



This is a repository copy of *Forensic investigation of cooperative storage cloud service: Symform as a case study*.

White Rose Research Online URL for this paper:  
<http://eprints.whiterose.ac.uk/128357/>

Version: Accepted Version

---

**Article:**

Teing, Y.-Y., Dehghantanha, A., Choo, K.-K.R. et al. (2 more authors) (2017) Forensic investigation of cooperative storage cloud service: Symform as a case study. *Journal of Forensic Sciences*, 62 (3). pp. 641-654. ISSN 0022-1198

<https://doi.org/10.1111/1556-4029.13271>

---

This is the peer reviewed version of the following article: Teing, Y.-Y., Dehghantanha, A., Choo, K.-K. R., Dargahi, T. and Conti, M. (2017), Forensic Investigation of Cooperative Storage Cloud Service: Symform as a Case Study. *J Forensic Sci*, 62: 641–654, which has been published in final form at <https://doi.org/10.1111/1556-4029.13271>. This article may be used for non-commercial purposes in accordance with Wiley Terms and Conditions for Self-Archiving.

**Reuse**

Unless indicated otherwise, fulltext items are protected by copyright with all rights reserved. The copyright exception in section 29 of the Copyright, Designs and Patents Act 1988 allows the making of a single copy solely for the purpose of non-commercial research or private study within the limits of fair dealing. The publisher or other rights-holder may allow further reproduction and re-use of this version - refer to the White Rose Research Online record for this item. Where records identify the publisher as the copyright holder, users can verify any specific terms of use on the publisher's website.

**Takedown**

If you consider content in White Rose Research Online to be in breach of UK law, please notify us by emailing [eprints@whiterose.ac.uk](mailto:eprints@whiterose.ac.uk) including the URL of the record and the reason for the withdrawal request.



[eprints@whiterose.ac.uk](mailto:eprints@whiterose.ac.uk)  
<https://eprints.whiterose.ac.uk/>

# Forensic Investigation of Cooperative Storage Cloud Service: Symform as a Case Study

Teing Yee Yang<sup>a</sup>, Ali Dehghantanha<sup>a</sup>, Kim-Kwang Raymond Choo<sup>b</sup>,  
Mauro Conti<sup>c</sup>, Tooska Dargahi<sup>d</sup>

<sup>a</sup>*School of Computing, Science and Engineering, University of Salford, United Kingdom*

<sup>b</sup>*Information Assurance Research Group, University of South Australia, Australia*

<sup>c</sup>*Department of Mathematics, University of Padua, Italy*

<sup>d</sup>*Department of Computer Engineering, West Tehran Branch, Azad University, Iran*

---

## Abstract

Researchers envisioned the Storage as a Service (StaaS) as an effective solution to the distributed management of digital data, since it provides an inexpensive and reliable online storage which is accessible by different types of computer devices (e.g., mobile and desktop). However, the proliferation of cloud storage services has fuelled concerns over privacy and security (e.g., dissemination of malware and unauthorised access of private information). Together with the growing of StaaS proposals, cloud storage is attracting the attention of research, although cooperative storage cloud forensic is relatively under-explored as this is a fairly new concept. Using Symform as a case study, we seek to determine the data remnants from the use of cooperative cloud storage services. In particular, we consider both mobile devices and personal computers running various popular operating systems, namely Windows 8.1, Mac OS X Mavericks 10.9.5, Ubuntu 14.04 LTS, iOS 7.1.2, and Android KitKat 4.4. Potential artefacts recovered during the research include data relating to the installation and uninstallation of the cloud applications, log-in to and log-off from symform account using the client application, file synchronisation (download, upload, and delete), as well as their timestamp information. This research contributes to an in-depth understanding of the types of terrestrial artefacts that are likely to remain after the use of cooperative storage cloud on client devices.

*Keywords:* Cloud Forensics, Computer Forensics, Mobile Forensics, Symform Analysis.

## 1. Introduction

Cloud computing is, arguably, one of the most discussed computing paradigms in recent years, due to its popularity among individual consumers and organisations. Gartner [1] forecasted that the cloud computing market will hit US\$250 billion by 2017 as cloud adoption increases in organisations. International Data Corporation (IDC) [2] also published a similar forecast, which indicated that the worth of the cloud computing market will exceed US\$107 billion and drive 17% of the IT product spending by 2017.

Despite the promising economical and technological opportunities, cloud storage services are also being exploited by criminals, both traditional and cyber ones [3], in several ways such as information theft [4, 5, 6, 7] or distributing copyright or illegal materials. Cloud servers have also been exploited as an avenue for conducting denial of service attacks [8], cracking passwords [9], and hiding criminal tracks [5].

Since the introduction of cloud computing, scholars have pointed out the challenges in conducting forensic investigations involving the use of cloud storage services. These challenges mostly come from the lack of physical access to digital artefacts over the servers spanning across multiple jurisdictional areas, as well as integrity of data artefacts (e.g., log files) provided by the Cloud Service Providers (CSPs) [10, 11, 12, 13, 14, 15, 16, 17]. Even if the evidence could be identified, it could be illegal to access the raw log data that contains records of multiple users in a multi-tenancy cloud environment [18]. The wide range of mobile devices [19] and the use of encryption by CSPs or individuals [12] further complicate cloud forensic investigations.

To ensure timely and cost-effective responses to investigating cloud-related incidents, it is imperative that forensic examiners are cognisant about different types of cloud products (or have access to such information), as well as the potential artefacts detectable on each platform [20, 21, 22, 14, 23]. Depending on the cloud storage solution in use, evidence (e.g., logs) of cloud usage could be recovered from the client devices [21, 22, 24, 25, 26, 27, 28]. Hence, we seek to identify potential terrestrial artefacts that may remain after the use of Symform cooperative storage cloud [29].

### 1.1. Background

A cooperative storage cloud is a fully decentralized storage model that aggregates multiple cooperating nodes (e.g., game consoles, laptop, and personal computers) in the cloud environment [30]. By eliminating the need

for storage servers, it offers a low cost, highly reliable, and secure cloud storage system. Symform (Symbiotic Storage Platform) claims to be the first cooperative storage cloud provider and reportedly has active users in 138 countries [31]. In 2014, Symform received the *Best Cloud Storage Solution* award in the 4th Annual Cloud Computing World Series [32]. The free service allows users to store up to 10GB of storage. Users may acquire additional space (up to 200GB) by donating a portion of unused space on their local hard drive to Symform network for other users to store data. The users, in return, receive 50% cloud storage space of the total hard drive space contributed. Alternatively, users are able to purchase additional space as low as \$10.00/month for 100GB storage [29].

To ensure secure storage of users' data across its network of contributed devices, Symform encrypts each sync/backup folder using 256-bit AES encryption and breaks down each file into 64MB small data blocks. Then, the blocks are shredded further into sixty-four 1MB encrypted fragments before geographically distributing those fragments in parallel across 96 different nodes in the Symform network [33]. The index and encryption key of the transmitted data reside on the file owner's computer. When a file is uploaded, Symform's *Cloud Control*, the cloud management service hosted by Amazon Web Services (EC2), will constantly monitor all the peer devices containing fragments of the file data blocks. In cases when a peer device is down or at risk of failure, the system will automatically regenerate the fragments at risk and distribute them to another contributing device in Symform network, thereby increasing reliability and integrity of the data storage.

Similar to other cloud storage services, Symform service can be accessed using a web browser (but limited to the downloading, viewing and deleting the files) or a client application available for devices running Microsoft Windows, Apple Mac OS X, Linux, Apple iOS, Android, and Blackberry. Unlike most cloud storage services, Symform allows users to selectively backup any folder across different devices. Symform users are required to install the client application to setup and enable file synchronisation. The user interface for the Linux and Mac (in alternative) client application is a web-based Graphical User Interface (GUI) known as the Remote Device Manager (RDM), which is accessible through 'localhost:59234' by default. The default Symform backup folders created by the OS are Music, Pictures, Desktop, and Documents folders; which can be modified by the users. Notice that Symform does not provide the ability for file sharing.

## 1.2. Related work

75 Since the early 2010's, a number of scholars have highlighted operational and legal challenges and various research opportunities associated with cloud forensic investigations [34, 35, 36, 37, 38, 39, 13, 10]. Recently, some researchers published a number of technical solutions to mitigate the identified challenges [40, 41, 42, 43, 23], particularly those associated with the remote collection of data artefacts from a decentralised cloud infrastructure. 80 Moreover, some other researchers also explored the potential of collecting evidence from client devices [11, 21, 22, 24, 20, 44, 45, 25, 26, 27, 28, 17] and servers [46, 47, 17]. Other research efforts include:

- 85 • Evaluation of the effectiveness of commercial forensic tools (e.g., Guidance EnCase, the Forensics Tool Kit (FTK), Memoryze, and AWS Export) in acquiring evidence remotely from the Amazon EC2 servers [48].
- Determining whether the integrity (e.g., any change in MD5, SHA1, and timestamps) of the synced files acquired from popular cloud storage providers, such as Dropbox, Google Drive, Microsoft SkyDrive [14] and 90 iCloud [45], are affected in the forensic collection.
- Proposal of frameworks, guidelines and methodologies with the aims of providing a systematic approach for forensic collection of cloud artefacts from servers and/or client devices. Martini and Choo [46] were the first to propose cloud forensic framework, which was used to investigate ownCloud [47], Amazon EC2 [44], VMWare [23], and XtremFS [49]. 95 Subsequently, Quick and Choo [21, 24] and Quick et al. [17] extended the four-stage framework and validated using SkyDrive, Dropbox, Google Drive and ownCloud [17]. Chung et al. [11] proposed a cloud investigation guideline and utilised it to investigate Amazon S3, Google Docs, and Evernote on Windows, Mac OS, iOS, and Android devices. 100 Farina et al. [25] investigated the artefacts left by Bit Torrent Sync and outlined an investigative framework for the remote collection of evidence from a decentralised file synchronisation network [50]. Scanlon et al. [51] further extended the work of Farina et al. [25] and designed a methodology for the network investigation of Bit Torrent Sync. 105

Due to the recency of cooperative storage cloud services, this is the first forensic research undertaken to identify artefacts of forensic interest that may remain after the use of such services on the client's device.

### 1.3. Contribution

110 Similar to the approaches of Quick and Choo [21, 22, 24], we attempt to answer the following questions in this research:

1. Does the act of file download or file upload using Symform cooperative storage cloud alter the file contents and timestamps of the original files?
- 115 2. What artefacts can be found on a computer hard drive and memory after a user has used the Symform client application and web application? where are their locations on Windows 8.1, Ubuntu 14.04 LTS, and Mac OS X Mavericks 10.9.5?
- 120 3. What data remains on an Apple iPhone 4 and an HTC One X after a user has used the Symform client apps? where are their locations on iOS Version 7.1.2, and Android KitKat 4.4?
4. What data can be seen in network traffic?

Findings from this research will contribute to the forensic community's understanding of the types of terrestrial artefacts that are likely to remain after the use of cooperative storage cloud on devices (e.g., personal computers and mobile devices) running different operating systems.

### 1.4. Outline

The structure of this paper is as follows. Section 2 outlines the research methodology. Section 3 details the evidence collection phase, which will answer the first research question. Sections 4 and 5 discuss the findings from the technical experiments involving the personal computers and mobile devices. These sections will answer the second and third research questions. Section 6 explains the network artefacts, which will answer the final research question of this research study. Finally, in Section 7, we conclude the paper and outline potential future research areas.

## 135 2. Research Methodology

This section provides an overview of the cloud investigation framework used to guide the investigations in this paper as well as the experimental setup.

## 2.1. Cloud investigation framework

140 It is essential that (digital) forensic investigators or practitioners adhere to generally accepted forensic principles, standards, guidelines, procedures and best practices when undertaking digital forensic investigations [52, 53]. In particular, Kent et al. [54] (p.5) define the forensic process as follows:

145 *“An individual performing forensic activities needs to understand forensic principles and practices, and follow the correct procedures for each activity, regardless of which group he or she is a member.”*

As an example, Mckemmish [55] explained that digital forensic investigations should be based on four principles, namely minimal of the original, account for any changes, comply with the rules of evidence, and not to exceed knowledge. Moreover, the National Institute of Standards and Technology (NIST) prescribed that a digital forensics framework should contain the necessary components, namely collection, examination, analysis, and reporting [54].

150 In this research, we adopt the cloud investigative framework proposed by Martini and Choo [46] as shown in Figure 1. While the framework shares several similarities with the frameworks of McKemmish [55] and NIST [54], it differs in a number of ways. The primary difference being that of the third phase, which emphasises one or more simultaneous iteration(s) of the framework with evidence source identification and preservation via the associated devices. These comprise remote servers, peer nodes (in P2P storage cloud investigation), and other connected devices typically identifiable from a client device.

165 In the following, we briefly explain each of the four investigation phases in the context of our research.

1. *Evidence source identification and preservation.* In the first phase, we identify the physical hardware of interest, which contained the virtual disk files (VMDK) and virtual memory files (VMEM) in each VM folder. The mobile devices used in this research were an HTC One X running Android KitKat 4.4 and an Apple iPhone 4 running iOS Version 7.1.2. We created and verified a forensic copy of each VMDK and VMEM file in E01 container and raw image file (dd) formats respectively. For the mobile devices, we acquired a bit-for-bit image of the internal storage and we converted it to the E01 container format. Then, we calculated

an MD5 hash value for each original file and subsequently we verified each copy.

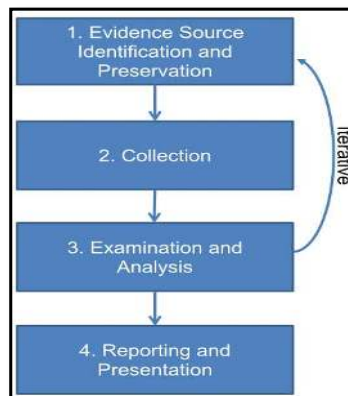


Figure 1: Cloud forensics framework of Martini and Choo [46].

- 180 2. *Collection*. In this phase, we collected files containing the details needed for analysis and keyword searching in the forensic copies. Similar to the earlier evidence source identification and preservation phase, we calculated the MD5 and SHA1 hash values of each original file and subsequently verified each collected or exported file. Further details of this phase are explained in Section 3.
- 185 3. *Analysis*. This phase is concerned with the examination and analysis of data at rest, in motion, or in execution. We undertook data parsing, carving, and keyword search for Symform artefacts located in the forensic copies of volatile and non-volatile data captures. We considered both indexed and non-indexed, as well as Unicode and non-Unicode string searches (in Hex editor) as part of our keyword search. The following
- 190 search terms were identified after examining the file names observed and the text from within the Enron data files:
- symform.com, symform.
  - 3111.zip, 3111.pdf, 3111.docx, 3111.jpg, 3111.rft, 3111.txt, 13100.zip, 13100.pdf, 13100.docx, 13100.jpg, 13100.rft, 13100.txt, Enron, Pensive Parakeet.
- 195



- the username and password created for this research.

4. *Reporting and presentation.* This phase involves legal presentation of the collected evidential data in a court of law.

200 *2.2. Experimental setup*

For our analysis, we created a total of 33 VMs each one representing different physical systems to simulate a series of real life scenarios of using Symform (e.g., install, access, upload, download, view, delete, and uninstall) on various operating systems, as detailed in tables 1, 2, 3, and 4, and figures 2, 3, and 4. We used the base VMs as a control media to determine changes during each experiment. As explained by Quick and Choo [21, 22, 24], using physical hardware to undertake setup, erasing, copying, and re-installing would have been an onerous exercise. Moreover, a virtual machine allows room for error by enabling the test environment to be reverted to a restore point if the results are unfavorable. We configured the hard drive and RAM with minimal space in order to reduce the time required to analyse the considerable amounts of snapshots. Similar to the approaches of Quick and Choo [21, 22, 14, 24], we used the 3111<sup>th</sup> and 13100<sup>th</sup> email messages of the UC Berkeley Enron email dataset (downloaded from [http://bailando.sims.berkeley.edu/enron\\_email.html](http://bailando.sims.berkeley.edu/enron_email.html) on 24<sup>th</sup> of September 2014) to create the sample files and saved in .RTF, .TXT, .DOCX, .JPG (print screen), .ZIP, and .PDF formats.

Immediately upon the completion of each experiment, we took a snapshot of each of the VMs after being shutdown in order to allow restoring at a later stage, if necessary. We remark that we captured the RAM immediately after each experiment, before the shutdown. As noted by Quick and Choo [21, 22], .VMEM files represent a capture of memory dump which is not being with the use of memory acquisition tools. Therefore, instead of acquiring RAM dump using live acquisition tools, e.g., ‘win32dd’ and ‘FTK Imager’, we captured the memory on a copy of the VMEM files created by VMware. A similar consideration was made in relation to creating a forensic copy of the hard drive; we instantiated the physical hard drive by the copy of the VMDK files created by VMware. Meanwhile, we facilitated the network traffic by the .LIBPCAP files captured using ‘Wireshark’.

Table 1: Configurations of virtual machines for Symform web application analysis on Windows 8.1.

<b>VM</b> (Parent and child VMs)	<b>Details</b>
<p><b>1.0 Base-VM</b> 1.1 IE, 1.2 MF, 1.3 GC</p>	<p>We prepared a base VM running Windows 8.1 Professional (Service Pack 1, 64-bit, build 9600), and equipped with 2GB RAM and 20GB hard disk. We then installed three well-known browsers namely Microsoft Internet Explorer Version 11.0.9600.17351 (IE), Mozilla Firefox Version 33.1 (MF), and Google Chrome Version 37.0.2062.124m (GC) into three separate VMs duplicated from the base VM. We used a separate computer to create Symform account, configure a Windows device, and upload the sample files using the client application.</p>
<p><b>Access-VM</b> 1.1.1 IE, 1.2.1 MF, 1.3.1 GC</p>	<p>We created a copy of each web browser base VM (1.0) and used to analyse the process of logging in Symform web application using the respective browsers on Windows 8.1.</p>
<p><b>Download/ Open-VM</b> 1.1.2 IE, 1.2.2 MF, 1.3.2 GC</p>	<p>We created extra copies of the web browser base VMs (1.0) to examine the process of downloading and viewing files through each investigating web browser on Windows 8.1. We noted the creation, modification, and last accessed times of each file to detect changes in timestamps following file download.</p>
<p><b>Delete-VM</b> 1.1.3 IE, 1.2.3 MF, 1.3.3 GC</p>	<p>We created additional copies of the web browser base VM (1.0) to investigate the artefacts of deleting synced files on the Symform web application.</p>

Table 2: Configurations of virtual machines for Symform client application analysis on Windows 8.1.

<b>VM</b> (Parent and child VMs)	<b>Details</b>
<b>1.4 Install-VM</b>	Using a duplicate copy of the base VM (1.0), we accessed Symform website to download and subsequently install the Symform client application Version 4.24.0.0 (for Windows). ( <a href="http://www.symform.com/download/windows/">http://www.symform.com/download/windows/</a> )
<b>1.4.1 Access-VM</b>	We made a copy of the install VM (1.4) to examine the process of logging in Symform account using the client application on Windows 8.1.
<b>1.4.2 Upload-VM (Synchronise)</b>	We made a copy of the install VM (1.4). We copied the Enron dataset files from the host machine to C:\sync (self-defined backup folder) of this VM and subsequently uploaded to the Symform server.
<b>1.4.2.1 Uninstall-VM</b>	We created a copy of the upload VM (1.4.2) to examine the process of uninstalling the Symform client application on Windows 8.1. We performed uninstallation using the default Windows uninstallation feature (Control Panel\All Control Panel Items\Programs and Features).
<b>1.4.3 Download-VM (Synchronise)</b>	We duplicated the install VM (1.4) (without being tainted with the dataset files) to examine the process of downloading or synchronising files using the Symform client application on Windows 8.1. We then downloaded all the files which were uploaded from the Upload-VM (1.4.2) to the C:\sync folder of this VM. We noted the creation, modification, and last accessed times of each file to detect changes in timestamps after transferring files.
<b>1.4.3.1 Delete-VM (Synchronise)</b>	We created a copy of the download VM (1.4.3) to assess the process of deleting files downloaded or synchronised using the Symform client application on Windows 8.1 (without emptying the Recycle Bin). No anti-forensic technique was applied to simulate a typical file-deleting situation.

Table 3: Configurations of virtual machines for Symform client application analysis on Ubuntu 14.04 LTS.

VM (Parent and child VMs)	Details
<b>2.0 Base-VM</b>	We prepared a base VM running Ubuntu 14.04 LTS, and equipped with 1GB RAM and 20GB hard disk.
<b>2.1 Install-VM</b>	We made a copy of the base VM (2.0) and used to access Symform website ( <a href="http://www.symform.com/download/linux/">http://www.symform.com/download/linux/</a> ) and download and subsequently install the Symform client application Version 4.24.0.0 (for Linux).
<b>2.2 Access-VM</b>	We created a copy of the install VM (2.1) to examine the process of logging in Symform account using the Symform client application. Since Symform (Linux) client application is a web GUI, we signed-in using the default Mozilla Firefox browser (Version 31.0) by accessing <a href="http://127.0.0.1:59234/">http://127.0.0.1:59234/</a> .
<b>2.3 Upload-VM (Synchronise)</b>	We used a duplicate copy of the install VM (2.1) to undertake the file upload process on Ubuntu 14.04 LTS. We copied the Enron dataset files from the host machine to <code>/root/sync/</code> (self-defined backup folder) of this VM and subsequently uploaded to the Symform server.
<b>2.3.1 Uninstall-VM</b>	Using a duplicate copy of the upload VM (2.3), we uninstalled the Symform client application. We used the uninstallation commands (e.g., <code>sudo apt-get remove symform</code> and <code>sudo apt-get purge symform</code> ).
<b>2.4 Download-VM (Synchronise)</b>	We duplicated the install VM (2.1) (without being tainted with the dataset files) to undertake the process of downloading or synchronising files using the Symform client application. We then downloaded all the files uploaded from the Upload-VM (2.3) to <code>/root/sync/</code> of this VM. We noted the creation, modification, and last accessed times of each file to detect changes in timestamps following file download.
<b>2.4.1 Delete-VM (Synchronise)</b>	We created a copy of the download VM (2.4), without emptying the Trash. We did not apply any anti-forensic technique to simulate a typical file-deleting situation.

Table 4: Configurations of virtual machines for Symform client application analysis on Mac OS X Mavericks 10.9.5.

VM (Parent and child VMs)	Details
<b>3.0 Base-VM</b>	We prepared a base VM running Mac OS X Mavericks 10.9.5, and equipped it with 1GB RAM and 60GB hard disk.
<b>3.1 Install-VM</b>	Similar to Windows 8.1, we duplicated the base VM (3.0) and used it to access Symform website ( <a href="http://www.symform.com/download/mac/">http://www.symform.com/download/mac/</a> ) to download and subsequently install the Symform client application Version 4.24.0.0 (for Mac).
<b>3.2 Access-VM</b>	We created a copy of the install VM (3.1) to examine the process of logging in Symform account using the client application and the web GUI on Mac OS X Mavericks 10.9.5.
<b>3.3 Upload-VM (Synchronise)</b>	We created a duplicate copy of the install VM (3.1) and used it to undertake file upload through the Symform client application. We copied the Enron dataset files from the host machine to <code>/Users/[User Profile]/sync/</code> (self-defined backup folder) of this VM and subsequently uploaded it to the Symform server.
<b>3.3.1 Uninstall-VM</b>	Using a duplicate copy of the upload VM (3.3), we uninstalled the Symform client application by the uninstallation commands for Mac OS (e.g., <code>sudo/Library/Application\Support/Symform/scripts/uninstall</code> and <code>sudo/Library/Application\Support/Symform/scripts/uninstall-purge</code> ).
<b>3.4 Download-VM (Synchronise)</b>	We duplicated the install VM (3.1) (without being tainted with the dataset files) to examine the process of downloading or synchronising files. In this VM, we downloaded all the uploaded files from the upload VM (3.3) to <code>/Users/[User Profile]/sync/</code> of this VM. We noted the creation, modification, and last accessed times of each file to detect changes in the timestamps after undertaking file download.
<b>3.4.1 Delete-VM (Synchronise)</b>	We created a copy of the download VM (3.4) to assess the process of deleting the files downloaded or synchronised (without emptying the Trash). We did not apply any anti-forensic technique to simulate a typical file-deleting situation.

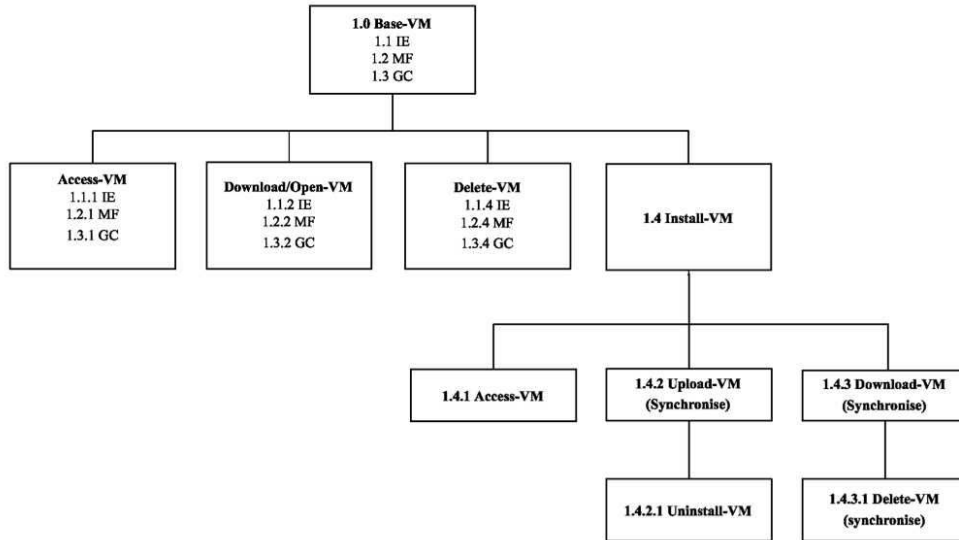


Figure 2: VMs created for Symform investigation on Windows 8.1.

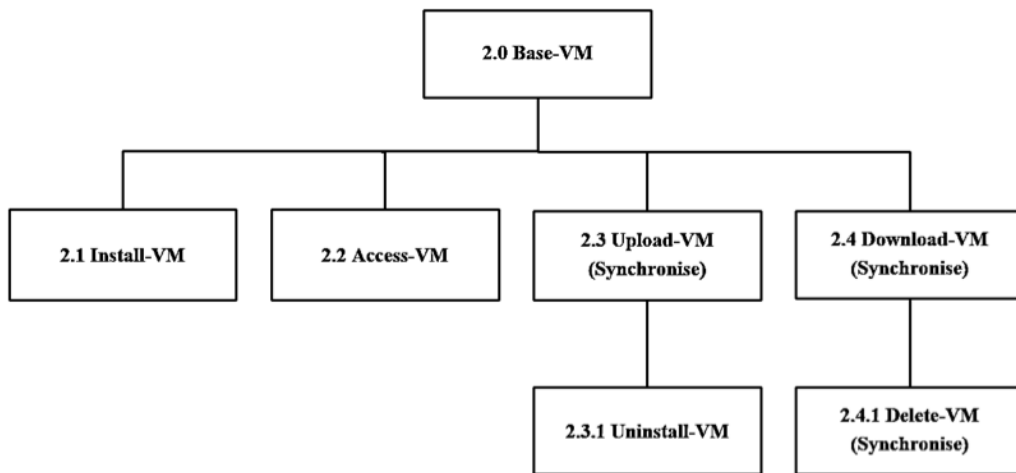


Figure 3: VMs created for Symform investigation on Ubuntu 14.04 LTS.

230 In order to analyse the Symform mobile app, we prepared a default factory restored iPhone 4 running iOS 7.1.2 and an HTC One X running Android KitKat 4.4 to simulate the use of the Symform app on both devices. To gain root access to the devices, we jailbroke the iOS device using ‘Pangu8 Version 1.1’ and rooted the Android device using ‘Odin3 Version 185’. To

235 examine the manner in which the file systems were treated with respect to  
the installation and viewing of files (the only feature supported by the mobile  
apps) and the uninstallation of the Symform apps, we created a binary image  
of the mobile devices for different Symform usage scenarios using ‘dd’ over  
SSH/ADB Shell. In particular, we took the first image prior to the installation  
240 of the Symform apps to ensure that neither Symform nor Enron-related data  
were on the devices. Then we installed the Symform iOS app Version 1.13  
and Android app Version 1.3 on the respective devices and took the second  
image of each device. Moreover, we took another image of the devices after  
viewing the dataset files in the Symform apps. Then we made the final image  
245 following the uninstallation of the apps.

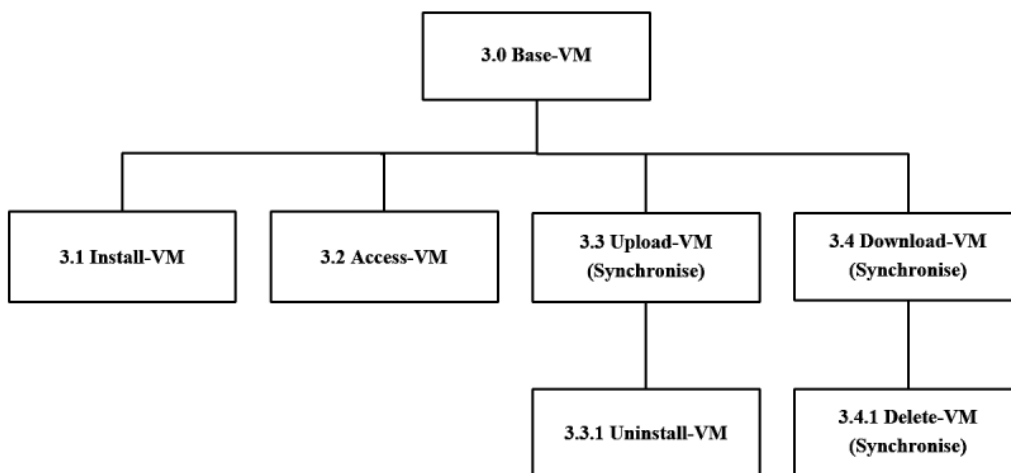


Figure 4: VMs created for Symform investigation on Mac OS X Mavericks 10.9.5.

‘Wireshark’ was hosted on the host machine to capture the network traffic  
of the suspect’s VM; started prior each experiment was carried out, and we  
saved a copy of the network capture after carrying out each experiment. To  
ensure consistency of the findings, we conducted each experiment at least  
250 thrice on different dates. Table 5 details the tools prepared to undertake this  
research.

Table 5: Tools prepared.

Tool	Usage
<b>FTK Imager v3.2.0.0</b> <b>dd v1.3.4-1</b>	To create forensic images for .VMDK and .VMEM files. To produce the bit-for-bit images of the internal storage of the mobile devices.
<b>emf_decrypter.py</b> <b>Autopsy 3.1.1</b>	To decrypt the iOS images acquired for analysis. To parse the file system, produce directory listings, and extracting or analysing images of the mobile devices, stored files, browsing history, swap files, unallocated partitions, as well as Windows system files, e.g., NTUSER.dat registry files (using the RegRipper plugin) and pagefile.sys Windows swap files, located within the forensics images of .VMDK files.
<b>HxD v1.7.7.0</b> <b>Volatility 2.4</b>	To conduct keyword searches in .VMEM files. To analyse running processes (using the plist function), network information (using the netscan function), and detecting the location of a string (using the yarascan function) in physical memory dumps.
<b>SQLite Browser v3.4.0</b> <b>Wireshark v1.10.1</b> <b>Network Miner v1.6.1</b> <b>Whois command</b>	To view the contents of SQLite database files. To analyse network files. To analyse and carve network files. To determine the registration information of an IP address.
<b>Photorec 7.0</b> <b>Nirsoft Web Browser</b> <b>Passview v1.58</b> <b>Nirsoft cache viewer,</b> <b>ChromeCacheView 1.56,</b> <b>MozillaCacheView 1.62,</b> <b>IECacheView 1.53</b>	To undertake data carving of memory files. To recover the credential details stored within web browsers. To parse and analyse web browsers' cache files.
<b>BrowsingHistoryView</b> <b>v1.60</b>	To parse and analyse web browsers' history files.
<b>Thumbcacheviewer</b> <b>v1.0.2.7</b>	To view the thumbnail images stored in Windows thumbcache folder.
<b>Windows Event Viewer</b> <b>v1.0</b>	To view Windows event files.
<b>Console v10.10 (543)</b>	To view Mac-OS-specific log files (e.g., Apple System Logs).
<b>Windows File Analyser</b> <b>2.6.0.0</b>	To analyse Windows prefetch and link files.
<b>Plist Explorer v1.0</b> <b>chainbreaker.py</b>	To examine the contents of Apple Plist files. To recover the master keys stored in Mac's Keychain dump.
<b>NTFS Log Tracker</b>	To parse and analyse the \$LogFile, \$MFT, and \$UsnJrnl New Technology File System (NTFS) files.



### 3. Collection and timestamp analysis

Before undertaking analysis into Symform, we collected data that would potentially contain information of interest for cloud forensics (e.g., sync and file management metadata, caches, cloud service and authentication data, encryption metadata, browser artefacts, mobile artefacts, as well as network artefacts) as identified by Martini and Choo [46]. These included files stored within unallocated partitions, swap files, log files (e.g., system logs stored under %Windows% \system32\config on Windows machine, /var/log on Ubuntu machine, /private/var/log on Mac OS machine, and Symform log files), thumbnail cache files, preference files (e.g., property list –Plist– files stored within /Users/[User profile]/Library/Preferences/ of Mac OS), Symform sync folders, web browser files, and Windows system files (e.g., \$LogFile, \$MFT, \$UsnJrnl, NTUSER.dat, prefetch files, thumbnail cache files, link files, as well as other installation and user-specific preference files saved under %ProgramFiles% \AppData% \). The network traffic and RAM captures were facilitated by the .LIBPCAP and .VMEM (preserved in AD1 format) files captured in our research, respectively.

During our collection of the downloaded/synced files on Windows 8.1, we observed that the last accessed and last modified times were the times when the files were downloaded. However, the last written times retained its original value unchanged. On Ubuntu, we observed that the modification, creation, and last opened times remained unchanged. On the other hand, the added times were the times when the files were downloaded to the machine. An inspection of the timestamps of the downloaded files on Mac OS determined that the modified times remained unchanged while the accessed times matched the download times. Of all the OS investigated, the MD5 and SHA1 hash values for the downloaded files remained the same as the original values in all experiments, which suggested that no alteration was made during transmission of the files.

### 4. Symform analysis on computer devices

In this section, we present the findings of our analysis on Windows 8.1, Ubuntu 14.04 LTS, and Mac OS X Mavericks 10.9.5. Symform artefacts collected included data remnants within the directory listings, log files, browser files, thumbnail cache, RAM, swap file/partition, unallocated space, as well as Windows system files (e.g., registry files, pagefile.sys, link files, and prefetch files).

#### 4.1. Directory listings and files of forensic interest

The directory information is essential when seeking to determine if an application has been used to initiate further investigation. An examination of the control base VMs (1.0 Windows Base-VM, 1.1 IE, 1.2 MF, 1.3 GC, 2.0 Ubuntu Base-VM, and 3.0 Mac Base-VM) confirmed that there was no data related to Symform and the Enron emails on these VMs. Therefore, any Symform-related data recovered from the remaining VMs would indicate Symform usage.

##### 4.1.1. Windows 8.1

The main starting point of Symform investigation on Windows machine is in `%Program Files% \Symform\`. The folder of particular interest is the `%Program Files% \Symform\Node Service\logs` folder, which maintains a list of Symform log files useful to assist forensic practitioners in scoping the investigation (see Section 4.2). Meanwhile, the `%Program File% \Symform\Node Service\node.config` XML file stores a wealth of node-specific configuration details for the file synchronisation and contribution services, which include the server address for the cloud management service (prefixed with `'serverAddress'`), unique SHA-1 node ID (prefixed with `'nodeId'` in the `node` tag), encrypted secret key (prefixed with `'secretKey='` in the `node` tag) in base64 format, login username or email address (prefixed with `'username='` in the `userCredentials` tag), encrypted password (prefixed with `'password='` in the `userCredentials` tag) in base64 format, location of the contribution folder (prefixed with `'fragmentStorePath='` in the `contribution` tag), port number of the contribution service (prefixed with `'port='` in the `contribution` tag), as well as bandwidth limits (prefixed with `'uploadBandwidthCapacity='` and `'downloadBandwidthCapacity='` in the `node` tag respectively).

Moreover, within the `folderMapping` tag we found an opening and closing `folder` sub-tag for the sync folder, which contain information such as the folder global ID of a sync folder (prefixed with `'remoteFolderGlobalId'`), the folder path (prefixed with `'localPath='`), the remote folder name (prefixed with `'remoteFolderName='`), and the folder editing permission (prefixed with `'direction='`) see Figure 5. Such information would be useful in identifying a user's self-defined sync folder(s). The node and FolderGlobal IDs could also assist investigators in correlating any external data obtained from an Internet service provider (ISP) or other external content or service provider.

```

<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <configSections>
    <section name="node" type="Symform.Node.NodeSettings, Symform.Node" allowLocation="true" allowDefinition="Everywhere" allowExeDefinition="MachineToApplication"
    overrideModeDefault="Allow" restartOnExternalChanges="true" requirePermission="true" />
  </configSections>
  <node serverAddress="https://control.symform.com" version="1.2" nodeId="
  secretKey="AQAAANCmnd8BFdERjHoAwE/Ci+8AAAAAXMPeb0hb0WFR8amqhgZgQAAAAACAAAAAAQZgAAAAEAACAAAAAbFgCkKJNEdAkQK6O2bN/CpdYHl/5a
  IUZ5ZjWX7EgAAAAAAGAAAAAIAACAAAAAzWfFekXMIObVenBarnQJ9pzz5E.XlgRzX6fqWkyHGAAADWPbO8REZwa+qXbGz6vbgNXObOkI0Q92TVOIvaPnw8zU
  PUiy+6cbQFQee8SGN6eH8o74Yg9fuN68R+QnnQHA1nvdIZGz3MyQW0AGB15RE7wimvOVBlasAmJZlpa2xUFAAAAaQZPZNe0E5F6XQo70Hh19Ue7+zxQUUkReLiBmEjG
  cz0Zlfs/Y6dRA9rg06Mp/5G/WyAw70IDQVR9dJ7g=="" stunHostName="" uploadBandwidthCapacity="5000" downloadBandwidthCapacity="5000" processorUsageLimit="4"
  bufferSizeMultiplier="4" userNameCache="" userIdCache="" suppressUp="" false" minimumUploadRate="16000" minimumUploadWriteRate="8000" showDesktopNotifications="true">
    <sync businessHoursStartTime="01/01/0001 08:00:00" businessHoursEndTime="01/01/0001 18:00:00" enabled="true" businessDays="Monday, Tuesday, Wednesday, Thursday, Friday"
    businessHoursBandwidth="0" businessHoursDownloadBandwidth="0" offHoursBandwidth="0" offHoursDownloadBandwidth="0" makeTemporaryCopy="false" temporaryCopyPath=""
    fragmentTransferThreshold="100" startingUploadParallelConnections="8" startingDownloadParallelConnections="20" globalCachePath="" globalCacheQuota="0" dataSizeTotalSize="0"
    dataSizeRateOfChange="0" watchFileSystemChanges="true">
      <folderMapping>
        <folder remoteFolderGlobalId="remoteFolderId="00000000000000000000000000000000" localPath="C:\sync" remoteFolderName="sync"
        direction="DownloadAndUpload" +++++ order="0" cache="Private" cacheQuota="0" vssEnabled="false" />
      </folderMapping>
      <sync>
        <contribution enabled="false" fragmentStorePath="C:\SymformContribution" port="59280" manualPort="false" localAddress="" registrationAddress=""
        enableBackgroundTasks="true" concurrentMoves="2" logicalVolumeSize="0" ignoreThrottle="false" enablePortForwarding="true" testFragmentConfirmAction="NotSpecified"
        enableUp="" true" />
      </sync>
      <userCredentials>
        <user userName="
        password="AQAAANCmnd8BFdERjHoAwE/Ci+8AAAAAXMPeb0hb0WFR8amqhgZgQAAAAACAAAAAAQZgAAAAEAACAAAAAYmbHDakUdp/uF+AFMOWZ7+gYzk
        WNIgY71R/Wmg8QAAAAAGAAAAAIAACAAAAQub5gKpwGGkbfJL4774eizRP30w4UL.RahnNd4zyAAAAACWmSet+QJzeicjX7G0zrw5bZnd0FTDwuM6kfsBq0AAA
        AACFCaPwLuvB4fJazeQ4ptgFRQIV5vXPJmQepLap0i3BXwKZC2JMoB+BlksB/RwV3eDKLX87UvYesPwx" />
      </user>
    </node>
  </configuration>

```

Figure 5: Content of node.config File (valuable remnants are bolded).

In our experiments, we determined that when a sync folder was configured, two hidden subfolders (.symform and .symform-store) would be created in the sync folder to store the synced file caches. The file of particular interest is the *metadata* sqlite3 database in the .symform subfolder, which maintains a list of synced file metadata such as synced filenames as well as their sizes, last modified times, and checksums in the *FolderItem* Table. The metadata remained in the database even after the synced files had been deleted, and the deleted files could be discerned from entries with empty *Size* and *Checksum* table fields. Figure 6 shows an example of the *FolderItem* Table of metadata database containing entries for deleted files namely *Enron3111.zip*, *Enron3111.txt*, *Enron3111.rtf*, and *Enron3111.pdf*. The findings also suggest that when a user uses Symform, there will be references to the creation or removal of the sync folder remaining in the NTFS files (e.g., \$LogFile, \$MFT, and \$UsnJrnl) that can be used to identify its usage or synced files; an example of the \$LogFile entries is shown in Figure 7.

Located within the %SymformContribution% folder were subfolders storing fragments of backup files from peer nodes. The file fragments were represented by the unique IDs with the following naming convention: [Unique SHA-1 for a file fragment].[File fragment number].[Folder global ID] (e.g., 0A1D16AF1E451120F32399828FEAEE3A8797XXXX.067.0000000220035142XX XX). However, the shredded and encrypted file fragments mean the actual files are not recoverable without the assistance from cloud service provider. In ad-



the `.symform` cache folder. These artefacts could be useful for ascertaining the sync folder and recovery of the synced file history after uninstallation of the client application, since a typical user would not manually delete the cache folders (hidden by default). In addition, the uninstallation could be  
370 ascertained from the presence of entries referencing removal of the client application filenames in the `$LogFile`, `$MFT`, and `$UsnJrnl` NTFS files.

#### 4.1.2. Ubuntu 14.04 LTS

An examination of the directory listing of the Install-VM (2.1) revealed that the main installation directory of the Linux Symform applica-  
375 tion could be located in `/opt/symform/` by default. Of particular interest is the `/opt/symform/bin/logs` subfolder which stores various Symform logs (see Section 4.2.4). In addition, we located copies of the log files in `/var/log/symform/`. A list of certificates used by Symform stored within the  
380 `/var/lib/symform/.mono/certs/Trust` folder and the user keypair metadata (e.g., the base64-encoded values of the P, Q, DP DQ, InverseQ, and D keys of the RSA algorithm used by Symform) could be located in the `/var/lib/symform/.mono/keypairs` folder.

The Symform contribution folder and `node.config` file (see Section 4.1) could be located in `/SymformContribution/` and `/var/lib/symform/` re-  
385 spectively. Similar to the Symform client application for Windows, we identified that when a sync folder is set up on Ubuntu OS, two hidden cache folders (`.symform` and `.symform-store`) will be created to store the sync file caches. The `/home/[User Profile]/.local/share/Trash/files`  
390 trash folder, when not emptied, would enable the recovery of deleted synced files. Additionally, the original file path and deletion time of each deleted file could be recovered from the `.TRASHINFO` file located in `/home/[User Profile]/.local/share/Trash/info/` – see Figure 8. When the uninstallation occurred, we observed that the `/opt/symform`, `/var/log/symform`,  
395 and `/var/lib/symform` folders were emptied. However, forensic practitioners could potentially recover the synced file history from the *metadata* database remained in the `.symform` cache folder (of each sync folder).

#### 4.1.3. Mac OS X Mavericks 10.9.5

The main installation directory of the Symform client application  
400 for Mac OS is in `/Library/Application Support/Symform/`. The `/Library/Application Support/Symform/bin/logs/` directory maintains a list of Symform logs. This would enable forensic practitioners to obtain

relevant Symform usage information (see Section 4.2.4). Alternatively, a copy of the *logs* folder could be located in `/private/var/logs/Symform/`.  
 405 The `node.config` file, *Trust* folder (which contains a list of certificates used by Symform), and *keypairs* folder (that contains the user key-pair information as outlined in Section 4.1.2) could also be located in `/private/var/lib/symform/`.

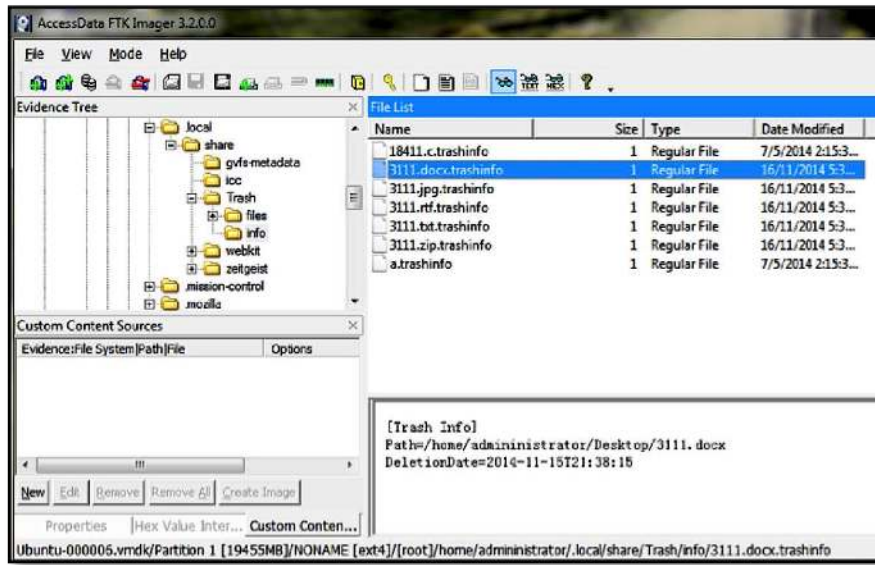


Figure 8: Trash info for deleted files in Linux.

When the file synchronisation occurred, we were able to recover  
 410 copies of the `node.config` file from the `receiver_data` table field of the `cfurl_cache_receiver_data` table of `Cache.db` database located in `/Users/[User Profile]/Library/Caches/com.symform.mac.Symform`.  
 Also located in the same table field were additional details relating to the file synchronisation sessions such as number of syncing folders, number  
 415 and sizes of syncing files, file transfer speeds, remote folder names, local folder names, the estimated sync durations, and the service start times.  
 In addition, the `cfurl_cache_response` table of `Cache.db` maintains a list of Symform URLs as well as the corresponding timestamps (see Figure 9). Forensic practitioners could potentially discern the start and stop  
 420 timestamps for the sync service from the timestamps of the `request_key` entries referencing `http://localhost:[Portnumber]/syncService/start` and `http://localhost:[Portnumber]/syncService/stop` respectively.

Alternatively, references to the download links for the installer and synced files could be potentially recovered from the /Users/[User Profile]/Library/Preference/com.apple.LaunchServices.Quarantine EventsV2 sqlite3 database.

entry_id	version	hash_value	storage_path	request_key	time_stamp	partition
Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	0	48826444361...	0	http://www.google-analytics.com/collect?utm=1480781:pageview&uid=10610272208502&dt=application%2Fminilaunching%2F18.dh:osapp.symform.com&dp=/application%2Fminilau...	2014-11-18 03:15:51
2	2	0	207308879	0	http://calibot520411.sync.Service	2014-11-18 03:15:51
3	3	0	42625252704...	0	http://www.google-analytics.com/collect?utm=21247524941:pageview&uid=110610272208502&dt=setup-welcome&id=1:18.dh:osapp.symform.com&dp=/setup-welcome&id=UA-10083620-1	2014-11-18 03:15:51
4	4	0	156554922704...	0	http://www.google-analytics.com/collect?utm=1422650077&id:pageview&uid=10610272208502&dt=setup-account-login:create&id=1:18.dh:osapp.symform.com&dp=/setup-account/login...	2014-11-18 03:16:21
5	5	0	1115274822704...	0	http://www.google-analytics.com/collect?utm=1115418185&id:pageview&uid=10610272208502&dt=setup-account-login:create&id=1:18.dh:osapp.symform.com&dp=/setup-account/login...	2014-11-18 03:16:40
6	6	0	388419248520...	0	http://www.google-analytics.com/collect?utm=1784481492&id:pageview&uid=10610272208502&dt=setup-folder-remote-step:1:18.dh:osapp.symform.com&dp=/setup-folder-remote-st...	2014-11-18 03:16:48
7	7	0	79161305202...	0	http://www.google-analytics.com/collect?utm=1424304284:pageview&uid=10610272208502&dt=setup-folder-remote-step:2:1:18.dh:osapp.symform.com&dp=/setup-folder-remote-st...	2014-11-18 03:17:03
8	8	0	53524182084...	0	http://www.google-analytics.com/collect?utm=114827467&id:pageview&uid=110610272208502&dt=setup-done&id=1:18.dh:osapp.symform.com&dp=/setup-done&id=UA-10083620-1	2014-11-18 03:17:18
9	9	0	587561151825...	0	http://calibot520411.sync.Service/stop	2014-11-18 03:17:35
10	10	0	126264722704...	0	http://calibot520411.sync.Service/start	2014-11-18 03:17:41
11	11	0	402214880395...	0	http://calibot520411.sync.Service	2014-11-18 03:17:41
12	12	0	1337112200	0	http://calibot520411.sync.Service	2014-11-18 03:17:51
13	13	0	17619722086...	0	http://www.google-analytics.com/collect?utm=352504807&id:pageview&uid=10610272208502&dt=menuitem-panel&id=1:18.dh:osapp.symform.com&dp=/menuitem-panel&id=UA-10083620-1...	2014-11-18 03:17:51

Figure 9: The cfurl\_cache\_response Table of Cache.db.

Similar to the Windows and Linux investigations, setting up of a Symform sync folder in Mac OS observed that two hidden cache folders (.symform and .symform-store) will be present. The deleted files were located in the non-emptied /Users/[User profile]/.Trash folder. Although the uninstal-

430 and .symform-store) will be present. The deleted files were located in the non-emptied /Users/[User profile]/.Trash folder. Although the uninstal-

435 (of each sync folder).

#### 4.2. Log files

Logs play a vital role in an incident investigation [57]. The log analysis in this research included searching for the term *Symform* and the *Enron* dataset filenames as well as going through the entries identifying events relevant to Symform.

440 to Symform.

##### 4.2.1. Windows 8.1

Similar to any other Windows application, when the Symform client application was installed or uninstalled on Windows 8.1, we located event entries referencing the installation or uninstallation of the services as well as the associated timestamps in Windows event files such as *Application.evtx* and *System.evtx* (see Figure 10) saved in %Windows% \system32\config\.

445 and *System.evtx* (see Figure 10) saved in %Windows% \system32\config\.

When file synchronisation took place via the client application, there were

entries referencing *symformsync* in the aforementioned Windows event files (see Figure 11).

450

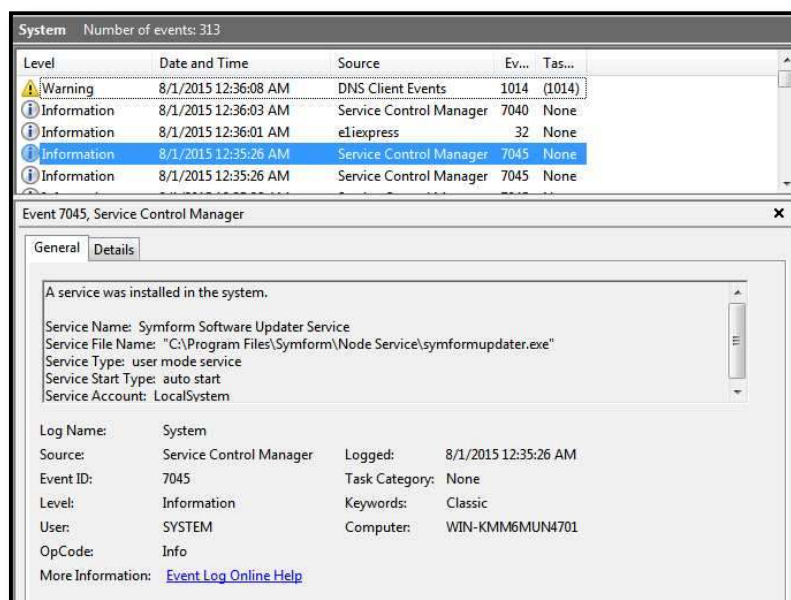


Figure 10: Windows event log entry for Symform installation.

#### 4.2.2. Ubuntu 14.04 LTS

Log files are particularly important in Linux investigations as almost all processes, events, and user account activities are logged [58]. When the Symform installation occurred on a Linux machine, forensic practitioners could potentially recover events relating to the creation of Symform services from the `/var/log/syslog`. Further information about the installation could be located in the `/var/log/dpkg.log`, since the log file maintains a list of events relating to the `Symform.deb` installer package such as the Symform version installed, installation status (e.g., unpacked, configured, and installed) as well as the corresponding installation time. Moreover, the `/var/log/auth.log` would provide information associated with the authorisation (granting *superuser* permission) details for Symform installation sessions, the paths accessed during the installation, and the corresponding installation time. The term *Symform* could be used in future searches.

455

460



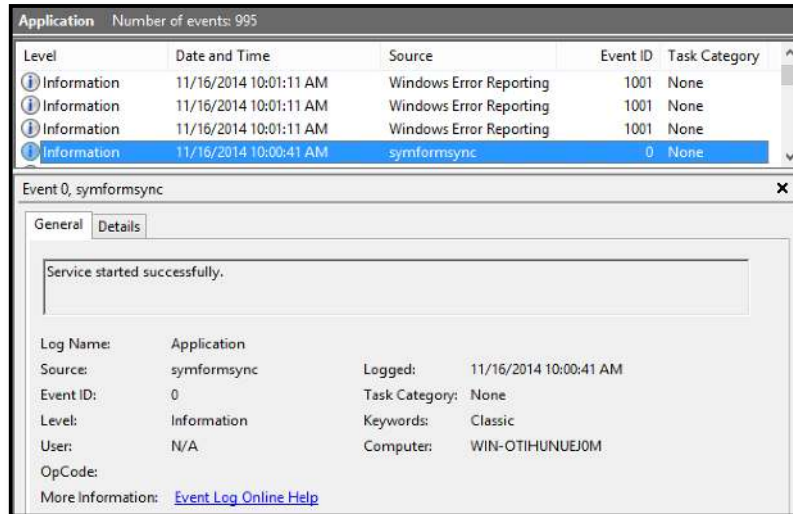


Figure 11: Windows event log entry for Symform sync service.

465 *4.2.3. Mac OS X Mavericks 10.9.5*

The installation or use of the Symform client application on a Mac machine could be discerned from the presence of Kernel and system events (e.g., program errors, installation data, privilege escalation) for the Symform main services in the `/private/var/log/system.log` log file, which contains information of interest such as the Symform application name, PID, and path references for the main services. Similar information could be found in the Apple System Logs (ASL) located at `/private/var/log/asl/YYYY.MM.DD.[User identification].[Group identification].asl` (where user identification (UID) and group identification (GID) are optional). Further information of the installation could be located in `/private/var/log/install.log`, which contains entries relating to the installation of the client application such as filename of the installer, the Symform version installed, as well as the time of installation. The term *Symform* could be used in future searches.

*4.2.4. Symform logs*

480 We determined that Symform logs were stored in `%Program Files% \Symform\Node Service\logs\` on Windows machine, `/var/log/symform/` on Ubuntu machine, and `/var/log/symform/` on Mac OS machine unencrypted. We also determined that by default, the logs would be archived hourly in compressed `.GZ` format and kept for a

485 month before being deleted, with the exception of the setup logs (e.g.,  
symformsetup.log, symformupdater.log, and loguploader.log). As the  
filename suggests, the symformsetup.log maintains a list of the client  
application setup information such as the version of the Symform client  
application installed, email addresses used to login the client application,  
490 login URLs, the full path(s) of the sync folder(s), as well as the installation  
time, which could be a potential starting point for Symform investigation.

The symformsync.log, symformsync-mono.log, symformcontrib.log,  
and symformcontrib-mono.log store a wealth of events relating to the Sym-  
form client application usage, comprising references to the node.config file  
495 path, copies of the node.config file, types of cryptographic algorithms used  
(e.g., MD5CryptoServiceProvider, SHA1CryptoServiceProvider, SHA256Cng,  
and AesCryptoServiceProvider), as well as the network details (e.g., download  
and upload speeds and its bandwidth limits). In addition, the filename and  
path references for the downloaded or uploaded files, request IDs for the  
500 backup file fragments, file synchronisation statuses, port numbers used by  
the file synchronisation services, as well as the times when the file synchro-  
nisation services were started and terminated could be recovered from the  
symformsync.log and symformsync-mono.log (see Figure 12). The terms  
download, downloading, starting download session, upload, uploading, starting  
505 upload session, syncing, and preparing could be used to locate the synced file  
details. Meanwhile, the node configuration information could be discerned  
from log entries referencing the term SyncHost, while the external IP ad-  
resses used by the host could be located using the term LastSeenIPAddress.  
Within the symformcontrib.log and symformcontrib-mono.log there ex-  
510 exists additional information relating to the contribution service such as the  
full path of the contribution folder, the port number for the contribution  
service, the times when the contribution service was started or terminated, as  
well as the external IP addresses of the corresponding nodes. The timestamp  
information noted along the log entries could be used for timeline or super  
515 timeline analysis [59].

When we accessed Symform on Ubuntu, we located references to the di-  
rectories accessed and created by the Symform client application in the  
/var/lib/dpkg/info/symform.list log file, which is useful when seek-  
ing to determine the installation and sync folder paths. In addition, the  
520 symformcloudcache.log (found only in Mac OS) maintains a list of HTTP  
requests for retrieving or updating the account or client-device information  
from or to the server along with the associated timestamps (see Figure 13).

```
2015-01-07 16:35:31,232Z [1] INFO NodeSettings - Loading configuration from 'C:\Program Files\Symform\Node Service\node.config'.
...
2015-01-07 16:35:31,670Z [1] INFO HttpServer - [59245] Listening for TCPv4 connections on '0.0.0.0:59245'.
...
2015-01-07 16:40:00,617Z [8] INFO SyncHost - <node serverAddress="https://control.symform.com" version="1.2" nodeId="
secretKey="AQAAANCmnd8BFdERjHoAwE/CI+sBAAA6cNaPY6b/UCVfRcHfSGqQAAACAAAAAAQZgAAAAEAAACAAAAADShOuaTnW8oGm56bKWyQwFDu8GZ++sAq
cJHjITAAAAAAGAAAAIAACAAAAADaknD+S0cD9IvgfxcD5zJ87EdphV9ZpVc48pUKrto2AAAAA+sOGioUkvSQOqmSIZZfBH9g7duSeWApAdDxjLH+XSHfzbxw4SvEdZm/
RfH2QIR0mA3ab6xPIRimJC+tpisuAT1hc85X69omsvrr+HQXcwx9jox4pXm2cDDe8RAAAAAZeE5YnHdvCHE2xQ4M7OmfvE4G6n6RjgRdKrtzBE9u9r0kF9eTbHtYXEm6NKIdj/34
a6DnnpDa/di/awCfK5g==" stunHostName="" uploadBandwidthCapacity="5000" downloadBandwidthCapacity="5000" processorUsageLimit="4" bufferSizeMultiplier="4"
userNameCache="" userIdCache="" suppressUp="false" minimumUpReadRate="16000" minimumUpWriteRate="8000" showDesktopNotifications="true"> <sync
businessHourStartTime="01/01/0001 08:00:00" businessHourEndTime="01/01/0001 18:00:00" enabled="true" businessDays="Monday, Tuesday, Wednesday, Thursday, Friday"
businessHoursBandwidth="0" businessHoursDownloadBandwidth="0" offHoursBandwidth="0" offHoursDownloadBandwidth="0" makeTemporaryCopy="false" temporaryCopyPath=""
fragmentTransferThreshold="100" startingUploadParallelConnections="8" startingDownloadParallelConnections="20" globalCachePath="" globalCacheQuota="0" datasetTotalSize="0"
datasetRateOfChange="0" watchFileSystemChanges="true"> <folderMapping> <folder remoteFolderGlobalId="
remoteFolderId="00000000000000000000000000000000" localPath="C:\sync" remoteFolderName="sync" direction="DownloadAndUpload" order="0" cache="Private"
cacheQuota="0" vsEnabled="false" /> <folder remoteFolderGlobalId="2199894739371" remoteFolderId="00000000000000000000000000000000" localPath="C:\sync"
remoteFolderName="Desktop" direction="DownloadAndUpload" order="0" cache="Private" cacheQuota="0" vsEnabled="false" /> </folderMapping> </sync> <contribution enabled="false"
fragmentStorePath="C:\SymformContribution" port="24360" manualPort="false" localAddress="" registrationAddress="" enableBackgroundTasks="true" concurrentMoves="2"
logicalVolumeSize="0" ignoreThrottle="false" enablePortForwarding="true" testFragmentConfirmAction="NotSpecified" enableUp="true" /> <userCredentials
userName="
password="AQAAANCmnd8BFdERjHoAwE/CI+sBAAA6cNaPY6b/UCVfRcHfSGqQAAACAAAAAAQZgAAAAEAAACAAACSUvyMKhHbFT8yuX5bWmaTspSeU2/576LEPq
76gLEAAAAAAGAAAAIAACAAAAADvInZahpBnSntF+zV709jgyu9QIYh+HHZ/owUriGdSAAAAAlxcC2vkOwPGJwNyUBDYulb4qcnvFIONQiahgAcOeUAAAAACHnpxU0w
u1oRIHnJUCtTefbopioqPCe9E9uJ0c4QTvQRVvQ+nm0vzrnzdwegnfxU8xR2t6Ako1" /> </node>
...
2015-01-07 16:40:12,837Z [8] INFO SyncHost - Syncing local folder 'C:\sync' with cloud folder 2199894739371, direction: DownloadAndUpload, sort order: 0, resuming after interruption:
False.
2015-01-07 16:40:12,852Z [8] INFO SymformClient - CachePath="C:\sync\symform" MaxUploadRate=2500000 MaxDownloadRate=2500000 ProcessorUsageLimit=4 BufferSizeMultiplier=4
MaxUploadConcurrency=200 MaxDownloadConcurrency=200 OverwriteSharedThrottle=True
2015-01-07 16:40:12,915Z [8] INFO CryptoUtil - Selected 'MD5CryptoServiceProvider' as the default algorithm for MD5 with minBuffer = 1
2015-01-07 16:40:12,930Z [8] INFO CryptoUtil - Selected 'SHA1CryptoServiceProvider' as the default algorithm for SHA1 with minBuffer = 32
2015-01-07 16:40:12,977Z [8] INFO CryptoUtil - Selected 'SHA256Cng' as the default algorithm for SHA256 with minBuffer = 128
2015-01-07 16:40:13,118Z [8] INFO CryptoUtil - Selected 'AesCryptoServiceProvider' as the default algorithm for AES with minBuffer = 128
...
2015-01-07 16:40:13,619Z [8] INFO LocalProvider - Opening metadata with SQLite v3.8.3.1, connectionString="data source=C:\sync\symform\metadata.pooling=False,default
timeout=30;datetimeformat=ISO8601;binaryguid=True;default isolationlevel=ReadCommitted;journal mode=Wal,synchronous=Full;useutf16encoding=False;datetimekind=utc;PRAGMA
locking_mode=EXCLUSIVE"
...
2015-01-07 16:40:14,899Z [8] INFO SyncSessionMetricsCollector - Starting download session.
...
2015-01-07 16:40:18,227Z [8] INFO SyncSessionMetricsCollector - Downloading '3111.docx' [0.9c868e9100cbb0849eb4110bd15e8285.1 2.1] 0% completed
2015-01-07 16:40:19,118Z [3] INFO FileDownloadJob - download file is BE7822B608B8ABE5.709D779B for file path '3111.docx' and version 1.1.
2015-01-07 16:40:19,118Z [3] INFO TransferLogger - Preparing file '3111.docx' for download (2.67 KB)
2015-01-07 16:40:20,384Z [10] INFO S3BlockDownloader - downloading block [BE7822B608B8ABE5.709D779B-0] with request '23815d5a-64a8-4f43-85f3-48a72a9960c1' succeeded. Bytes
transferred: '2736'
2015-01-07 16:40:20,399Z [10] INFO SyncSessionMetricsCollector - Downloading '3111.docx' [0.9c868e9100cbb0849eb4110bd15e8285.1 2.1] 100% completed
2015-01-07 16:40:20,446Z [10] INFO TransferLogger - Stopped downloading file '3111.docx': no fragments transferred
2015-01-07 16:40:20,462Z [10] INFO SyncSessionMetricsCollector - Downloading '3111.docx' [0.9c868e9100cbb0849eb4110bd15e8285.1 2.1] 100% completed
2015-01-07 16:40:20,524Z [10] INFO SyncSessionMetricsCollector - Applied '3111.docx' [0.9c868e9100cbb0849eb4110bd15e8285.1 2.1]
```

Figure 12: An excerpt of symformsync.log recovered from our research.

```
2014-11-16T0055Z INFO MKDIR ./var/folders/hp/20p09h_973xe7zdyv2lv4hs0000gn/T/downloads 0 17 File exists2014-11-16T0056Z INFO [0x0x1008ac000] URL = POST
https://control.symform.com/scsion
2014-11-16T0056Z INFO [0x0x1008cc200] URL = GET https://control.symform.com/api/v1/account/
2014-11-16T0056Z INFO [0x0x1008dd600] URL = GET https://control.symform.com/api/v1/device
2014-11-16T0056Z INFO [0x0x1008df600] URL = GET https://control.symform.com/api/v0/folder?format=bin
2014-11-16T0057Z INFO [0x0x1008df600] URL = GET https://control.symform.com/api/v1/device
2014-11-16T0057Z INFO [0x0x101180600] URL = GET https://control.symform.com/Node/Restore??:id=DE396C632649BA9E0F45DD31FFEEB287EB90F812
2014-11-16T0057Z INFO [0x0x101180600] URL = GET https://control.symform.com/api/v0/folder?format=bin
2014-11-16T0057Z INFO [0x0x101180600] URL = GET https://control.symform.com/api/v1/device
2014-11-16T0057Z INFO [0x0x101180600] URL = PUT https://control.symform.com/api/v0/folder/2199753308332?nodeId=DE396C632649BA9E0F45DD31FFEEB287EB90F812
2014-11-16T0057Z INFO [0x0x1009a0600] URL = GET https://control.symform.com/node/DE396C632649BA9E0F45DD31FFEEB287EB90F812
2014-11-16T0057Z INFO [0x0x1009a0600] URL = PUT https://control.symform.com/node/DE396C632649BA9E0F45DD31FFEEB287EB90F812
2014-11-16T0057Z INFO [0x0x10119c200] URL = GET https://control.symform.com/api/v0/folder?format=bin
2014-11-16T0057Z INFO [0x0x1009ccc00] URL = GET https://control.symform.com/api/v1/device
2014-11-16T0057Z INFO [0x0x1009ccc00] URL = GET https://control.symform.com/node/DE396C632649BA9E0F45DD31FFEEB287EB90F812
2014-11-16T0057Z INFO [0x0x1009e1600] URL = GET https://control.symform.com/api/v1/account/
...
```

Figure 13: An excerpt of symformcloudcache.log.

### 4.3. Thumbnail cache

525 Thumbnail cache is a potential alternative source for recovery of images relevant to an investigation [60]. Analysis of the Thumbcache files stored under `%AppData% \Local\Microsoft\Windows\Explorer\` of the Install-VM (1.4) and Access-VM (1.4.1, 1.1.1 IE, 1.2.1 MF, and 1.3.1 GC) located copies of  
530 thumbnail images for the client or web application such as Symform logo and image icons appeared on the GUI. When the sample files were synced, we located copies of the thumbnail images in the Windows Thumbcache files. However, no thumbnail relevant to Symform was located in the Ubuntu and Mac OS VMs.

### 4.4. Web browser artefacts

535 Web browsing information is another potential source of information in cloud investigations [21, 22, 24, 46]. In this section, we present the artefacts of the Symform web application and RDM recovered in our experiments.

#### 4.4.1. Symform web application

Symform only allows users to download or delete files through its web  
540 application interface. Whilst accessing the web application, we observed that the username could be located at the top right corner of the browser. The web application retains a record of devices/nodes linked with an account in the left-hand pane of the browser. When we select a device, the web application will display the backup folders/files associated with the device.  
545 When hovered over an inactive device (marked with *X*), a message “Device has not been reported in *XX* days” will be displayed. The web application also has an option to show or restore deleted files (within 7 days of deletion). Unlike Dropbox and Google Drive investigated by Quick and Choo [21, 22], other than the duration (in days) from the last modified date, Symform web  
550 application does not show timestamp information such as last accessed time, file creation time, and last written time of the backup files.

Similar to any other web application, accessing Symform through a web browser would leave URLs referencing `www.symform.com` alongside the associated timestamp information and view counts in the  
555 browsing history. When we logged in Symform, we observed the URL `control.symform.com` in the web browsing history, as well as the download link `content.symform.com/api/v0/folder/[Folder global ID]/[Filename]` when we performed file download. When we accessed Symform using Google Chrome, we located the aforementioned URLs

560 in the *thumbnails* Table of the %AppData% \Local\Google\Chrome\User  
 Data\Default\Top Sites database, indicative of frequent Symform  
 usage. Moreover, the presence of the URLs aforementioned in  
 the %AppData% \Local\Google\Chrome\User Data\Default\Current  
 Tabs and %AppData% \Local\Google\Chrome\User  
 565 Data\Default>Last Session files of Google Chrome,  
 %AppData% \Roaming\Mozilla\Firefox\Profiles\%PROFILE%.default\  
 sessionstore.js of Mozilla Firefox, as well as  
 %AppData%\Local\Microsoft\Windows\WebCache\V01.log and  
 %AppData%\Local\Microsoft\Windows\WebCache\WebCacheV01.dat of  
 570 Internet Explorer would suggest recent Symform usage.

Within the web caches of all the investigating web browsers, we located  
 the intact copies of downloaded files (Figure 14) as well as Symform images,  
 HTML, Cascading Style Sheets (CSS), and Javascripts referencing *Symform*  
 used by the web application, which included the last access timestamp infor-  
 575 mation for the cache files. Additionally, when we closed the Mozilla Firefox  
 tab for Symform, we could locate thumbnail images for Symform page in  
 %AppData%\Roaming\Mozilla\Firefox\Profiles\%PROFILE%.default\  
 thumbnails\.

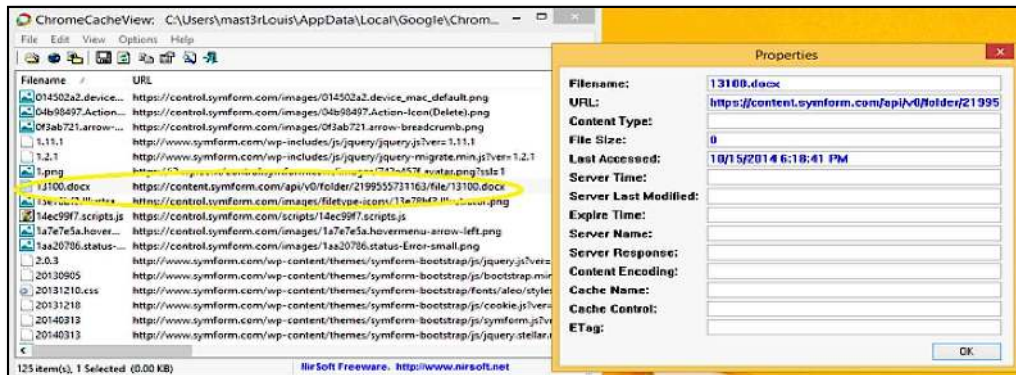


Figure 14: A copy Symform downloaded sample file recovered from Google Chrome browsing cache.

580 We could recover the login credentials (if saved) easily using  
 tools like Nirsoft Web Browser Passview [61]. We determined  
 that in Google Chrome, the credential details (login email

and encrypted password) are stored in the *Login Data* and *Web Data* files located under `%AppData%\Local\Google\Chrome\User Data\Default\`. Meanwhile, the credential details can be located  
585 in the `formhistory.sqlite` database and `logins.json` file under `%Users%\user\AppData\Roaming\Mozilla\Firefox\Profiles\` when saved in Mozilla Firefox. The login credentials recovered could facilitate user profiling and the accessing of the relevant accounts or applications.

#### 590 4.4.2. Remote Device Manager

In this research, we accessed the RDM using Mozilla Firefox v3.10 (the default Ubuntu web browser) on Ubuntu and the inbuilt Apple Safari browser (Version 7.0.6 (9537.78.2)) on the Mac OS machine. When we set up the client application using the RDM on  
595 Ubuntu, we recovered URLs making reference to `127.0.0.1:59234/tour`, `127.0.0.1:59234/setup`, and `127.0.0.1:59234/setup/done` in the browsing history and `sessionstore.js` file located in `/home/[User profile]/.mozilla/firefox/[Random ID].default/`, which comprised the associated timestamp information and view counts (in the former).  
600 When the sign-in occurred, we located URLs referencing `127.0.0.1:59234/login` and `127.0.0.1:59234/general` in the Ubuntu Mozilla Firefox files aforementioned as well as `/Users/[User Profile]/Library/Caches/Metadata/Safari/History.plist` Safari history file (when accessed on Mac OS). Additionally, we located a thumbnail image  
605 (screenshot) of the RDM webpage which appeared on the new tab page of Safari in `/Users/[User Profile]/Library/Caches/com.apple.Safari/Webpage Previews/`. This is indicative of Symform usage.

#### 4.5. Memory analysis

Memory forensics allow one to recover volatile information which would  
610 otherwise be lost [62]. In this research, we undertook data carving using Photorec, keyword search using Hex Workshop, and contextualising the RAM contents using Volatility.

When we analysed the running processes using the `pslist`, `linux_pslist`, and `mac_pslist` functions of Volatility, we recovered several process  
615 instances for Symform services, which included the process names (e.g., `symformstatus.exe`, `symformupdater.exe`, `symformcontrib.exe`, and `symformsync.exe` on Windows 8.1; `symformstatus`, `symformupdater`, `symformcontrib`, and `symformsync` on Ubuntu; `symform` on Mac), process

identifiers (PIDs), parent process identifiers (PPIDs), and the process initiation times; Figure 15 shows an example of the Windows processes. Our analysis also showed that the Symform processes are amalgamated into a single process, namely *Symform* on the Mac OS machine (in comparison with the process names presented for the Windows and Linux client applications). The PIDs could assist the investigator in obtaining data associated with the Symform client application during further analysis of the RAM (i.e., mapping a string of relevance to the instances resided in the memory space of the PID using the *yarascan* Function of Volatility). Analysing the network details using the *netscan*, *netstat\_linux*, and *netstat\_mac* functions of Volatility, we observed that the local, foreign, and peer node IP addresses could be recovered, which included its port numbers, socket states, PIDs, and process names. Such information is particularly useful for timeline analysis as well as in requests for assistance from counterparts overseas (e.g., via Interpol).

Offset(V)	Name	PID	PPID	Thds	Hnds	Sess	Wow64	Start	Exit
0xfffffa801a4b5940	services.exe	688	652	8	0	0	0	2014-11-15 17:55:14 UTC+0000	
0xfffffa801b5c0940	explorer.exe	1200	1848	53	0	1	0	2014-11-15 17:56:01 UTC+0000	
0xfffffa801b557080	symformstatus.exe.	3232	1200	16	0	1	0	2014-11-15 17:56:27 UTC+0000	
0xfffffa801a5b2940	symformupdater.exe	700	688	10	0	0	0	2014-11-15 17:58:40 UTC+0000	
0xfffffa801914a940	symformcontrib.exe	3704	688	15	0	0	0	2014-11-16 02:18:37 UTC+0000	
0xfffffa801b7af940	symformsync.exe	2488	688	21	0	0	0	2014-11-16 02:18:44 UTC+0000	

Figure 15: An excerpt of Symform processes recovered using the *pslist* Function of Volatility.

When we accessed Symform using the client applications, we were able to carve the image icons used by the client applications as well as files of forensic interest, such as *symformsync.log*, *node.config*, and the *metadata* database from the RAM captures. We could also recover the thumbnail images, script files, HTML documents, cache files, and other web browser files used by the web application after accessing the Symform web application. In all the investigated VMs, we could recover sample files from the RAM captures intact.

Manual analysis of the RAM captures of the client applications' file synchronisation VMs (1.4.2, 1.4.3, 2.3, 2.4, 3.3, and 3.4) revealed copies of the files of forensic interest in the memory space of *symformsync.exe* on Windows machine, *symformsync* on Ubuntu machine, and *symform* on Mac OS machine in plain text; Figure 16 and Figure 17 show examples of the remnants of the *node.config* and *metadata* database, respectively. This could prove useful to forensic practitioners when seeking to determine the origin of

the texts in the absence of the original files. Our findings suggested that  
 650 there are more than one way to recover the filename and path references  
 for synced files from the *symformsync.log* in the RAM, for example, by  
 searching for the terms *uploading* and *downloading*. Moreover, the terms  
 ‘*userName=*’ and *creationversion* could be used to locate remnants of the  
*node.config* and *metadata* database respectively during future searches.  
 655 When we used a browser to download the synced files, we were able to recover  
 copies of the download links from the RAM (see Figure 18) by searching for  
*content.symform.com*.

```

Owner: Process symformsync.exe Pid 1708
Ox72218cc476 75 73 65 72 4e 61 6d 65 3d [REDACTED] userName=[REDACTED]
Ox72218cc486 [REDACTED] password="AQAAAN
Ox72218cc496 70 61 73 73 77 6f 72 64 3d 22 41 51 41 41 41 4e CMnd8BFdERjHoAwE
Ox72218cc4a6 43 4d 6e 64 38 42 46 64 45 52 6a 48 6f 41 77 45 /C/+SBAAAA6cNaPY
Ox72218cc4b6 2f 43 6c 2b 73 42 41 41 41 41 36 63 4e 61 50 59 6b/UCVFhRCHFSGsg
Ox72218cc4c6 36 62 2f 55 43 56 46 68 52 63 48 46 53 47 73 67 QAAAAACAAAAAAQZg
Ox72218cc4d6 51 41 41 41 41 43 41 41 41 41 41 41 41 51 5a 67 AAAAEAAACAAACSUV
Ox72218cc4e6 41 41 41 41 45 41 41 43 41 41 41 41 43 53 55 76 yMkbHTBfT8yux5bw
Ox72218cc4f6 79 4d 4b 62 48 74 42 66 54 38 79 75 58 35 62 57 maTspScu2/576lLE
Ox72218cc506 6d 61 54 73 70 53 63 55 32 2f 35 37 36 6c 4c 45 Pg76pLEAAAAA0gA
Ox72218cc516 50 67 37 36 70 4c 45 41 41 41 41 41 41 41 4f 67 41 AAAAIAACAAADvIn
Ox72218cc526 41 41 41 41 49 41 41 43 41 41 41 41 41 44 76 49 6e ZahpBuN5ntF+zV70
Ox72218cc536 5a 61 68 70 42 75 4e 35 6e 74 46 2b 7a 56 37 30 9jgyu9QIYh+HHZ/o
Ox72218cc546 39 6a 67 79 75 39 51 49 59 68 2b 48 48 5a 2f 6f wUrIugdSAAAAAx
Ox72218cc556 77 55 72 69 75 47 64 53 41 41 41 41 41 6c 78 65 C2vkOWPGjwNYuBDy
Ox72218cc566 43 32 76 6b 4f 77 50 47 4a 77 4e 59 75 42 44 79
  
```

Figure 16: Remnants of *node.config* located within the memory space of *symformsync.exe*.

```

Task: symformsync pid 4153 rule r1 addr 0xb65ce031
Oxb65ce031 33 31 31 31 2e 7a 69 70 22 2c 22 69 73 53 75 62 3111.zip", "issub
Oxb65ce041 46 6f 6c 64 65 72 22 3a 66 61 6c 73 65 2c 22 69 Folder":false,"i
Oxb65ce051 73 54 6f 6d 62 73 74 6f 6e 65 22 3a 66 61 6c 73 sTombstone":fals
Oxb65ce061 65 2c 22 73 69 7a 65 22 3a 32 37 33 34 2c 22 72 e,"size":2734,"r
Oxb65ce071 65 64 75 6e 64 61 6e 74 53 69 7a 65 22 3a 32 37 edundantsize":27
Oxb65ce081 33 36 2c 22 6d 6f 64 69 66 69 65 64 54 69 6d 65 36,"modifiedTime
Oxb65ce091 55 74 63 22 3a 22 32 30 30 34 2d 31 31 2d 31 30 utc":"2004-11-10
Oxb65ce0a1 54 31 33 3a 34 30 3a 35 31 22 2c 22 69 73 52 65 T13:40:51","isRe
Oxb65ce0b1 73 74 6f 72 61 62 6c 65 22 3a 66 61 6c 73 65 2c storable":false,
Oxb65ce0c1 22 69 74 65 6d 49 64 22 3a 22 30 2e 64 63 31 34 "itemId":"0.dc14
Oxb65ce0d1 65 37 66 38 36 35 65 63 37 63 38 35 35 31 36 66 e7f865ec7c85516f
Oxb65ce0e1 33 62 61 65 33 63 35 66 61 36 34 32 2e 35 22 2c 3bae3c5fa642.5",
Oxb65ce0f1 22 63 72 65 61 74 69 6f 6e 56 65 72 73 69 6f 6e "creationversion
Oxb65ce101 22 3a 7b 22 72 65 70 6c 69 63 61 22 3a 31 2c 22 ":{"replica":1,"
Oxb65ce111 74 69 63 6b 22 3a 35 7d 2c 22 63 68 61 6e 67 65 tick":5},"change
Oxb65ce121 56 65 72 73 69 6f 6e 22 3a 7b 22 72 65 70 6c 69 version":{"repli
  
```

Figure 17: Remnants of *metadata* database located within the memory space of *symformsync*.





```

00C4B010 10 18 64 09 00 00 00 00 00 00 00 00 00 40 00 00 ..d.....@..
00C4B020 7B 22 65 6D 61 69 6C 22 3A 22 61 ██████████ {"email": "██████████"}
00C4B030 ██████████ 40 67 6D 61 69 6C 2E 63 6F 6D 22 2C ██████████ @gmail.com".
00C4B040 22 70 61 73 73 77 6F 72 64 22 3A 22 ██████████ "password": "██████████"
00C4B050 ██████████ 22 2C 22 64 75 72 61 ██████████ ██████████, "dura
00C4B060 62 6C 65 22 3A 74 72 75 65 2C 22 76 65 72 73 69 ble":true,"versi
00C4B070 6F 6E 22 3A 22 34 2E 32 32 2E 30 2E 30 22 7D 00 on":"4.22.0.0").

```

Figure 20: Symform login credentials recovered from RAM.

```

[+] Generic Password Record
[-] Create DateTime: 20141116005640Z
[-] Last Modified DateTime: 20141116005640Z
[-] Description :
[-] Creator :
[-] Type :
[-] PrintName : symform
[-] Alias :
[-] Account : username
[-] Service : symform
[-] Password
00000000: ██████████ 67 6D 61 69 6C ██████████ @gmail
00000010: 2E 63 6F 6D ██████████ .com

[+] Generic Password Record
[-] Create DateTime: 20141116005640Z
[-] Last Modified DateTime: 20141116005640Z
[-] Description :
[-] Creator :
[-] Type :
[-] PrintName : symform
[-] Alias :
[-] Account : password
[-] Service : symform
[-] Password
00000000: ██████████ ██████████

```

Figure 21: An excerpt of chainbreaker.py output.

When we searched for the term *Symform* in the swap files (also known as *pagefile.sys* on Windows [63]) of all the file synchronisation VMs (1.1.2 IE, 1.2.2 MF, 1.3.2 GC, 1.4.2,1.4.3, 2.3, 2.4, 3.3, and 3.4), we could only recover *filename/path* references for the synced files and client application files (when accessed using the client applications). Although the findings from swap files were not as conclusive as those presented for the RAM analyses, the presence of the *filename/path* references in the swap files could prove useful to indicate recent Symform usage.

690 *4.6. Unallocated space*

Unallocated space is a potential source of information in an investigation, such as intact files as well as filename and path references that were created and deleted by the user or the system [21, 22]. Similar to the RAM analysis, in our examination of the unallocated client applications' file synchronisation  
695 VMs (1.4.2, 1.4.3, 2.3, 2.4, 3.3, and 3.4), we identified that the files of forensic interest such as `symformsync.log`, `node.config`, `synced files`, as well as the *metadata* database could be recovered from the unallocated spaces intact. Moreover, we were able to recover copies of the image icons and files of relevance (e.g., script files, HTML documents) used by the client application,  
700 RDM, and web application from the unallocated spaces; indicative of Symform usage. The presence of the data remnants aforementioned in the Delete-VM (1.4.3.1, 2.3.1, 3.3.1) and Uninstall-VM (1.4.2.1, 2.4.1, 3.4.1) reinforced that unallocated partition is an important source for recovering deleted Symform or synced files.

705 *4.7. Windows system files*

In this section, we present the Symform artefacts located within the registry, prefetch and link files of Windows 8.1.

*4.7.1. Registry files*

Windows registry provides a rich source of information associated with  
710 installed programs [64]. Although five hives could be seen in the registry, only HKEY\_USERS (HKU) and HKEY\_LOCAL\_MACHINE (HKLM) hives are tangibly real, since the remaining are merely symbolic links to the two master keys [65]. An analysis of the registry of the Install-VM (1.4) revealed that the installation of the Symform client application would result in the addition  
715 of approximately 82 registry keys. We could detect the installation from the presence of entries referencing Symform executable files, the Symform version installed, the installation paths, Symform URLs, the date it was installed, or other options in the following registry keys:

- 720 • HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Installer\UserData\S-1-5-18\Products\[Product GUID]\InstallProperties
- HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\{[Product GUID]}

In addition to the aforementioned registry keys, we located descriptions and full path references for Symform services (e.g., symformsync, symformupdater, and symformcontrib) in the following registry subkeys:

- HKLM\SYSTEM\ControlSet001\Services\[Symform service name]
- HKLM\SYSTEM\CurrentControlSet\Services\[Symform service name]

While using the Symform client application, we observed that six sub-branches for Symform setup, contribution, and status services were created in the Windows Routing and Remote Access service (RRAS) registry keys listed below. According to [66], the keys are used to enable file tracing for diagnosing network problems (relating to the Symform client application in this case). The keys remained in the registry even after the Symform client application was uninstalled.

- HKLM\SOFTWARE\Microsoft\Tracing\Symformcontrib\_RASAPI32
- HKLM\SOFTWARE\Microsoft\Tracing\Symformcontrib\_RASMANICS
- HKLM\SOFTWARE\Microsoft\Tracing\Symformsetup\_RASAPI32
- HKLM\SOFTWARE\Microsoft\Tracing\Symformsetup\_RASMANICS
- HKLM\SOFTWARE\Microsoft\Tracing\Symformstatus\_RASAPI32
- HKLM\SOFTWARE\Microsoft\Tracing\Symformcontrib\_RASMANICS

Each time an application is started, Windows automatically extracts the application name from the version resource of the executable file and stores it in the *MuiCache* registry key for later use [67]. When the Symform client application was started up, we located entries referencing Symform executable files in the following *MuiCache* keys:

- HKU\< SID >\Classes\Local Settings\MuiCache
- HKU\S-1-5-21\Software\Classes\Local Settings\MuiCache

Similar to any other Windows application, when we configured Symform  
750 to run automatically whenever Windows starts, we located entries referencing  
*Symform* under `Software\Microsoft\Windows\CurrentVersion\Run`  
registry key (Figure 22 depicts an example). Other evidence indi-  
cating the use of the Symform client application include the pres-  
ence of entries referencing `%Desktop%\SymformNodeNew.exe` (1) as  
755 as well as the time when the client application was last executed in  
`Software\Microsoft\Windows\CurrentVersion\Explorer\UserAssist`  
(which holds information about the .EXE files and links that a user opens  
frequently [68]).

```
user_run v.20140115
(NTUSER.DAT) [Autostart] Get autostart key contents from NTUSER.DAT hive

Software\Microsoft\Windows\CurrentVersion\Run
LastWrite Time Mon Oct 13 07:13:53 2014 (UTC)
Symform Status: "C:\Program Files\Symform\Node Service\symformstatus.exe"

listsoft v.20080324
(NTUSER.DAT) Lists contents of user's Software key

listsoft v.20080324
List the contents of the Software key in the NTUSER.DAT hive
file, in order by LastWrite time.

Tue Oct 14 15:55:15 2014      Microsoft
Tue Oct 14 15:04:44 2014      BreakPoint License Manager
Mon Oct 13 17:02:13 2014      Macromedia
Mon Oct 13 07:13:53 2014      Symform
Mon Oct 13 07:09:22 2014      Mozilla
Wed Sep 24 05:52:08 2014      Policies
```

Figure 22: Symform entry located under `Software\Microsoft\Windows\CurrentVersion\Run`.

760 The *TypedURLs* key contains a listing of 25 recent URLs (or file  
path) typed in the Internet Explorer or Windows Explorer address  
bar [69]. When we used Internet Explorer to access Symform ac-  
count, we could locate references to Symform URLs and the accessed  
times under `Software\Microsoft\Internet Explorer\TypedURLs` and  
765 `Software\Microsoft\Internet Explorer\TypedURLsTime` registry keys re-  
spectively, Figure 23 shows an example. It is to the best of the authors'  
knowledge that none of the remaining browsers utilise the registry in the way  
that Internet Explorer does.

```
typedurls v.20080324
(NTUSER.DAT) Returns contents of user's TypedURLs key.

TypedURLs
Software\Microsoft\Internet Explorer\TypedURLs
LastWrite Time Thu Sep 4 14:38:24 2014 (UTC)
url1 -> http://symform.com/

typedurlstime v.20120613
(NTUSER.DAT) Returns contents of user's TypedURLsTime key.

TypedURLsTime
Software\Microsoft\Internet Explorer\TypedURLsTime
LastWrite Time Thu Sep 4 14:38:24 2014 (UTC)
url1 -> Thu Sep 4 14:38:24 2014 Z (http://symform.com/)
```

Figure 23: Symform URLs located under Software\Microsoft\Internet Explorer\TypedURLs and \Software\Microsoft\Internet Explorer\TypedURLsTime.

Another registry key of forensic interest is the Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32. According to [70], the *CIDSizeMRU* (MRU is the abbreviation for Most-Recently-Used) subkey maintains a list of recently used applications, the *OpenSaveMRU* registry subkey records list of files that have been opened or saved within a Windows shell dialog box, and the *LastVisitedMRU* subkey is responsible for tracking specific executable files used by an application to open the files documented in the *LastVisitedMRU* subkey. When we downloaded the sample files, we identified that there were various entries making reference to Symform or the browser's executable file (when downloaded using a browser) in the aforementioned registry subkeys, which included *filenames* for the downloaded files and references to the last accessed times (see Figure 24).

The *RecentDocs* key corresponds to %Recent%\ (My Recent Documents), which maintains a list of local or network files recently executed or opened through Windows Explorer [71]. Undertaking a download of the sample files resulted in the creation of entries making reference to filenames and file extensions for the downloaded files as well as the last written times in Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs (see Figure 25). We were also able to locate entries referencing the HTML document of the Symform web application (namely *Symform Web App.htm*) in the *RecentDocs* key after we had accessed the Symform account using a browser.

```
-----
comdlg32 v.20121008

Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32
LastWrite Time Fri Sep 19 02:50:41 2014 (UTC)
CIDSSizeMRU
LastWrite: Tue Oct 14 15:59:19 2014
Note: All value names are listed in MRUListEx order.

    Safari.exe
LastVisitedPidlMRU
LastWrite: Tue Oct 14 15:59:19 2014
Note: All value names are listed in MRUListEx order.

    Safari.exe - Dataset\
OpenSavePidlMRU
LastWrite: Tue Oct 14 15:59:19 2014
OpenSavePidlMRU\*
LastWrite Time: Tue Oct 14 15:59:19 2014
Note: All value names are listed in MRUListEx order.

    Dataset\dfwrs-13-challenge-tests.zip
    Dataset\13100.txt
    Dataset\13100.rft
    Dataset\13100.jpg
    Dataset\13100.docx
```

Figure 24: File download information located in Software\Microsoft\Windows\CurrentVersion\Explorer\ComDlg32.

#### 4.7.2. Prefetch files

A prefetch file contains information about a loaded application on Windows (e.g., executable filename, path, associated dlls, number of times an application has been loaded, last run time, and other associated timestamps). An examination of the prefetch files in %SystemRoot%\Prefetch of the Install-VM (1.4) identified that the installation would result in the creation of Symform prefetch files such as SYMFORMSETUP.EXE.pf, SYMFORMNODENEW.EXE.pf, and SYMFORMUPDATER.EXE.pf. When we performed file synchronisation, we could observe the prefetch files namely SYMFORMCONTRIB.EXE.pf (for the contribution service), SYMFORMSTATUS.EXE.pf, and SYMFORMSYNC.EXE.pf in the file synchronisation VMs (1.4.2 and 1.4.3). However, we could not locate any prefetch entry relating to the synced files. We determined that the Symform prefetch files remained in the hard drive even after uninstalling the client application.

```

RecentDocs v.20100405
(NTUSER.DAT) Gets contents of user's RecentDocs key

RecentDocs
**All values printed in MRUList\MRUListEx order.
Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs
LastWrite Time Tue Oct 14 20:30:55 2014 (UTC)

  0 = dfws-13-challenge-tests.zip
  1 = Browse_Symform Web App.htm
  2 = 13100.txt
  3 = 13100.rft
  4 = 13100.jpg
  5 = 13100.docx

Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\docx
LastWrite Time Mon Oct 14 07:17:17 2014 (UTC)
MRUListEx = 0
  0 = 13100.docx

Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\txt
LastWrite Time Mon Oct 14 07:17:17 2014 (UTC)
MRUListEx = 0
  0 = 13100.txt

Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\jpg
LastWrite Time Mon Oct 14 07:17:17 2014 (UTC)
MRUListEx = 0
  0 = 13100.jpg

Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\rft
LastWrite Time Mon Oct 14 07:17:17 2014 (UTC)
MRUListEx = 0
  0 = 13100.rft

Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\zip
LastWrite Time Mon Oct 14 07:17:17 2014 (UTC)
MRUListEx = 0
  0 = dfws-13-challenge-test.zip

Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs\htm
LastWrite Time Mon Oct 14 07:17:17 2014 (UTC)
MRUListEx = 0
  0 = Browse _ Symform Web App.htm

```

Figure 25: Recently opened documents located in Software\Microsoft\Windows\CurrentVersion\Explorer\RecentDocs.

805 4.7.3. *Link files*

Link (.lnk) files are shortcut metadata files used by Windows to maintain a list of linked paths relating to a file (commonly the paths where the original files are located), associated timestamps (create, write, and last accessed times), and file sizes (original and modified), which could be used to identify the origin of a file [72]. Analysing the link files of the client application's file synchronisation VMs (1.4.2 and 1.4.3), we located two link files namely Symform Setting.lnk and Symform Status.lnk for %Program Files% \Symform\Node Service\symformsetup.exe and %Program Files% \Symform\Node Service\symformstatus.exe, respectively in %ProgramData% \Microsoft\Windows\Start Menu\Programs\Symform\.

815 Of all the VMs investigated, we were only able to locate link files for the sample files under %AppData% \Roaming \Microsoft\Windows\Recent\ of the Mozilla Firefox Download-VM (1.2.2).



## 5. Symform analysis on mobile devices

820 In this section, we present the findings of our analysis on iPhone 4 running  
iOS 7.1.2 and an HTC One X running Android KitKat 4.4.

### 5.1. Symform analysis on iOS 7.1.2

Examination of the directory listing identified that the Symform  
iOS app installation could be discerned from the presence of the  
825 `/private/var/mobile/Applications/[Unique SHA-1 identifier  
for the Symform iOS app]/Symform.app` application folder. The  
`/private/var/mobile/Applications/[Unique SHA-1 identifier for  
the Symform iOS app]/iTunesMetadata.plist` maintains a list of mobile-  
specific metadata associated with the Symform app such as the Apple ID  
830 used to purchase the app, the purchase date, the Symform version installed  
and other information, as detailed in Figure 26.

The `Cache.db` (see Section 4.1.3) for the Symform iOS app could be located  
in `/private/var/mobile/Applications/[Unique SHA-1 identifier for  
the Symform iOS app]/Library/Caches/com.symform.ios.Symform/`.  
835 When we viewed the sample files, we located copies of the files in  
`/private/var/mobile/Applications/[Unique SHA-1 identifier for  
the Symform iOS app]/private/var/mobile/Applications/[Unique  
SHA-1 identifier]/tmp/downloads/` intact, which included the file formats  
and viewing timestamps.

840 Whilst viewing the synced files in the app, we observed that Symform  
iOS application (Version 1.13) offers an option to upload logs. When  
this option is enabled, a copy of the `symformcloudcachelogs.log`  
(see Section 4.2.3) will be emailed to the account owner. In ad-  
dition, we could detect the trace of Symform usage from the  
845 `/private/var/log/DiagnosticMessages/YYYY.MM.DD.asl` log of iOS.  
Other than the Kernel and system events (similar to those discussed  
in Section 4.2.3), we could recover the file download URLs as well as  
the timestamp information from the ASL logs (see Figure 27). Neither  
Symform-related folder nor files remained after the uninstallation of the  
850 Symform iOS app.

Type	Name	Value
... AnsiString	softwareIcon57x57URL	http://a1871.phobos.apple.com/us/r30/Purple4/v4/a0/7
... Integer	s	143441
... Integer	itemId	682011904
... Boolean	softwareIconNeedsShine	False
... AnsiString	playlistName	Symform
... AnsiString	softwareVersionBundleId	com.symform.ios.Symform
... Integer	softwareVersionExternalIdentifier	517612672
... Integer	versionRestrictions	16843008
[-] Dictionary	com.apple.iTunesStore.downloadInfo	
[-] Dictionary	etags	
... AnsiString	artwork	"d195-4f602de7ea7ee"
... AnsiString	media	"26576f73a38702c5530c43190b31a310:1396550934"
[-] Dictionary	accountInfo	
... Boolean	AccountSocialEnabled	False
... Integer	AccountAvailableServiceTypes	0
... Integer	AccountServiceTypes	0
... Boolean	DidFallbackToPassword	False
... AnsiString	AccountStoreFront	143441-1,24
... AnsiString	AccountURLBagType	production
... Integer	DSPersonID	2031127169
... Boolean	AccountIsNewCustomer	False
... AnsiString	CreditDisplayString	
... AnsiString	AccountSource	device
... Integer	AccountKind	0
... AnsiString	AppleID	[REDACTED]
... AnsiString	purchaseDate	2015-02-04T12:27:01Z
... AnsiString	artworkAssetFilename	-4014248785902955583.jpeg
... AnsiString	mediaAssetFilename	6647492915351722469...app
... AnsiString	itemName	Symform
[-] Dictionary	rating	
... AnsiString	bundleVersion	1.13
... Integer	drmVersionNumber	0
... AnsiString	fileExtension	.app
... Integer	genreId	6007
... Integer	artistId	682011907
[-] Dictionary	asset-info	
... AnsiString	releaseDate	2013-10-08T22:46:56Z
... AnsiString	artistName	Symform
... AnsiString	playlistArtistName	Symform
... AnsiString	bundleShortVersionString	1.13
[-] Array	softwareSupportedDeviceIds	
... Integer	vendorId	2114536

Figure 26: Content of iTunesMetadata.plist.

```
ASL LOGS
(
  {
    ASLMessageID = 1153721;
    Facility = com.symform.ios.Symform;
    Level = 4;
    Message = URL = privatevarmobile.ApplicationsECA8178C-SED7-401E-AB72-5D978B0D384DtmpdownloadsEC1BC9A0891364EF06EF7EC827CC333F25192B84_0_1411728190.txt;
    PID = 8955;
    ReadUID = 501;
    Sender = Symform;
    Sender_Mach_UUID = E299BD2A-E45C-3363-8A0C-475F81D41A90;
    Time = 1411728194;
    TimeNanoSec = 87680000;
  },
  {
```

Figure 27: An example of ASL log containing file download information of the Symform iOS app.

## 5.2. Symform analysis on Android KitKat 4.4

The installation of the Symform Android app resulted in the creation of `/data/data/com.symform.android.symform` folder which has five subfolders, namely `app_webview`, `cache`, `files`, `lib`, and `shared_prefs`. The file of particular interest is `/data/data/com.symform.android.symform/shared_prefs/SymformPrefs.xml` file, which stores the username (email address) and encrypted password used to login the app. Meanwhile, the folder of interest is the `/data/data/com.symform.android.symform/files/downloads` folder, which holds a list of files downloaded to the device. No Symform-related folder or file remained after uninstalling the Android app.

## 6. Network analysis

When accessing Symform using a web browser, we established the initial session with the main web server (`www.symform.com` with IP address 104.130.154.151 in our research) and then to the login page (`control.symform.com` with IP address 173.193.191.132 in our research) over port 80 (HTTP). We observed the network traffic only on port 443 (HTTPS) as soon as sign-in took place, and Starfield Technologies [73] issued the certificates. Also occurring was a session with Google Analytic services (e.g., IP addresses 74.125.\*.\* in our research) during sign-in. The next servers accessed were the AmazonAWS servers (e.g., IP addresses 54.231.\*.\*) which host Amazon Web Services (EC2) with additional information referencing Symform. An examination of the client application's network traffic revealed similar observations, but we did not observe any session established with the Symform web server.

When we downloaded the sample files using a web browser, we located URLs that made reference to `content.symform.com` within the TCP stream. When file synchronisation was undertaken using the client application, we identified the UDP as the carrying protocol and we could recover the IP addresses of the peer nodes. However, the port numbers appeared to be random, thereby, making the ports unpredictable. Although the files were encrypted using 256-bit AES encryption, we were able to locate remnants of the HTTP requests for the backup file fragments including the requests IDs in the UDP stream (see Figure 28). The corresponding timestamp information recorded alongside the relevant IP addresses could be used to facilitate re-constructing of the user activity timeline.

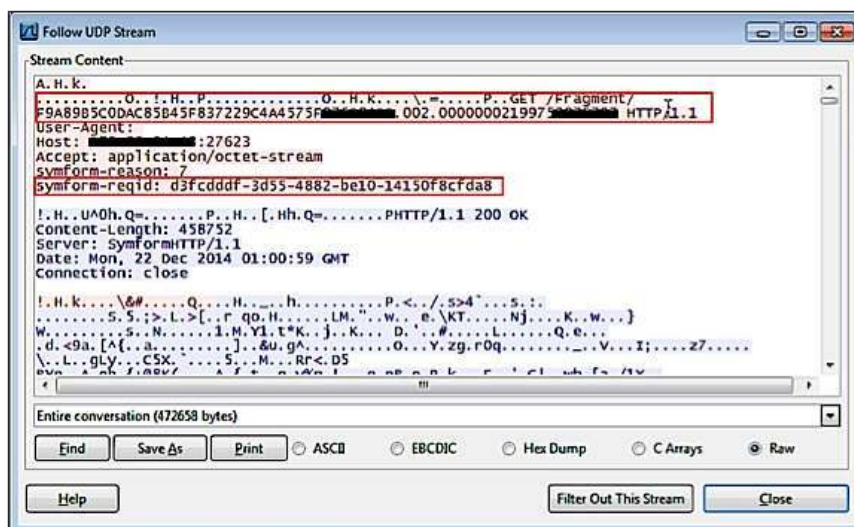


Figure 28: An excerpt of UDP stream containing remnants of the HTTP request for a backup file fragment.

Rebuilding the network captures of Symform web application using *Network Miner*, we recovered the HTML documents, Javascripts, CSS files, and image files from the unencrypted traffic. Of all the platforms investigated, we were able to recover certificates used to authenticate the HTTPS sessions.

## 7. Concluding Remarks

The increasing popularity of cloud computing among individual users and organisations, as well as criminals resulted in the need for forensic investigators to have a contemporary understanding of the artefacts that may be left behind by different types of cloud storage system on a client device (e.g., personal computers and mobile devices). In this paper, we described the potential terrestrial artefacts that may be left behind on a client device running Window 8.1, Ubuntu 14.04 LTS, Mac OS X Mavericks 10.9.5, iOS 7.1.2 and Android KitKat 4.4 after the use of Symform (a cooperative storage cloud service).

Our timestamp analysis indicated that the MD5 and SHA1 hash values did not change during the process of uploading, storage, and downloading files from Symform. Examinations of the directory listings revealed that the Symform client application maintains a list of log files for the main

services under %Program Files% \Symform\Node Service\logs\ on Windows 8.1, /var/log/symform/ on Ubuntu, as well as /Library/Application Support/Symform/bin/ and /private/var/logs/Symform/ on Mac OS. These logs could be useful in recovering history of Symform usage. This  
910 was not the case for machines and devices running Ubuntu, Mac OS, and Android (i.e. we could not recover any system log that could be used to identify synced files/folders).

The `node.config` file (located under %Program Files% \Symform\Node Service\ on Windows 8.1, /var/lib/symform/ on Ubuntu, and  
915 /private/var/lib/ on Mac OS) is another potential evidence source, which contains node-specific configuration details for the file synchronisation and contribution services. In all the operating systems investigated, two hidden subfolders namely `.symform` and `.symform-store` were created in the sync/backup folders to store the synced file caches. The  
920 *metadata* database located within the `.symform` subfolder would be of particular interest to forensic practitioners when seeking to recover the synced file history of the sync folder. The deleted files could be potentially located in the %\$Recycle.Bin% \SID folder on Windows 8.1, /home/[User Profile]/.local/share/Trash/files folder on Ubuntu, and  
925 /Users/[User profile]/.Trash folder on Mac OS when not emptied. The filename and location references, located as part of our research, may facilitate the identification of other sources of evidence and result in timely preservation of the evidence.

When we accessed Symform using a web browser, we were  
930 able to locate URLs referencing `www.symform.com` (Symform webpage), `control.symform.com` (Symform login link), and `content.symform.com/api/v0/folder/[Folder global ID]/[Filename]` (Symform file download link) alongside the associated timestamps and the view counts in the web browsing information. Meanwhile, accessing the RDM  
935 would leave URLs referencing `127.0.0.1:59234` in the web browsing history. The presence of the downloaded files in the web browsing caches would create potential for alternative methods for recovery of the downloaded files.

Undertaking data carving of the unallocated partitions and memory files confirmed that we could recover the files of forensic interest from unallocated  
940 partitions and memory files. When we used a browser to access Symform web application, the username and password could be potentially recovered from the RAM. Analysing the system processes using the *'pslist'* function of Volatility, we could discern the process names from *symformweb*, *symformsync*,

and *symformcontrib* on Windows and Ubuntu OS, and *Symform* on Mac  
945 OS. However, it is noteworthy that we captured the RAM immediately after  
performing each experiment, but prior to a system shutdown in our research.  
Thus, the data remnants identified in this research do not represent those  
recoverable in a typical “real world” circumstance, unless capturing the RAM  
on a suspect system “in real life” immediately after the action but prior to a  
950 shutdown.

In our investigations on the mobile applica-  
tions, we forensically recovered copies of the viewed  
files from `/private/var/mobile/Applications/[Unique  
SHA-1 identifier]/tmp/downloads/` on iOS and  
955 `/data/data/com.symform.android.symform/files/downloads/` on  
Android devices. In our examinations of the network captures, we determined  
that most of the data from the application layer were encrypted, but the IP  
addresses of the peer nodes could be located from the UDP traffic. When  
file synchronisation took place using the client application, we were able to  
960 forensically recover the request IDs for the backup file fragments from the  
UDP stream. The corresponding timestamp information recorded alongside  
the relevant IP addresses is, particularly, useful for timeline analysis. The  
summary of findings from the mobile and computer devices investigated in  
this research study is presented in tables 6 and 7.

965 To keep pace with technological advances, future work would include  
extending this research to other popular cooperative storage cloud services  
(e.g., Storj), as well as developing a forensically sound tool to automate  
collection of artefacts common to popular cooperative storage cloud services.

Table 6: Summary (R =Recoverable, P = Possibly Recoverable, N = Not Recoverable).

Platform	Source of Evidence	Data artefacts found		
		Installation/ uninstallation information	Username/ Email address	Password
Windows 8.1(client and web applications)	Directory listings/ Stored files	R	R	R
	Registry files	R	N	N
	Log files	R	R	N
	Web browser files	P, client app download link	P	R, only saved credentials in the web browser
	Prefetch	R, prefetch files for the executable files	N	N
	Thumbcache files	R, client app icons	N	N
	Link files	R	N	N
	Fileslack	N	N	N
	RAM	P	P	P, only web app login pass- word
	Pagefile.sys Unallocated space	P P	P P	N N
Ubuntu 14.04 LTS	Directory listings/ Stored files	R	R	R, login password from node.config
	Log files	R	R	N
	Web browser files (only RDM)	P, client app download link	P	N
	Thumbcache files	N	N	N
	RAM	P	P	P, RDM's login password
	Swap partition	P	P	N
	Unallocated partition	P	P	N
Mac OS X Mavericks 10.9.5	Directory listings/ Stored files	R	R	R, login password from node.config
	Log files	R	R	N
	Web browser files	P, client app download link	P	N
	Thumbcache files	N	N	N
	RAM	P	P	P, RDM's login password
	Swap partition	P	P	N
	Unallocated partition	P	P	N
iOS 7.1.2	Directory listings/ Stored files	R	R	N
	Log files	P	N	N
Android Kitkat 4.4	Directory listings/ Stored files	R	N	R, login password from SymformPrefs.xml
Network traffic		R	N	N

Table 7: Summary (R =Recoverable, P = Possibly Recoverable, N = Not Recoverable).

Platform	Source of Evidence	Data artefacts found		
		Symform log or configuration files/path references	Symform URLs	Enron sample files/path references
Windows 8.1(client and web applications)	Directory listings/	R, login password for the client app from node.config	R	R
	Stored files	N	R	P
	Registry files	R, copies of node.config file in symformsync.log	R	R
	Log files	N	R	P
	Web browser files	N	N	N
	Prefetch	N	N	R, only the synced Enron image
	Thumbcache files	N	N	N
	Link files	N	N	N
	Fileslack	P	P	P
	RAM	P	P	P
Pagefile.sys	P	P	P	
Unallocated space	P	P	P	
Ubuntu 14.04 LTS	Directory listings/	R	R	R
	Stored files	R, copies of node.config file in symformsync.log	R	R
	Log files	N	R	N
	Web browser files (only RDM)	N	N	N
	Thumbcache files	P	P	P
	RAM	P	P	P
	Swap partition	P	P	P
Unallocated partition	P	P	P	
Mac OS X Mavericks 10.9.5	Directory listings/	R	R	R
	Stored files	R, copies of node.config file in symformsync.log	R	R
	Log files	N	R	N
	Web browser files	N	N	N
	Thumbcache files	P	P	P
	RAM	P	P	P
	Swap partition	P	P	P
Unallocated partition	P	P	P	
iOS 7.1.2	Directory listings/	R	N	R
	Stored files	N	P, only in the ASL log	N
Android Kitkat 4.4	Log files	N		
	Directory listings/	N	N	R
Network traffic	Stored files			
		N	R, IP addresses of the peer nodes	N



## References

- 970 [1] Forecast: It services, 2011-2017, 4q13 update, <https://www.gartner.com/doc/2637515/forecast-it-services--q>(Accessed 20 November 2014) (2013).
- [2] F. Gens, M. Adam, D. Brandshaw, C. Christiansen, Worldwide and regional public IT cloud services 2013-2017 forecast, <http://www.idc.com/getdoc.jsp?containerId=242464> (Accessed 20 November 2014) 975 (2013).
- [3] K. K. R. Choo, Organised crime groups in cyberspace: a typology, *Trends in organized crime* 11 (3) (2008) 270–295.
- [4] K.-K. R. Choo, et al., Cloud computing: challenges and future directions, *Trends and Issues in Crime and Criminal Justice* (2010) 1–6. 980
- [5] J. Galante, O. Kharif, P. Alpeyev, Sony network breach shows amazon clouds appeal for hackers, <http://www.bloomberg.com/news/2011-05-15/sonyattack-shows-amazon-s-cloud-service-lures-hackers-at-pennies-an-hour.html> (Accessed 5 June 2014) (2011).
- 985 [6] Symantec, The trojan. hydraq incident: Analysis of the aurora 0-day exploit, <http://www.symantec.com/connect/blogs/trojanhydraq-incident-analysis-aurora-0-day-exploit> (Accessed 20 November 2014) (2011).
- [7] A. Duke, 5 things to know about the celebrity nude photo hacking scandal, <http://edition.cnn.com/2014/09/02/showbiz/hacked-nude-photos-five-things/> (Accessed 18 November 2014) (2014). 990
- [8] R. Lemos, Cloud-based denial of service attacks looming, researchers say, <http://www.darkreading.com/smb-security/167901073/security/perimeter-security/226500300/index.html>(Accessed 27 November 995 2014) (2010).
- [9] C. Bagh, Amazon ec2 helps researcher to crack wi-fi password in 20 minutes, <http://www.ibtimes.com/articles/100314/20110112/amazon-ec2-password-wi-hacking-cracking-brute-force-attack-wpa-psk-encryption-cloud-computing-iaa.htm> (Accessed 20 November 1000 2014) (2011).

- [10] M. Taylor, J. Haggerty, D. Gresty, P. Almond, T. Berry, Forensic investigation of social networking applications, *Network Security 2014* (11) (2014) 9–16.
- [11] H. Chung, J. Park, S. Lee, C. Kang, Digital forensic investigation of cloud storage services, *Digital investigation* 9 (2) (2012) 81–95.
- [12] G. Grispos, T. Storer, W. B. Glisson, Calm before the storm: the challenges of cloud, *Emerging Digital Forensics Applications for Crime Detection, Prevention, and Security* 4 (2) (2013) 28–48.
- [13] C. Hooper, B. Martini, K.-K. R. Choo, Cloud computing and its implications for cybercrime investigations in australia, *Computer Law and Security Review* 29 (2) (2013) 152–163.
- [14] D. Quick, K.-K. R. Choo, Forensic collection of cloud storage data: Does the act of collection result in changes to the data or its metadata?, *Digital Investigation* 10 (3) (2013) 266–277.
- [15] N. I. of Standards, T. (NIST), Nist cloud computing forensic science challenges, <http://safegov.org/media/72648/nist-digital-forensics-draft-8006.pdf> (Accessed 28 October 2014) (2014).
- [16] B. Martini, K.-K. R. Choo, Cloud forensic technical challenges and solutions: a snapshot, *IEEE Cloud Computing* 1 (4) (2014) 20–25.
- [17] D. Quick, B. Martini, R. Choo, *Forensics Cloud storage*, Syngress, 2014.
- [18] G. Hogben, M. Dekker, Procure secure: A guide to monitoring of security service levels in cloud contracts, *European Network and Information Security Agency (ENISA) Report*.
- [19] C. Tassone, B. Martini, K.-K. R. Choo, J. Slay, et al., Mobile device forensics: A snapshot (2013) 1–7.
- [20] J. S. Hale, Amazon cloud drive forensic analysis, *Digital Investigation* 10 (3) (2013) 259–265.
- [21] D. Quick, K.-K. R. Choo, Digital droplets: Microsoft skydrive forensic data remnants, *Future Generation Computer Systems* 29 (6) (2013) 1378–1394.

- [22] D. Quick, K.-K. R. Choo, Dropbox analysis: Data remnants on user machines, *Digital Investigation* 10 (1) (2013) 3–18.
- [23] B. Martini, K.-K. R. Choo, Remote programmatic vcloud forensics: a six-step collection process and a proof of concept, in: Proc. of the 13th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom'14), IEEE, 2014, pp. 935–942.
- 1035
- [24] D. Quick, K.-K. R. Choo, Google drive: Forensic analysis of data remnants, *Journal of Network and Computer Applications* 40 (2014) 179–193.
- [25] J. Farina, M. Scanlon, M.-T. Kechadi, Bittorrent sync: first impressions and digital forensic implications, *Digital Investigation* 11 (2014) S77–S86.
- 1040
- [26] M. Shariati, A. Dehghantanha, K.-K. R. Choo, Sugarsync forensic analysis, *Australian Journal of Forensic Sciences* (ahead-of-print) (2015) 1–23.
- [27] M. Shariati, A. Dehghantanha, B. Martini, K. Choo, Ubuntu one investigation: Detecting evidences on client machines, *Cloud Security Ecosystem*, Syngress, an Imprint of Elsevier.
- 1045
- [28] B. Martini, Q. Do, K.-K. R. Choo, Mobile cloud forensics: An analysis of seven popular android apps, *Cloud Security Ecosystem*, Syngress, an Imprint of Elsevier.
- [29] Q. Corporation, Affordable cloud storage pricing plans, <http://www.symform.com/plans-pricing/> (Accessed 28 November 2014) (2013).
- 1050
- [30] Q. Corporation, Symform announces worlds first cooperative storage exchange, <http://www.symform.com/news/press-releases/symform-announces-worlds-first-cooperative-storage-exchange/> (Accessed 2 July 2014) (2013).
- 1055
- [31] Q. Corporation, Symform announces 11 million\$ series b with strong business momentum, <http://www.symform.com/news/press-releases/symform-wins-two-awards/> (Accessed 21 November 2014) (2013).
- [32] Q. Corporation, Symform wins two awards for its disruptive cloud storage technology and leading corporate culture, <http://www.symform.com/news/press-releases/11-million-series-b-announce/> (Accessed 21 November 2014) (2013).
- 1060

- 1065 [33] S. Works.com, Resilient storage architecture, <http://www.symformworks.com/Resilient-Storage-Architecture.asp> (Accessed 20 November 2014) (2014).
- [34] D. Birk, C. Wegener, Technical issues of forensic investigations in cloud computing environments, in: Proc. of Sixth International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'11), IEEE, 2011, pp. 1–10.
- 1070 [35] K. Ruan, J. Carthy, T. Kechadi, M. Crosbie, Cloud forensics, in: Advances in digital forensics VII, Springer, 2011, pp. 35–46.
- [36] M. Damshenas, A. Dehghantanha, R. Mahmoud, S. Bin Shamsuddin, Forensics investigation challenges in cloud computing environments, in: Proc. of International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec'12), IEEE, 2012, pp. 190–194.
- 1075 [37] F. Daryabar, A. Dehghantanha, N. I. Udzir, et al., A review on impacts of cloud computing on digital forensics, International Journal of Cyber-Security and Digital Forensics (IJCSDF) 2 (2) (2013) 77–94.
- [38] S. Simou, C. Kalloniatis, E. Kavakli, S. Gritzalis, Cloud forensics: identifying the major issues and challenges, in: Proc. of the 26th International Conference on Advanced Information Systems Engineering (CAiSE'14), Springer, 2014, pp. 271–284.
- 1080 [39] S. Mason, E. George, Digital evidence and cloudcomputing, Computer Law and Security Review 27 (5) (2011) 524–528.
- 1085 [40] R. Marty, Cloud application logging for forensics, in: Proc. of the 2011 ACM Symposium on Applied Computing (SAC'11), ACM, 2011, pp. 178–184.
- [41] J. Dykstra, A. T. Sherman, Design and implementation of frost: Digital forensic tools for the openstack cloud computing platform, Digital Investigation 10 (2013) S87–S95.
- 1090 [42] S. Zawoad, A. K. Dutta, R. Hasan, Seclaas: secure logging-as-a-service for cloud forensics, in: Proc. of the 8th ACM SIGSAC symposium on Information, computer and communications security (CCS'13), ACM, 2013, pp. 219–230.

- 1095 [43] T. Gebhardt, H. P. Reiser, Network forensics for cloud computing, in: Proc. of the 13th International IFIP Conference on Distributed Applications and Interoperable Systems (DAIS'13), Springer, 2013, pp. 29–42.
- [44] N. Thethi, A. Keane, Digital forensics investigations in the cloud, in: Proc. of the 2014 IEEE International Advance Computing Conference (IACC'14), IEEE, 2014, pp. 1475–1480.
- 1100 [45] K. Oestreicher, A forensically robust method for acquisition of icloud data, *Digital Investigation* 11 (2014) S106–S113.
- [46] B. Martini, K.-K. R. Choo, An integrated conceptual digital forensic framework for cloud computing, *Digital Investigation* 9 (2) (2012) 71–80.
- 1105 [47] B. Martini, K.-K. R. Choo, Cloud storage forensics: owncloud as a case study, *Digital Investigation* 10 (4) (2013) 287–299.
- [48] J. Dykstra, A. T. Sherman, Acquiring forensic evidence from infrastructure-as-a-service cloud computing: Exploring and evaluating tools, trust, and techniques, *Digital Investigation* 9 (1) (2012) S90–S98.
- 1110 [49] B. Martini, K.-K. R. Choo, Distributed filesystem forensics: Xtremfs as a case study, *Digital Investigation* 11 (4) (2014) 295–313.
- [50] M. Scanlon, J. Farina, N. A. L. Khac, T. Kechadi, Leveraging decentralization to extend the digital evidence acquisition window: Case study on bittorrent sync, arXiv preprint arXiv:1409.8486.
- 1115 [51] M. Scanlon, J. Farina, M. Kechadi, et al., Bittorrent sync: Network investigation methodology, in: Proc. of the Ninth International Conference on Availability, Reliability and Security (ARES'14), IEEE, 2014, pp. 21–29.
- [52] S. Wilkinson, *Acpo good practice guide for digital evidence* (2011).
- 1120 [53] N. I. of Standards, T. (NIST), U. S. of America, Forensic examination of digital evidence: A guide for law enforcement, <http://nij.gov/nij/pubs-sum/199408.htm> (Accessed 14 June 2014) (2004).
- [54] K. Kent, S. Chevalier, T. Grance, H. Dang, Guide to integrating forensic techniques into incident response, NIST Special Publication.

- 1125 [55] R. McKemmish, What is forensic computing? australian institute of criminology, <http://aic.gov.au/documents/9/C/A/%7B9CA41AE8-EADB-4BBF-9894-64E0DF87BDF7%7Dt1118.pdf> (Accessed 15 March 2015) (1999).
- [56] About url security zones, <https://msdn.microsoft.com/en-us/library/ms537183.aspx#internet> (Accessed 13 January 2015) (2015).
- 1130 [57] N. H. Ab Rahman, K.-K. R. Choo, A survey of information security incident handling in the cloud, *Computers and Security* 49 (2015) 45–69.
- [58] C. H. Malin, E. Casey, J. M. Aquilina, *Malware Forensics Field Guide for Linux Systems: Digital Forensics Field Guides*, Newnes, 2013.
- 1135 [59] R. Lee, Digital forensic sifting: Super timeline creation using log2timeline, <http://digital-forensics.sans.org/blog/2011/12/07/digital-forensic-sifting-super-timeline-analysis-and-creation>(Accessed 12 January 2015) (2011).
- [60] D. Quick, C. Tassone, K.-K. R. Choo, Forensic analysis of windows thumbcache files, Scholarly Paper, Rochester, NY, Social Science Research Network.
- 1140 [61] Nirsoft, Webbrowserpassview v1.58, [http://www.nirsoft.net/utils/web\\_browser\\_password.html](http://www.nirsoft.net/utils/web_browser_password.html) (Accessed 20 January 2015) (2015).
- [62] E. S. Canlar, M. Conti, B. Crispo, R. Di Pietro, Windows mobile livesd forensics, *Journal of Network and Computer Applications* 36 (2) (2013) 677–684.
- 1145 [63] Ram, virtual memory, pagefile, and memory management in windows, <http://support.microsoft.com/en-us/kb/2160852> (Accessed 12 February 2015) (2015).
- [64] Q. Do, B. Martini, J. Looi, Y. Wang, K.-K. Choo, Windows event forensic process, in: *Advances in Digital Forensics X*, Springer, 2014, pp. 87–100.
- 1150 [65] D. J. Farmer, A forensic analysis of windows registry, <http://forensicfocus.com/downloads/windows-registryquick-reference.pdf>(Accessed 12 January 2015) (2007).

- 1155 [66] H. S. Lallie, P. J. Briggs, Windows 7 registry forensic evidence created by three popular bittorrent clients, *Digital investigation* 7 (3) (2011) 127–134.
- [67] Nirsoft, Muicacheview v1.01, [http://www.nirsoft.net/utils/muicache\\_view.html](http://www.nirsoft.net/utils/muicache_view.html) (Accessed 25 November 2014) (2010).
- 1160 [68] Nirsoft, Userassistview v1.02, [http://www.nirsoft.net/utils/userassist\\_view.html](http://www.nirsoft.net/utils/userassist_view.html) (Accessed 20 January 2015) (2010).
- [69] J. Garcia, Typedurls, <http://forensicartifacts.com/2010/08/typedurls/>(Accessed 25 November 2014) (2012).
- [70] H. Carvey, *Windows Forensic Analysis Toolkit: Advanced Analysis Techniques for Windows 8*, Elsevier, 2014.
- 1165 [71] J. Garcia, Recentdocs, <http://forensicartifacts.com/2011/02/recentdocs/> (Accessed 25 November 2014) (2011).
- [72] File and directory linking, [https://msdn.microsoft.com/en-us/library/windows/desktop/aa364215\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa364215(v=vs.85).aspx) (Accessed 12 February 2015) (2015).
- 1170 [73] S. Technologies, Our products, <https://www.starfieldtech.com> (Accessed 25 February 2015) (2015).