Cranfield University Research Ethics System (CURES).

We are pleased to inform you your CURES application, reference CURES/9355/2019 has been reviewed. You may now proceed with the research activities you have sought approval for.

If you have any queries, please contact CURES Support.

We wish you every success with your project.

Regards,

CURES Team