An Analysis of Hotmail Artefacts in Firefox 9

Anne David[1], Christopher Hargreaves[2]

Centre for Forensic Computing
Department of Engineering and Applied Science
Cranfield University,
Shrivenham
SN6 8LA
United Kingdom
[1]a.david@cranfield.ac.uk
[2]c.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.

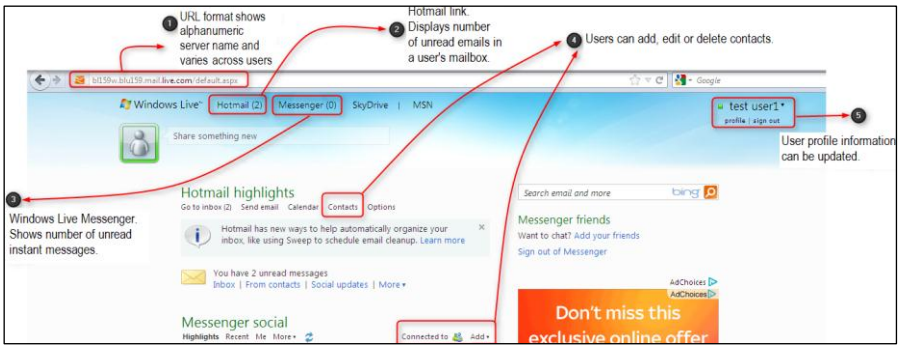


Figure 1–An Illustration of Hotmail home page showing links to the main features.

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*****App Data\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*****App Data\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*\App Data\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*\App Data\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

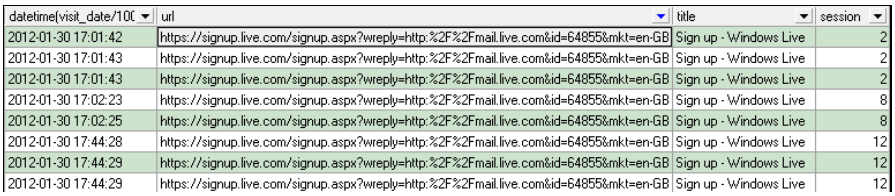
Artefacts of interest in this research include those generated as a result of account creation, sending, receiving and deleting messages, and managing contacts in a user's address book. This section discusses the experimental results for each of these.

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_Windows+%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.

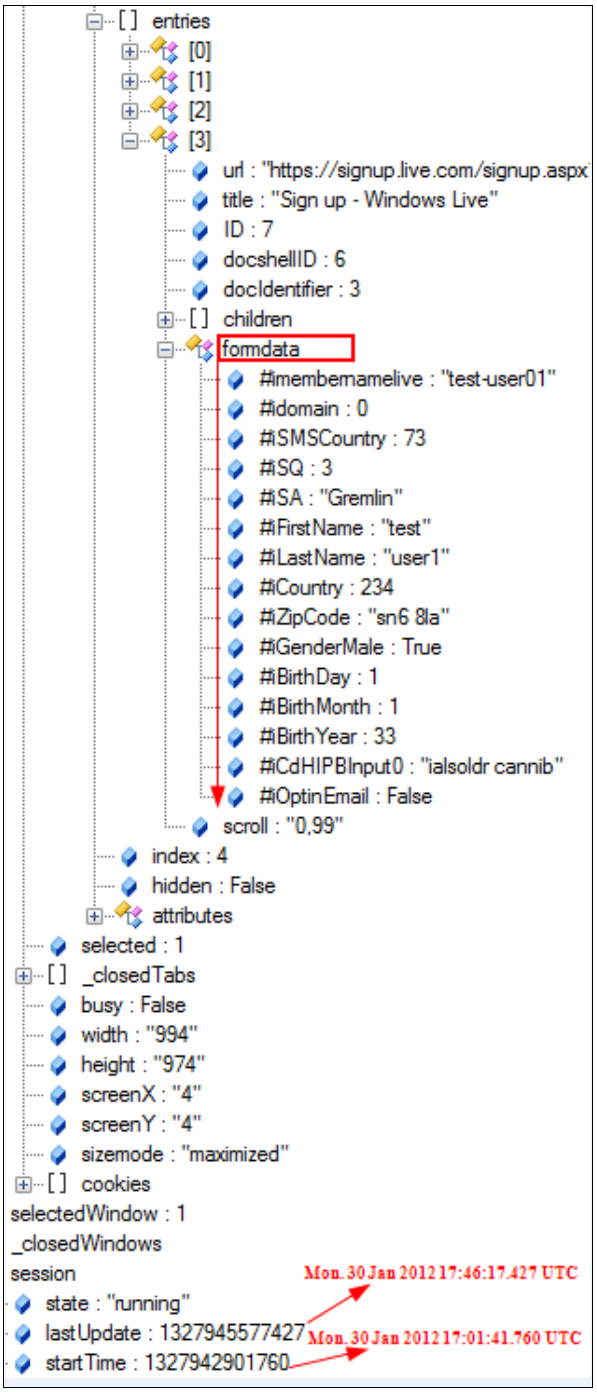


Figure 3 - Screenshot of account set up info when viewed in JsonViewer. As can be seen, the user's sign up details are available in plain text.

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&snsc=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).

fieldnam	value	datetime(firstUsed/1000000	datetime(lastUsed/1000	timesUse	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.

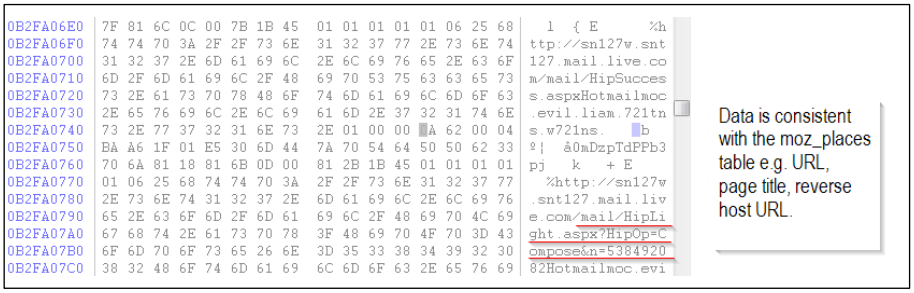


Figure 5 - Fragment of data consistent with moz_places showing message compose URL

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.

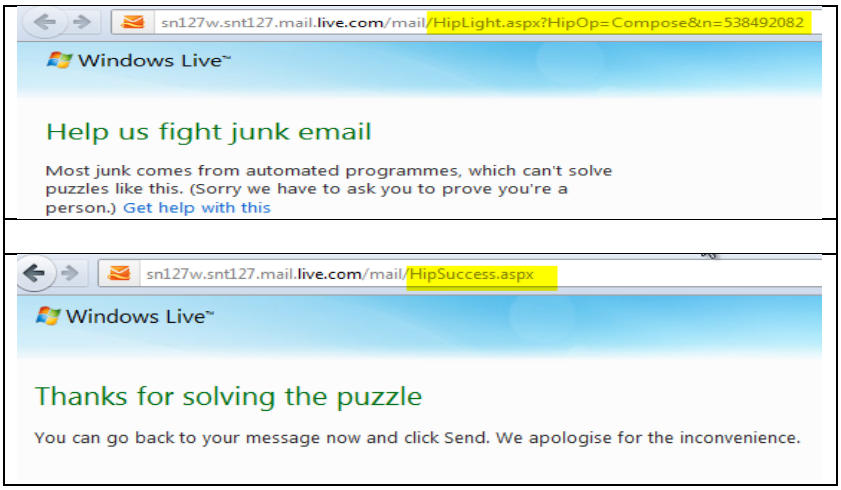
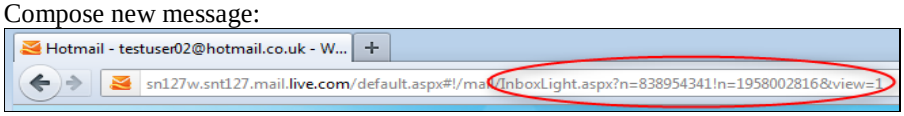


Figure 6 - Account verification request.

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:

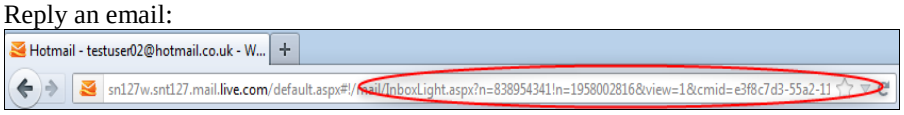


Figure 7 - Changes in URL format depending on past user behaviour.

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**"

This is my own test message
From: test-user01@hotmail.co.uk
To: testuser02@hotmail.co.uk
Subject: first test email
Date: Sun, 12 Feb 2012 17:56:33 +0000

</style>

This is a test message.

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/xhtml:textarea[@name='fMessageBody']":"\nThis is three lines of a test message.

3f65bb87b7a8aaea0ad529cb7c85717c

22b597f64a2484189f4e3b9a3e0bc73a

"}

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.

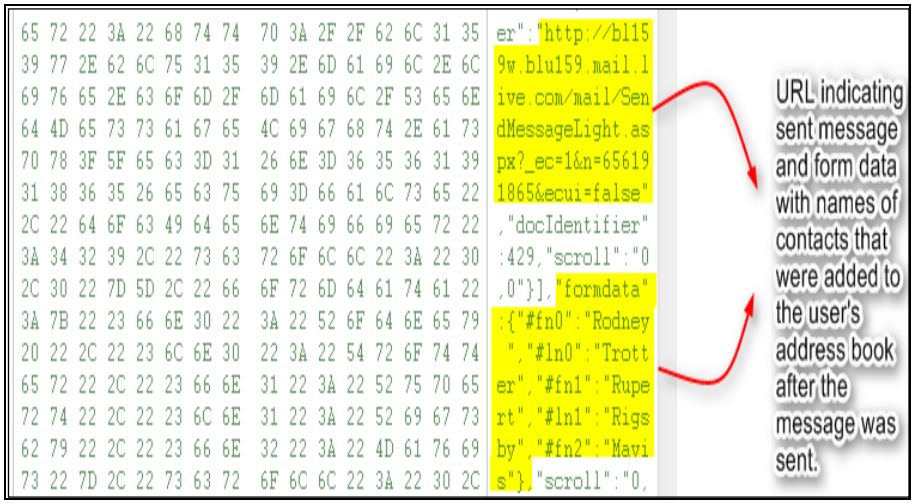


Figure 12 - Sessionstore.js fragment with contact names as formdata values.

4.4.1 Editing, deleting and restoring contacts

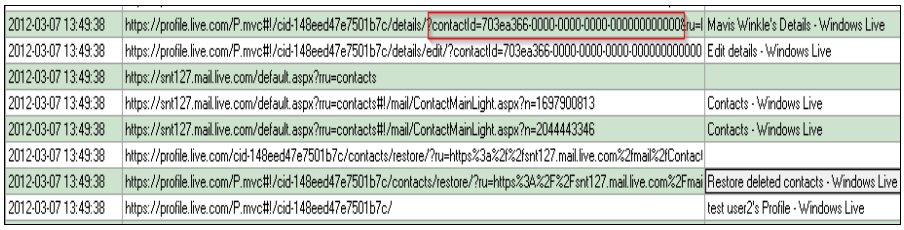
When a message is sent to recipients, the email address only is written to formhistory.sqlite however, when a user is prompted to add a contact to the address book, more information can be stored e.g. names, nicknames. As seen in Figure 13, the first entry in the To, Cc and Bcc fields are email addresses while the second has first and last names associated with it.

fieldnam	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	HPbTLoqaRHKNbHPS
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	ZTNHuuFNQrmObi7d
ln0	user1	2012-03-07 12:18:27	2012-03-07 12:18:27	1	eoPrR1UYRtKHsbFo
fTo	"test user1" <test-user01@hotmail.co.uk>;	2012-03-07 13:32:13	2012-03-07 13:38:10	2	NwpBuLxTSEmr0ba2
fCc	"" <ruprigsby@gmail.com>;""	2012-03-07 13:38:10	2012-03-07 13:38:10	1	2U7ckwMMRyyh63ci
fBcc	"" <rodtter@gmail.com>;	2012-03-07 13:38:10	2012-03-07 13:38:10	1	p6pJDJrrSrCfY5gb
fn0	Rupert	2012-03-07 13:39:17	2012-03-07 13:39:17	1	Mr5QftI4RAecEpU0
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	JniUjbY8RsOFcPiO
ln2	Trotter	2012-03-07 13:39:17	2012-03-07 13:39:17	1	U1Pcx50iTOKRyolq
fNameVal	Mavis	2012-03-07 13:49:36	2012-03-07 13:49:36	1	wkcZC1OLRMilhm4R
lNameVal	Winkle	2012-03-07 13:49:36	2012-03-07 13:49:36	1	2sb8aJ1xTtKAuaFA
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	Eo13BuxtQ3SF6KEq

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=l	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://snt127.mail.live.com/default.aspx?rru=contacts	
2012-03-07 13:49:38	https://snt127.mail.live.com/default.aspx?rru=contacts#!/mail/ContactMainLight.aspx?n=1697900813	Contacts - Windows Live
2012-03-07 13:49:38	https://snt127.mail.live.com/default.aspx?rru=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%2fsnt127.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%2Fsnt127.mail.live.com%2Fmai	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, "nslFormHistory," *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].

An analysis of Hotmail artefacts in Firefox 9

David, A.

2012-09-06T00:00:00Z