Digital Investigation xxx (2013) 1–13

Contents lists available at ScienceDirect

Digital Investigation

journal homepage: www.elsevier.com/locate/diin

Cloud storage forensics: ownCloud as a case study

Ben Martini*, Kim-Kwang Raymond Choo

Information Assurance Research Group, School of Information Technology & Mathematical Sciences, University of South Australia, GPO Box 2471, Adelaide, SA 5001, Australia

ARTICLE INFO

Article history:
Received 6 May 2013
Received in revised form 22 August 2013
Accepted 24 August 2013

Keywords:
Digital forensics
Private cloud
SaaS
Storage as a service
StaaS
Cloud forensics
Cloud storage forensics
Open source cloud

ABSTRACT

The storage as a service (StaaS) cloud computing architecture is showing significant growth as users adopt the capability to store data in the cloud environment across a range of devices. Cloud (storage) forensics has recently emerged as a salient area of inquiry. Using a widely used open source cloud StaaS application – ownCloud – as a case study, we document a series of digital forensic experiments with the aim of providing forensic researchers and practitioners with an in-depth understanding of the artefacts required to undertake cloud storage forensics. Our experiments focus upon client and server artefacts, which are categories of potential evidential data specified before commencement of the experiments. A number of digital forensic artefacts are found as part of these experiments and are used to support the selection of artefact categories and provide a technical summary to practitioners of artefact types. Finally we provide some general guidelines for future forensic analysis on open source StaaS products and recommendations for future work.

1. Introduction

Cloud computing and in particular cloud enabled storage services have become an increasingly important part of the information technology industry in recent times. The number of cloud storage options is ever increasing with most vendors providing a variable amount of free storage before charging for higher storage tiers. Due to the large number of these services available many commentators have used the phrase Storage as a Service (StaaS) to describe this type of service (Kovar, 2009; Wipperfeld, 2009; Meky and Ali, 2011; Waters, 2011). This is an addition to the traditional cloud computing architectures documented by Mell and Grance (2009) of Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). Consumers have adopted the cloud storage paradigm in huge numbers with Gartner forecasting massive growth in the area stating that users will be storing a third of their data in the cloud by 2016 (Gartner, 2012). However many enterprises have remained cautious in moving their data into the public cloud storage environment due to issues such as data sovereignty and security, and complying with regulatory obligations. For example, enterprises who fail to comply with data protection legislation may lead to administrative, civil and criminal sanctions.

A number of open and closed source cloud software products have been developed and/or are in development to address the needs of enterprises and even individuals who want to leverage the features of cloud computing while continuing to store data on-site or otherwise under the control of the data custodian. Storing data on-site and/or having the data centres physically in the jurisdiction are increasingly seen as ways to reduce some of the location risks that cloud (storage) service clients currently face. For example, it has been suggested at one of the hearings of the Australian Government Parliamentary Joint Committee on Intelligence and Security that "the default position should be that governments, agencies and departments ought to keep their information onshore but use cloud for providers,

* Corresponding author.
E-mail addresses: ben.martini@unisa.edu.au (B. Martini), Raymond.Choo@unisa.edu.au (K.-K.R. Choo).

1742-2876/$ – see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.diin.2013.08.005

because there are great cost savings to government by using cloud, using digital storage and accessing the digital economy, being a model user of things like the NBN [Australian National Broadband Network], data centres and cloud computing. We think there is a real leadership role for government, but it needs to be done within something of a risk minimisation strategy, which means that you keep the data onshore and you do not look to send it offshore to a jurisdiction that you do not know about" (Australian Government Parliamentary Joint Committee on Intelligence and Security, 2012, p.16). More recently in 2013, the Australian Government has also released the National Cloud Computing Strategy (Australian Government Department of Broadband, Communications and the Digital Economy, 2013) and the policy and risk management guidelines for the storage and processing of Australian Government information in outsourced or offshore ICT arrangements (Australian Government Attorney-General's Department, 2013).

As with most new technologies cloud storage services have the capacity to be used for criminal exploitation and to form the basis of civil litigation. As such those working in both digital forensics and eDiscovery must have the capability to forensically analyse these storage platforms. Security and privacy issues associated with cloud services are generally better documented and understood than digital forensic issues. By physically displacing the storage from the user cloud storage solutions introduce numerous challenges for digital forensic and eDiscovery practitioners. For example, a 2012 report by the European Network and Information Security Agency (ENISA) explained that '[m]ulti-tenant outsourced services usually cannot give access to raw log data as it contains records of multiple users and thus would compromise the privacy of other customers' (ENISA, 2012, p.45) and, therefore, features synonymous with cloud (storage) services such as multi-tenancy, data security, file encryption and communications encryption also need to be addressed as part of a digital forensics investigation as suggested by various other researchers (Ruan et al., 2011; Chung et al., 2012).

1.1. Related work

Academic publications in the area of cloud forensics remain somewhat elusive. Many of the published papers in the area have provided a sound grounding for the research required in cloud forensics by highlighting the issues for digital forensic practitioners (Birk and Wegener, 2011; Martini and Choo, 2012). Papers with a specific technical perspective often focus on server forensic analysis providing recommendations for issues such as logging and remote extraction of data (Marty, 2011; Dykstra and Sherman, 2012). In recent months, a small number of papers discussing the forensic collection of cloud storage products have appeared, and their focus is on the client side digital forensic process assumedly due to the difficulties in obtaining access to a cloud providers datacentre to conduct server analysis (see Chung et al., 2012; Quick and Choo, 2013a, 2013b).

The small quantity of existing research demonstrates that research in the area of cloud based storage forensics is still in its infancy and there are a number of gaps in the existing research which need to be addressed. While addressing all of the research needs in this area is beyond the scope of a single paper, this paper seeks to address a number of gaps which have not been covered comprehensively (if at all) in existing literature.

1.2. Contributions

The key gap which we seek to contribute to with this research is an in-depth understanding of the artefacts available to forensics researchers and practitioners when conducting analysis on cloud StaaS environments on both the client and server. One of the key reasons this has not been explored in the existing literature has been the focus on public cloud StaaS services, while client analysis is possible, server platforms for public StaaS services are rarely available to researchers. In addition, access to the public cloud datacenters is generally not feasible for the purposes of digital forensic research for both security and privacy reasons. However there are private StaaS products available that a researcher can use as a case study to determine the forensic artefacts which potentially exist on all StaaS systems.

One cloud software product which provides features common in larger public StaaS products while being a freely available software package is ownCloud, which has seen rapid development and is now one of the major open source StaaS products. For example Google Trends shows a significant rise in popularity for the search term "owncloud" in the last two years (Google, 2013). In addition, a major tech website has recently promoted its potential use as a private cloud storage service (Klosowski, 2013) and Australia's Academic and Research Network (with over one million end users from 38 Australian universities, the Commonwealth Scientific and Industrial Research Organisation (CSIRO) and other academic, research and education institutions) is deploying ownCloud as the basis for its CloudStor+ service (AARNET 2013). These features make it a very appropriate case study software platform for this research into the forensic challenges of private StaaS. ownCloud (version 4 at time of research) provides a number of the features synonymous with cloud based storage solutions including a web based file access (view/manage/upload/download) and a "desktop sync" client for Windows, OS X and Linux which allows for automated synchronised copies of data on both the client and cloud server. Mobile clients are also available for Android and iOS devices. Other features particularly pertinent to digital forensics research include optional server-side file encryption and file versioning (storing multiple versions of a file).

We regard the contributions of this paper to be threefold:

1. Technical recommendations on forensic analysis of ownCloud StaaS instances will be the main focus in this paper.
2. Drawing from these technical findings, recommendations will be made on forensic analysis of open source StaaS products generally which will inform practitioners

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

of findings relevant to this general field and potential trends.
3. Finally this research seeks to validate our published cloud forensic framework (Martini and Choo, 2012).

1.3. Cloud forensics framework

Cloud computing introduces a number of complications to traditional digital forensic practices, as cloud servers are generally physically located in a different jurisdiction from that of the investigating law enforcement agency (LEA) and/or suspect. In addition, there are various methods of collection suited for the different cloud computing platforms and deployment models. For example IaaS may provide an export of the virtual hard disk and memory provided to the user while SaaS may only provide a binary export of the data stored on the hosted software environment. It is, therefore, important for the LEA collecting the evidence in one jurisdiction for use in a criminal prosecution taking place in another jurisdiction to work and cooperate closely with their foreign counterparts to ensure that the methods used in the collection are in full accordance with applicable laws, legal principles and rules of evidence of the jurisdiction in which the evidence is ultimately to be used (UNODC, 2012).

Unsurprisingly, a number of authors have suggested that a focus needs to be placed on cloud computing specific digital forensics guidelines (Birk and Wegener, 2011; National Institute of Standards and Technology, 2011; Zatyko and Bay, 2012). As such, it is appropriate that a cloud computing specific digital forensics framework be used to frame the discussion of these research experiments.

The cloud computing digital forensics framework used in this paper is based on our previously published work (Martini and Choo, 2012) and is discussed as follows (see Fig. 1).

The framework is based upon the stages outlined by McKemmish (1999) and NIST (Kent et al., 2006) but differs in a number of significant ways. One of the key features of this framework is its iterative nature which is essential in our research – the client is used both to identify the existence of cloud storage and to recover any data synced/cached on the client. As such forensic analysis of the client is carried out before analysis of the server environment.

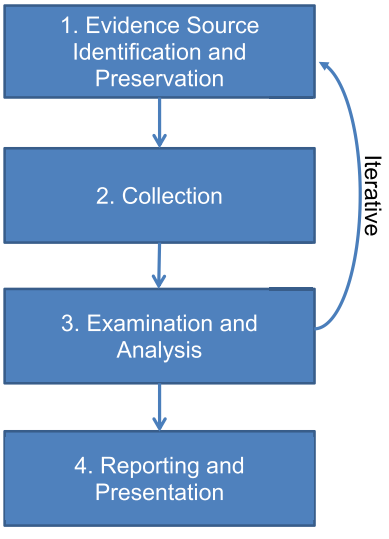

Fig. 1. Cloud forensics framework.

1. Evidence Source Identification and Preservation: This phase is concerned with identifying sources of evidence in a digital forensics investigation. During the first iteration, sources of evidence identified will likely be a physical device (e.g. desktop computers, laptops and mobile devices). During the second iteration, this phase is concerned with identifying cloud services/providers relevant to the case, possible evidence stored with the cloud provider and processes for preservation of this potential evidence. Regardless of the identified source of evidence, forensic investigators need to ensure the proper preservation of the evidence.
2. Collection: This phase is concerned with the actual capture of the data. Timely preservation of forensic data is critical and due to the complications of cloud computing data collection (e.g. significant potential for the cloud service to be hosted overseas and the potential for technical measures such as data striping to complicate collection), collection is separated from preservation. The collection may first involve the seizure of a client device, with the collection of the server data occurring on the second iteration.
3. Examination and Analysis: This phase is concerned with the examination and analysis of forensic data. It is during this phase that cloud computing usage would most likely be discovered based upon the examination and analysis of physical devices and this would lead to a second (or more) iteration(s) of the process.
4. Reporting and Presentation: This phase is concerned with legal presentation of the evidence collected. This phase remains very similar to the frameworks of McKemmish and NIST (as discussed in Martini and Choo, 2012). In general, the report should include information on all processes, the tools and applications used and any limitations to prevent false conclusions from being reached (see US NIJ, 2004).

This paper mainly focuses on the analysis stage of the framework with some discussion where pertinent of evidence source identification, preservation and collection.

1.4. Outline

In the next section, we provide an overview of the experiment environment and detail the experiments undertaken on both the client and server ownCloud instances. The findings in section 3 are framed around “artefacts”, which are defined at the beginning of each experiment. These artefacts are some of the common items of evidential value which a forensic practitioner may collect. Finally the last section concludes the paper with a summary of results, general conclusions on forensic investigation of open source cloud storage services and recommendations for future work.

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

2. Experiment setup

2.1. ownCloud overview

For the purposes of forensic analysis, the ownCloud software package can be separated into two related parts – the client software (including the sync clients and the web interface) and the server software running the cloud environment.

The ownCloud server software is primarily PHP code designed to be hosted on a web server. The software appears to be geared towards running on an Apache server on a *nix distribution but installations on other web servers and operating systems do exist. The server uses a database for metadata persistence and offers the administrator the option of using an SQLite database for smaller installations and MySQL for larger installations. By default, files stored in the ownCloud instance are stored relatively unmodified on the server operating system file-system in a subdirectory of the ownCloud application files. Advanced storage features associated with cloud storage such as file clustering for redundancy and scalability are not managed internally by ownCloud. These features would need to be implemented at the operating system level (using a product such as GlusterFS, XtreemFS or ZFS). The server software can be extended by installing/enabling "Apps" (both internal and third party), which can add features such as server-side encryption, integration with other cloud services/storage providers and additional authentication systems.

The ownCloud client software consists of both a web interface and several client applications. The web interface is standard for this type of cloud StaaS implementation. Beyond standard file upload, download, delete, rename, etc. capabilities, the default web interface also allows the user to play media files, view images in a gallery and maintain a contact list and calendar. These non-file related features are considered out of scope for this research on cloud based storage as a service and as such were not conclusively studied.

The desktop sync clients are available for many major operating systems as "binaries" and as source code for compiling manually. The ownCloud sync clients download page advises that "Linux, MacOSX and Windows are built with these sources" under the sources section (ownCloud, 2013), based upon this and our observations it is assumed that the core features of the sync client operate equivalently across all operating systems.

For this research the ownCloud server is hosted in a CentOS 6 environment with a default Apache, PHP and MySQL setup. The local ext4 file system on CentOS was used to store the uploaded files. The ownCloud desktop sync client was tested in a Windows 7 environment. For further information on the experiment environment see Tables 1 and 2.

2.2. Environment configuration

Tables 1–3 represent the environment specifications used in these experiments.

Virtualisation was used to implement both the client and server environments. This allowed for efficient data collection (both disk and network based) and in the case of the ownCloud server instance demonstrates a common configuration in many medium/large environments where ownCloud would be found on a virtualised platform.

Table 1
Server software specifications.

Operating System	CentOS 6.3
Web Server	Apache HTTP Server 2.2.15
Database Server	MySQL 5.1.61
ownCloud Server Application	Version 4.07

3. Findings

3.1. Client forensics

Client forensic analysis was conducted on a Windows client using the desktop sync client predominantly and three major web browsers (Microsoft Internet Explorer, Mozilla Firefox and Google Chrome) to access the cloud web interface.

Commonly in digital forensics research of this nature, "artefacts" are defined before commencement of the research that outline the types of evidence/data the practitioner is looking to recover/present, which can be used to link the suspect(s) to the device and/or cloud services used relevant to the commission of an alleged offence. In the case of private StaaS, we are seeking to recover the following artefacts of evidential value from the client:

- Sync and file management metadata – This includes logging, database and configuration data stored to facilitate the sync process between client and server. These artefacts can be useful in identifying the available evidence for collection from the server environment and used to build a file management history (e.g. sync/update times for individual files).
- Cached files – This artefact describes the files the user has stored on the client device and uploaded to the cloud environment or downloaded from the cloud environment to the client device. In cases where the cloud environment cannot be accessed, cached files may be relied upon as the only evidence available from the cloud environment.
- Cloud service and authentication data – Cloud service and authentication data is primarily used by the forensic practitioner to discover StaaS usage and potentially gives the practitioner the opportunity to connect to the cloud computing environment using the user's credentials if no other formal method is feasible. It will commonly consist of an address (DNS, IP, URL, etc.) that identifies which StaaS instance was used and potentially stored credentials (commonly a username and password) for the user.

Table 2
Client software specifications.

PC Operating System	Windows 7
ownCloud Sync Client	Version 1.05
Web Browsers	Internet Explorer 9, Firefox 15, Chrome 21
iOS Version	5.1.1
iOS ownCloud App Version	2.03

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

Table 3
Forensic tool specifications.

Guidance Software EnCase	Versions 6.19.6 and 7.04.01
Micro Systemation XRY	Version 6.3

- Encryption metadata – Client encryption metadata could include databases/configurations detailing which files are encrypted and using which algorithm, keys, etc.
- Browser artefacts – Browser artefacts can be critical data for a forensic practitioner (see Badger et al., 2012) both in terms of evidence source identification and examination and analysis, as (like cloud service data) it can often be used to identify which StaaS instance the user is communicating with and may also include file metadata often found in URLs.
- Mobile client artefacts – With the increasing prevalence and usage of mobile devices, mobile client artefacts may prove an invaluable evidence source for forensic practitioners (Tassone et al., 2013). The mobile clients may store any combination of the other artefacts discussed.
- Network analysis – Preliminary network analysis must be conducted to determine the feasibility of collecting StaaS data (with a focus on identification data) via network interception. This evidence source is beyond the scope of this paper.

3.1.1. (1) Evidence source identification and preservation and (2) collection

Identification, preservation and collection steps were not formally undertaken as part of this research as the client was setup in a controlled virtual machine environment. During a normal investigation, however, identification would commence with law enforcement identifying electronic devices (PCs, tablets, phones, etc.) that could be of evidential value and seizing these devices for preservation and collection (under a search warrant issued by the court). The devices would then be imaged using the appropriate forensically sound tool (see Table 3) depending on the device.

Virtual disk files (VMDK) were provided for examination and analysis as part of this experiment. In a typical law enforcement situation, these steps would likely be part of standard procedures for seizing a client device and the preservation/collection activity of image collection would result in an equivalent physical disk image file.

3.1.2. (3) Client examination and analysis

Examination and analysis commenced with examinations of the image file system to locate the artefacts listed above. The following describes the findings from this examination:

Sync and file management metadata – ownCloud client sync metadata information is predominantly stored in the "%localappdata%\ownCloud\folders" (e.g. C:\Users\[Username]\AppData\Local\ownCloud\folders) directory, located in which are a number of files (named for the sync directories they represent) that contain the configuration for each sync directory in the "ini" configuration file format.

In this experiment, a sync directory named "Pictures" was created. The configuration file for Pictures includes the following directives: "localPath" which describes the location on the client device where the synced data is stored (e.g. C:/Users/[Username]/Pictures), "targetPath" which describes the folder name on the cloud service (e.g. Pictures), and "backend" and "connection" directives appear to relate to the cloud connection used. Other sync directories configuration files listed the same configuration directives and similar configuration values.

File level metadata is stored on the client by the csync library (which forms part of the ownCloud client) in the form of SQLite databases located at "%userprofile%\.csync" named in the format of "csync_statedb_[HASH].db". It appears that one of these databases is created for each sync directory which is setup. In each database is a solitary table named "metadata" which contains an entry for each file in the directory and includes the following fields: "phash" which is a numerical hash of the filename (a cursory analysis of the relevant sync client source code indicates that this numerical hash is derived from "Jenkins Hash" – see Jenkins, 1997), "pathlen" which is the length of the filename string, "path" which is the filename string, and "modtime" which is a POSIX timestamp representing the last modified time of the file.

Cached files – The ownCloud client keeps copies of all files in synchronised directories on the local disk. The file metadata configurations/databases discussed above can be used to locate the files/directories synced to the local client. While the server supports storing multiple versions of files, these do not appear to be synced to the client.

Cloud service and authentication data – Located in "%localappdata%\ownCloud" is an "owncloud.cfg" client configuration file that contains valuable cloud identification and authentication data. The file is in the "ini" configuration file format and contains the following directives: "url" which lists the http or https URL for the ownCloud instance synced with this client, "user" which lists the username used to connect to the ownCloud instance in plaintext (if stored), "passwd" which lists the stored password for the ownCloud client stored using base64 encoding (if stored), and "nostoredpassword" which is a Boolean representation of the option to prompt for password at sync application launch. These details are critical to forensic practitioners as it allows them to identify that cloud computing StaaS has been used and the particular cloud computing provider/instance used. If law enforcement cannot be specific about where the data is (a specific device) and the information can be collected, search warrants may need to be created and executed in somewhat of an iterative fashion that supports an analytic discovery process. For example, an initial warrant that is minimally invasive (limited to externals only, no content) might be sought to use to identify which services the person of interest is using. Therefore, information obtained in our experiment allows them to contact the cloud provider/administrators and ensure that the cloud data is preserved, while applying for a more specific search warrant to obtain more content rich information.

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

The practitioner can also potentially use the username and password listed in the file to access the cloud server and access all the data available to the user to determine if there is any further evidence stored on the cloud if this is permissible. It should be noted that the capacity to legally execute this process is dependent on the statutory authority of the LEA in the jurisdiction where the client is located, a practitioner using this method should also be mindful of the processes required when handling live data especially to ensure no data is inadvertently overwritten. For example in Australia, Section 3L of the *Crimes Act 1914* (Cth) has a provision for the executing officer of a warrant to access data which includes data not held at the premises (i.e. accessible from a computer or data storage device). If assistance is required in operating the equipment and the executing officer believes on reasonable grounds that the material is liable to be destroyed, modified or tampered with, they may 'do whatever is necessary to secure the equipment, whether by locking it up, placing a guard or otherwise' (see Section 3L(4)) for a period of up to 24 hours or until expert assistance can be obtained, whichever happens first (see subsection 3L(6)). Section 3L(7) of the Act also allows the period to be extended, but the notice of these arrangements must be given to the occupier as detailed in Sections 3L(5) and 3L(8). Cloud service providers may also disclose the data if it is legally required and permissible to do so as part of a civil proceedings (e.g. in eDiscovery cases) – see Hooper et al. (2013).

Section 3LA of the *Crimes Act 1914* (Cth) and s201A of the *Customs Act 1901* (Cth) also make it an offence for persons with knowledge of computers or computer networks of which computers form a part, or measures applied to protect data held in, or accessible from, computers, to fail to provide any information or assistance that is reasonable and necessary to allow access to data held in, or accessible from, a computer that is on warrant premises, to copy the data to a data storage device, or to convert the data into documentary form. Failure to comply carries a maximum penalty of two years and six months imprisonment respectively. Such provisions are designed to overcome the efforts of accused persons to conceal data through the use of passwords or encryption, including in cloud storage services.

The storage of the user password in this manner raises two interesting points. Firstly due to the (small) size of the configuration file it will generally be stored as part of the MFT (Master File Table) on Windows clients (as it was in this case). This creates potential for alternative methods of recovery in the event that the file is securely deleted (such as external backups of the MFT data stored on the client PC). While configuration data such as that in owncloud.cfg is often stored in the registry on Windows clients, this is not always the case. The use of plaintext files such as owncloud.cfg may arise especially in cases where applications are written once and compiled for multiple operating systems. This may be a common trend in cloud computing clients which aim to be available across as many operating systems as possible.

Another point of interest with this configuration file is its inclusion in "System Restore" on Windows clients due to its ".cfg" extension (Microsoft, 2013) which can potentially allow a practitioner to recover the ownCloud instance, username and password used on the PC even after the ownCloud configuration files are securely deleted (overwritten). This was detected as part of our experiments after the owncloud.cfg file was securely deleted from the client device and a keyword search was able to reveal the base64 encoding of the users password stored on the disk. A keyword search for the phrase "passwd="@ByteArray" was able to locate two instances of the owncloud.cfg file stored in a file in the System Volume Information directory.

The usefulness of these techniques must not be understated as in the case of server-side encryption, a forensic practitioner may need to rely upon the password stored on the client to access encrypted data stored on the cloud server which may have already been securely deleted on the client device.

Encryption metadata – The ownCloud server supports encryption of user data however this encryption appears to be handled entirely on the server. The client does not internally support client-side encryption of user data and does not appear to be aware if encryption is enabled on the server. As such no notable encryption metadata is stored on the client in an ownCloud installation.

SSL encryption can optionally be used when communicating with the ownCloud server. However, this is dependent on the URL (specifically the use of the http or https scheme) used during the initial client setup. Further discussion on the network communications of the ownCloud client/server is included below.

Browser artefacts – Three of the major internet browsers (Microsoft Internet Explorer, Mozilla Firefox and Google Chrome) were used to access the ownCloud web interface and perform a number of common operations (e.g. download/upload a file, access calendar/contacts). An internet artefact search revealed artefacts relating to the ownCloud instance from all three browsers. History and downloads records revealed the files which were downloaded from the ownCloud instance and provided information on their original filename, path in the ownCloud profile and initial storage location on the client disk. For example one URL found in the Chrome downloads list "http://owncloud.local/owncloud/?app=files&getfile=ajax/download.php?files=Chrome.txt&dir=/BrowserUploads" indicates that a file was downloaded which was named Chrome.txt from the Browser Uploads directory. The "?app=" section of the URL changes to reflect the ownCloud "app" which is being accessed, "files", "calendar" and "contacts" were noted values. A request to index.php with the parameter "?logout=true" indicates that the user has gracefully logged out (pressed the logout button on the page). Page titles can be used to indicate the username of the ownCloud user as it is displayed in the format "[App] | ownCloud ([username])" for example "Files | ownCloud (johnsmith)". Many references were found to these page titles both within browser artefacts and throughout the unallocated clusters on the disk.

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

Recommended keyword search (GREP) expressions to locate ownCloud usage in browser artefacts include:

- \?app=(files)|(calendar)|(contacts)
- (files)|(calendar)|(contacts) \| ownCloud \(.*\)

Mobile client artefacts – ownCloud has mobile sync clients available for both the iOS and Android platforms. At the time of writing, development of these apps is still continuing. However, a forensic analysis of the current iOS app (version 2.03) was conducted for completeness using the Micro Systemation XRY Complete product. The iOS version of the ownCloud sync client used in these experiments only allowed for the upload of images and videos from the device, as such images and text files (created via the ownCloud web interface) were the primary test files for the mobile sync client experiments. The "App PIN" functionality of the ownCloud app (which permits the user to protect the app separately from the main device with a four digit PIN) was enabled.

A "physical" acquisition was undertaken as this allows the practitioner to view the mobile device data partition, extract the actual files stored on the device and potentially recover deleted files under some circumstances. The analysis revealed a number of files of forensic interest below its application root "Documents" directory. The iOS ownCloud client maintains a cache of accessed files in the "Documents/1" directory, the 1 appears to correlate with the "id" identifier in the users table of the DB.sqlite file. The DB.sqlite file contains a number of tables of forensic interest. The "users" table contains the URL of the ownCloud instance used with the client as well as the username and password of the user stored in plaintext. The "passcode" table contains the "App PIN" used to secure the application (if set). The "files" table contains a list of file paths (relative to the server URI using WebDAV access), file names, size and the full local path to the file (if cached otherwise <null>) as well as a number of other values. The "files_backup" table remained empty in our experiment.

This shows that the mobile client can be of significant value to a forensic practitioner as it not only provides the server details and credentials for the user (potentially allowing the practitioner to connect to the ownCloud instance to collect evidence) but also a cache of files accessed and a list of files stored on the ownCloud instance (for the particular user) at the time of last sync.

Network analysis – Basic analysis of the network communication between the ownCloud client and server was undertaken using packet captures. HTTP traffic was monitored between the client and server to determine that the ownCloud sync client is using the WebDAV protocol to handle file transfers. The ownCloud iOS client appears to use similar WebDAV requests. If plain HTTP (as opposed to HTTPS) is used to establish the connection with the ownCloud server, the content and commands sent and received from the server to the client is readable in plaintext as part of normal HTTP and WebDAV requests (e.g. HTTP PUT requests revealed data). Standard HTTP Basic authorization (comprising the username and password supplied by the user for the ownCloud instance base64 encoded) is sent with each WebDAV request.

3.1.3. (4) Reporting and presentation

Reporting on many of these client artefacts is currently a manual process and detection or heuristics of cloud computing use is not integrated into the major digital forensics analysis products used. While browser artefacts from major browsers were extracted via standard evidence preparation scripts the other artefacts above were located manually. If identification of cloud computing usage were to become standard practice for digital forensic investigations this would be a very time consuming process for a forensic practitioner.

3.2. Server forensics

Server forensic analysis was conducted on a CentOS 6 virtual machine hosting the ownCloud PHP software via an Apache HTTP server and using MySQL as the database backend (see Table 1). For the purposes of this experiment CentOS, Apache and MySQL were configured using setup defaults where possible or logical selections.

Before commencement of the server forensic analysis, we seek to define the "artefacts" which we hypothesise exist. In the case of StaaS, we are seeking the following artefacts of evidential value from the server:

- Administrative and file management metadata – Administrative data which stores the configuration of the cloud instance and of individual users within the cloud instance as well as database and configuration files which list the files and data stored by the user on the cloud instance.
- Stored files – The data uploaded by the user to the cloud instance.
- Encryption metadata – Data relating to encryption (if enabled) in the cloud instance, specifically data relating to decryption of user data.
- Cloud logging and authentication data – Logging and authentication data associated with transactions made by the user with the cloud instance (files uploaded/downloaded, login events, etc.). As emphasized in the ENISA (2012, p.45) report, '[m]onitoring of log availability is crucial to trace back events and allocate liabilities and responsibilities. The more sensitive the information involved, the more monitoring of log availability is crucial. Log data is also important for incident response so business continuity requirements should be taken into account when reviewing this parameter. Finally, log data is often needed to satisfy corporate data governance and compliance requirements – e.g. Data Protection Law and SOX-like laws in Europe'.

3.2.1. Evidence source identification and preservation

As outlined in the framework, early identification and preservation of cloud computing use is critical for successful forensic analysis. In the case of ownCloud servers, the most straightforward method of identification is via the installation of the ownCloud client or web browser logs located on the client device of the suspect.

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

Section 3.1.2 outlines methods of identification of the ownCloud server on the client device which may include searching for the configuration files of the ownCloud client or analysis of web browser history logs. Once the ownCloud server instance has been identified, it is expected that standard processes for locating web servers (e.g. via owner of IP address) will allow the forensic practitioner to physically locate the server and server owners/administrators.

For the purposes of this research, it is assumed that the physical server that is hosting the ownCloud instance is located within the jurisdiction of the forensic practitioner. If the server is located out of jurisdiction the practitioner may need to rely upon a "mutual assistance request" (e.g. *Mutual Assistance in Criminal Matters Act 1987* (Cth) in Australia) to request the assistance of a law enforcement agency within jurisdiction of the physical server or rely solely upon any evidence attained as part of the client analysis (which in many cases may still be substantial). There are, however, various challenges in relying on mutual legal assistance arrangements to obtain evidential data from overseas cloud service providers in the current environment (e.g. procedural and technical challenges). For example, Cuthbertson (2012, pp.128–129) explained that "the types of assistance provided for under the mutual assistance regime generally involve the exercise of coercive powers … [and at] the international level, it has been generally accepted that such measures raise issues of state sovereignty and should only be made available pursuant to formal government-to-government requests". Similar observations were also echoed in the 2012 UNODC report: "[c]ases requiring the investigation or prosecution of cross-border activities of terrorists or other criminals might have sovereignty implications for those countries in which investigations need to be undertaken. It is therefore important, when considering investigative actions involving the collection of evidence related to computers or the Internet, for investigators and prosecutors to be mindful of the potential implications such investigative actions might have for the sovereignty of other States (e.g. authorities in one country remotely searching the computer being operated by a suspect located in another country)" (UNODC, 2012, p. 92). Therefore, it is important for (forensic) "investigators [to] have sufficient understanding of the legal rules/principles applicable to investigative actions they are undertaking as part of an investigation and/or to communicate closely with prosecutors, by both updating them and seeking legal advice" and to work closely with their foreign law enforcement agencies to obtain key evidence for legal proceedings (UNODC, 2012, p. 94).

In terrorism or counterintelligence investigations, there are generally special access mechanisms to request for data from cloud service providers (compelled disclosure). For example in Australia, Section 25A of the *Australian Security Intelligence Organisation Act 1979* (Cth) allows a government minister to issue a computer access warrant if the minister is satisfied that there are reasonable grounds for believing that access by the Organisation to data held in a particular computer (the target computer) will substantially assist the collection of intelligence in accordance with this Act in respect of a matter (the security matter) that is important in relation to security. The warrant may also allow the Australian Security Intelligence Organisation (ASIO) to do anything reasonably necessary to conceal the fact that anything has been done under the warrant although the addition, deletion or alteration of data, or the doing of anything, that interferes with, interrupts or obstructs the lawful use of the target computer by other persons, or that causes any loss or damage to other persons lawfully using the target computer is explicitly prohibited (subsection25A(5)).

When the physical server hosting the ownCloud instance has been located, preservation must commence immediately. ownCloud does not have built in preservation features for the purposes of law enforcement. While a "litigation hold" or similar freezing feature would be useful (especially at a per user granularity level), forensic practitioners will need to use different preservation methods depending on the type of evidence to be preserved and the size of the ownCloud instance and supporting infrastructure. If possible, disconnecting the ownCloud instance from the network while preparing for collection of data can help in preventing the suspect from deleting data via standard access methods (sync client, browser, etc.) while collection is being performed.

If the ownCloud instance is small (as in the case of this research), then taking a bit-stream image of the entire instance is preferable wherever possible (depending on the underlying infrastructure) as this will preserve and collect both the users current data, system wide data (such as logs) and potentially deleted data which remains on the disk(s). However it is acknowledged that for a number of reasons (e.g. size, time, customer privacy), it will often not be possible to take a complete bit-stream image of the cloud instance. In these circumstances, it is advisable to commence collection of the data stored on the ownCloud instance specifically relevant to the suspect (e.g. data directories, logs, encryption keys/authentication data – discussed further as part of collection) as quickly as possible while keeping the instance disconnected from the network.

3.2.2. Collection

There are a number of different collection methods available to a forensic practitioner when collecting evidence from an ownCloud instance. The use of these methods depends on the individual attributes and circumstances of the investigation as well as the resources of the forensic practitioner. Regardless of collection method to ensure the maximum possible useable evidence is collected and preserved, the following list of ownCloud server artefacts are recommended for collection:

1. **Data uploaded by the suspect** – These artefacts include the main source of evidence in an ownCloud installation. User files uploaded to the ownCloud instance are stored in a directory accessible to the web server defined as part of the initial setup of the application.

This data can be located via the web server hosting the ownCloud instance. The ownCloud configuration file ([owncloud-web-root]/config/config.php) will indicate the location of the ownCloud data directory using the

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

"datadirectory" configuration directive. On a live system the practitioner can use this information to determine if the data is stored on the local device which is hosting the front end software or if the data is being stored on a mounted external or network based storage device as would be common in a cloud computing environment.

Once the physical location of the data has been determined, the practitioner can make a decision as to the feasibility of taking a bit-stream image of the physical media source or taking a logical copy of the visible data structure (which may be the only practical option if the physical media is too large or complex to acquire in a timely manner).

The structure of the "datadirectory" varies somewhat depending on the configuration of the ownCloud instance and is discussed further below in step 3 however the users data uploads should be located in the files subdirectory of the users directory e.g. "[datadirectory]/[username]/files". It is recommended that the "[datadirectory]/[username]" directory be copied in full.

The practitioner must also give consideration to assuring the provenance of the data collected using this method. One of the reviewers pointed out that, for example, due to the highly shareable nature of client devices as well as usernames and passwords; it may not be possible to determine the person responsible for upload or download of the data. A child exploitation materials case in Australia highlighted the challenge of accurately determining the person responsible for upload or download of the data in question,

> *'the affidavit of the father filed by leave in court on 22 July … alleged that if the annexures to the mother's affidavit are correct somebody logged in on the laptop at 6.50am on the morning of Monday 14 January using the password "[B] Rd" and remained logged in continuously until at least 7.28pm and also signed in to a hotmail account at 11.16am. His evidence is that he did three separate jobs on behalf of [S] Pty Ltd on that day … [and] there is no actual evidence before the court as to the nature of the items alleged to have been viewed by the father — only conclusions drawn by others that the dumps included child pornography; there is no evidence of a chain of custody of the computer; the contents of the lap top have been wiped; and the evidence as to who had access to the computer when the parties were together is disputed' (Waters v Waters and Anor [2009] FMCAFAM 819).*

Therefore, as is the case with other online services such as email and social networking, the practitioner may need to rely upon other forensic methods (e.g. internet history records from PCs and ISPs, document metadata, source copies on physical devices under the suspects control) to determine provenance of the data collected from the cloud server.

2. **Data generated by the ownCloud instance** – Data generated by a cloud instance relating to a particular user may be very useful in linking a user with the data located in the cloud environment (e.g. log data) or a necessary part in extracting evidence relating to a particular user from the cloud environment (e.g. encryption keys). As such it is recommended that this data also be collected.

In the case of ownCloud, the web server hosting the front end of the ownCloud instance typically generates log data. Apache was used as the web server for this experiment and as such, logging data could be found in "/var/log/httpd" where the "access_log" and "ssl_access_log" are of particular interest to forensic practitioners as discussed further in section 3.2.3.

ownCloud has the optional capability to enable encryption on the files uploaded to the ownCloud instance. The encryption key is stored with the users data in their "datadirectory" and, as noted, this should be collected as part of the acquisition of the users data.

ownCloud stores metadata in an SQLite or MySQL database. The database should be collected for examination and analysis, in the case of MySQL the "mysqldump" utility can be used (the database name and credentials for accessing the database can be located in the ownCloud configuration file "config/config.php") or the database files can be copied directly from "/var/lib/mysql/[database name]" (in our experiment environment). In the case of SQLite, the database is stored in the "[datadirectory]/owncloud.db" file which should be collected.

It is also recommended that practitioners make a copy of the ownCloud application files as there may be version specific issues which need to be overcome before evidence can be extracted. The application files (in combination with the other data collected) will allow the practitioner to setup a duplicate instance of the ownCloud environment for testing. Locating the application files can be achieved via a variety of methods, searching the default web server root path ("/var/www/html" in our experiment), searching the web servers configuration file to determine the path being used for the ownCloud instance or performing a keyword search on the server image for common ownCloud application files/configuration directives (e.g. owncloud, owncloud.db, ocsclient.php, files_encryption).

3.2.3. Server examination and analysis

Examination and analysis commenced with examination of the image file system to access the artefacts listed above. The following describes the findings from this examination:

Administrative and file management metadata – ownCloud stores the majority of the file management metadata on the server in the SQL database which would have been collected as part of a typical digital forensic process. Tables prefixed with oc_calendar, oc_contacts and oc_media have not been included for analysis as they are not within the focus of this paper. However, they are expected to contain the data stored relevant to those applications (media, calendar and contacts).

Tables of StaaS forensic interest are discussed below.

oc_users: oc_users lists the usernames and a hashed password for the users of the ownCloud instance. The passwords appear to be hashed using a Blowfish based hashing algorithm using the "portable PHP password hashing framework" (phpass) – http://www.openwall.com/phpass/.

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

oc_sharing: oc_sharing lists files (owner and path) which have been shared with other users or "public" in the case of files shared using the "Share with private link" functionality. A "target" UID is listed which matches the "token" parameter in the shared URL. In the case of files shared between users/groups on the same ownCloud instance, the table has the following fields of interest:

- "uid_owner" lists the username which is sharing the file.
- "uid_shared_with" lists the username of the user who has been granted access to the file.
- "source" lists the file system path to the shared file (relative to the ownCloud "datadirectory").
- "target" lists the path to the shared file as it appears to the shared_with user.
- "permissions" is set to 0 for read-only access and 1 for writable access.

Where a target is shared with a group, it appears that one row is created for each user of the group with the "uid_shared_with" format of "[username]@[groupname]".

oc_pictures_images_cache: oc_pictures_images_cache lists images stored for each ownCloud user (presumably generated for/by the "Pictures" or "gallary" app). Path, width and height (as well as uid_owner – the username of the image owner) are stored. Notably entries are not immediately deleted when a user deletes an image.

oc_log: The oc_log table has remained empty during our experiments, and it is assumed that this table is to be used with a future feature. The table contains the following fields: id, moment (datetime), appid, user, action and info. If this level of logging is enabled by default in future versions of ownCloud, this table could prove to be a very useful forensic artefact.

oc_appconfig: This table lists a number of configuration parameters for the ownCloud internal applications. One of particular interest is the "appid" of "files_encryption" and specifically the "configkey" of "enabled" which with a "configvalue" of "yes" (in a "files_encryption" row) denotes that encryption has been enabled in this ownCloud instance.

oc_fscache: As the name suggests, this table appears to be a cache of the file system objects (data uploaded) stored within the ownCloud data directory. The fields in this table are as follows: id, path (relative to the ownCloud data directory), user, size (bytes), ctime (create time in POSIX format), mtime (modified time in POSIX format), mimetype and mimepart (representing the mimetype of the file), encrypted (1 if the file is encrypted, 0 otherwise), versioned (1 if the file has previous versions, 0 otherwise) and writable (1 if the file is writable, 0 otherwise). If files are deleted, entries in this table appear to be deleted immediately.

The "encrypted" value is of particular interest to a practitioner as not all files appear to be encrypted even if encryption is enabled in an ownCloud instance (jpg files being a notable example in our experiments) and a practitioner will need to know which files are encrypted before attempting to manually decrypt the files. Equally, fields such as "ctime", "mtime" and "size" may be of importance to a practitioner in terms of reporting and presentation on the evidence sourced from the ownCloud instance.

Stored files – As part of the collection phase of the framework (see Martini and Choo, 2012), stored files in an ownCloud instance are located in the "datadirectory" which should have been located and collected as part of that phase. The "datadirectory" contains a subdirectory for each user in the ownCloud instance that in turn contains a "files" subdirectory and a "versions" subdirectory (if file versioning is enabled and available).

The structure of the "files" directory is as setup by the user (it appears as the root directory to the user), which contains files and directories as created/synced by the user. The view of these files/directories to the forensic practitioner is largely identical to that of the user. If encryption is not enabled, a practitioner should be able to extract files of interest from this directory and sub-directories directly.

The structure of the "versions" directory appears to mirror the "files" directory. However, this directory structure is not directly accessible to the user. This directory is used to store previous versions of files stored by the user in the ownCloud instance. Users can use the "History" button in the ownCloud web interface (accessed per file) to view previous versions stored. File names of files stored in the "versions" directory are appended with a POSIX timestamp that represents the time when the version of the file was copied to the "versions" directory.

It appears that previous versions of files are not immediately deleted when they are deleted by the user from their mirror location(s) in the "files" directory. ownCloud describes "Version Control" on its website (http://owncloud.org/support/version-control/) and notes that "[c]hanges made at intervals greater than two minutes are saved in data/[user]/versions".

Encryption metadata – If encryption has been enabled on the ownCloud instance (which can be determined using the "oc_appconfig" table as discussed above), each user (with files stored on the instance) should have an encryption.key file in the root of their individual data directory (e.g. data/username/encryption.key). Each file uploaded which is not exempted from encryption (see list of exempted file types below) should be encrypted using this key. ownCloud uses a type of Blowfish encryption (provided by the third party "Crypt_Blowfish" PEAR package – http://pear.php.net/package/Crypt_Blowfish) to encrypt individual files. The process of encryption is as follows: The "encryption.key" file is created (if it does not already exist) and consists of 20 random digits encrypted with the users password. These 20 random digits are the "key" used to encrypt the users files using the same Blowfish encryption method. This is a preferred method of encryption as it allows a user to change their password without needing to decrypt the files with their old password and re-encrypt the files with the new password, in this case the system only needs to decrypt and re-encrypt the encryption.key file with the users updated password.

The following file types are exempt from encryption by default: jpg, png, jpeg, avi, mpg, mpeg, mkv, mp3, oga, ogv and ogg. This is configurable (globally to the ownCloud instance) from the admin ownCloud settings page.

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

Using artefacts collected from the client(s) (and potentially the server) users files can be decrypted, three items are required, the users password (in any of a number of formats), the encryption.key file and the file(s) which are to be decrypted. The users password can be collected using a number of methods including directly from the user, from a saved version in the sync client configuration and mobile clients database (as discussed above), if using the browser client potentially in the saved passwords lists stored by the user and captured from WebDAV network traffic if the user is not connected via HTTPS.

The process for decrypting files is as follows: Use the contents of the encryption.key file as the data and the users password as the key in the Crypt_Blowfish Decrypt method which will return the 20 digit "encryption key". Then use the "encryption key" as the key and the encrypted file contents as the data in the Crypt_Blowfish Decrypt method which will return the decrypted file. This method was tested successfully during our experiments.

Junod (2012) discussed the ownCloud encryption module and lists a number of flaws with this feature. One of these flaws leads to another variation of this decryption procedure which uses the plaintext "encryption key" stored on the server as part of the users PHP session file. In our experiments, these files were stored on the server in "/var/lib/php/session". Using the PHPSESSID cookie stored on the client for the ownCloud domain, we were able to locate the PHP session file (stored in the format sess_[PHPSESSID]) which remained on the server after we completed collection of the server image. It should also be possible to keyword search the files in the "session" directory searching for the username of the user and string "enckey" in the file to locate the appropriate "sess_[PHP_SESSID]" file. Using the 20 digit "encryption key" stored in this file, encrypted files can be decrypted directly using Crypt_Blowfish (as discussed above) without needing the user's password. This method of decryption was also tested successfully as part of our experiments.

Cloud logging and authentication data – By default, ownCloud does not generate internal logging data which appears to be of use to a forensic practitioner. The data which is logged by default by the ownCloud application relates mostly to internal operational issues such as "file not found in cache" errors. A logging level option is available in the admin settings section of ownCloud. Surprisingly, when debug level logging is enabled (which generates a log file in the "datadirectory" named "owncloud.log"), no entries of general interest were found after simple testing (login, logout, file upload, etc.).

However in our experiments, the web server (Apache) did produce a number of useful log entries in its "access.log" and "ssl_access.log" files located in the "/var/log/httpd directory" relating to the ownCloud instance. The "access.log" file contains an entry for each request served by the web server and the "ssl_access.log" file contains similar entries for requests which were made via HTTPS connections. These requests commonly include file "GET" requests for page content in the web client and file downloads, "POST" requests for form data/AJAX file uploads transferred to the server and a number of other WebDAV specific request types commonly used by the sync and mobile clients. Although the ownCloud specific GET and POST requests do record date/time and IP addresses by default (as do all request entries in the log file), usernames are not recorded against these requests as Basic HTTP authentication is not used with web client requests. However, usernames are recorded against requests that are made via WebDAV (e.g. sync clients) as HTTP Basic authentication is used as part of these requests. File names, paths, etc. are also common in many requests. This allows a practitioner to determine in many instances the times and dates when individual files are uploaded, downloaded and modified by particular users which is of interest.

3.3. Summary of findings

Tables 4 and 5 summarise the key artefacts located as part of the client and server experiments and note the relevance of individual artefacts to a digital forensics investigation.

4. Conclusion and future work

This research demonstrated that cloud StaaS provides a significant number of useful artefacts for forensic practitioners in an investigation. Using ownCloud as a case study, we successfully undertook a forensic examination of the client and server components of an ownCloud installation and discussed the relevance of a number of artefacts to a forensic investigation (see Tables 4 and 5 in Section 3 for a summary). To the best of our knowledge, this is the first paper that provides a holistic discussion on cloud StaaS forensics from both client and server perspectives – previous papers have focused on either the server (see Marty, 2011; Dykstra and Sherman, 2012) or the client devices (see Chung et al., 2012; Quick and Choo, 2013a, 2013b).

Our analysis of the client devices demonstrated that in many cases significant data can be found which links the user to a particular ownCloud instance. This provides a forensic trail to the ownCloud server instance even when evidential data on the client may be securely deleted. The client artefacts found in the ownCloud experiments are likely to be common with other open source cloud storage products developed in the future as cloud products mature and develop a common feature set. While individual implementations may vary, practitioners can use the artefacts discovered in these experiments as a basis for their investigation of the client as a potential evidence source and perhaps more importantly as a link to the cloud computing instance on which other data may be stored. The file metadata and cloud authentication artefacts found are of particular interest in an investigation which heavily involves cloud computing use. These artefacts can be used not only to determine the cloud computing instance used but also provide authentication data potentially allowing an investigator to collect data from the cloud instance directly and help link user actions with the data stored in the cloud computing environment via the use of file metadata such as permissions and timestamps.

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

Table 4
Client artefact summary.

Category	Artefact	Relevance
Sync and file management metadata	ownCloud "folders"	Assists the practitioner in determining which client folders are synced with an ownCloud instance.
	File metadata	May assist a practitioner in determining files stored within a synced directory and file modification times.
Cached files	Synced files	Files synced to the client from the ownCloud instance appear as regular files. They can be located using the sync and file management metadata.
Cloud service and authentication data	owncloud.cfg	The owncloud.cfg file is one of the key ownCloud artefacts on the client. It allows the forensic practitioner to determine the ownCloud instance which is being used with the sync client and allows the practitioner to collect the users credentials (if stored). If the file has been deleted, a number of avenues for recovery are available including a keyword search of unallocated space, MFT backups and system restore.
Browser artefacts	URL parameters	When using the ownCloud web client, URL parameters (in history, bookmarks, download lists, etc.) can provide a practitioner with a broad range of information (potentially including date and time) on the ownCloud "app" being used, server file names and directories for files downloaded and logoff events.
	Page titles	A keyword search for ownCloud page titles (e.g. "Files | ownCloud (username)") is a key identifier of ownCloud use and may assist a practitioner in determining the ownCloud instance used and ownCloud username if the web client has been used.
Mobile client artefacts	Accessed files	Files which have been accessed on the iOS client appear to be cached locally, and this may allow a practitioner to access files not available on other devices.
	DB.sqlite	The SQLite database used by the ownCloud client stores valuable authentication data and file metadata that relates both to files stored on the device and the server. This may assist a practitioner in gaining access to the ownCloud instance used or in contributing as evidence of files stored and file times.
Network analysis	HTTP/WebDAV artefacts	It was noted that the ownCloud client uses WebDAV over HTTP or HTTPS to facilitate file synchronisation between the server and the client. When a non-SSL HTTP connection is setup to communicate between client and server data can be recovered from network captures. HTTP Basic authentication can also be captured in this setup which is another method by which a practitioner can collect a users ownCloud credentials.

Our server analysis showed that while collection of data in an environment with one server such as the instance in these experiments may be relatively straightforward, factors such as encryption could complicate investigations significantly. While many practitioners may focus upon collection of the files uploaded by the suspect in the first instance it has been demonstrated that it is important to collect the range of artefacts suggested as they may be required to assist in linking a user with the data stored in the cloud instance, recovering previous data stored by the user in the cloud instance or in decrypting data stored by the user. In many cases, it will not be possible to collect the entire cloud storage instance due to the size and amount of unrelated (other users) data stored on the physical device(s). Consequently, this makes collection of the full range of artefacts critical as once the preservation methods are no longer being applied to the cloud instance, critical data such as encryption keys and metadata may be lost.

The utility of our iterative cloud forensics framework was demonstrated with client artefacts being used to identify cloud storage usage and being used to decrypt files stored on the server. The iterative nature of the framework suggests that client devices are analysed first to both identify cloud usage and allow practitioners to request preservation by the cloud computing provider in a directed manner providing as much information on the data requested to be preserved as possible. Analysis of sync and file management metadata on the client can also help prevent time being spent on investigation of cloud services which are unlikely to be of evidential value.

Table 5
Server artefact summary.

Category	Artefact	Relevance
Administrative and file management metadata	SQL database	The SQL database on the ownCloud server stores a range of data which could be of use to a forensic practitioner. This includes a user list, sharing permissions, encryption configuration and a cache of file system information such as file paths, owner, size, modified types, encryption status, etc.
Stored files	"datadirectory"	The "datadirectory" contains the structure and files uploaded by the user to the ownCloud instance. This is a primary source of evidence for a forensic practitioner.
	File versioning	Within the "datadirectory" is a versions directory which contains past versions of files and potentially deleted files.
Encryption metadata	Blowfish encryption	Encryption can be optionally enabled on an ownCloud instance, when enabled most files uploaded are encrypted (some file types are exempt by default). The encryption key is stored in the "encryption.key" file stored in the users "datadirectory" subfolder and encrypted with the users password. A practitioner can collect the users password from a number of other artefacts and decrypt the files stored.
Cloud logging and authentication data	Web server logging data	The default logging data stored by the web server (Apache in these experiments) can be of use to a forensic practitioner to determine when a user has communicated with the ownCloud server and the changes made by the user as part of that session. The usefulness of this data was limited when the web client was used. However a large amount of information was available on sync client transactions.

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005

While it may be possible to preserve an ownCloud instance by disconnecting the environment from the network, this approach is not guaranteed to ensure preservation and will result in potentially significant downtime for all users of the cloud instance. It is instead recommended that StaaS developers integrate preservation technologies directly into the product. In this case, ownCloud could "freeze" a user's account preventing them from making any further changes (after a valid request is received from law enforcement) and provide a forensic practitioner with a package containing the contents of the users files directory, previous versions, encryption key and any relevant metadata and logging information. The provision of this package would not only simplify the extraction of evidence for the practitioner but also ensure minimal downtime for the cloud instance (in this case none should be required).

It is recommended that future work be continued in this area looking at the other cloud StaaS products available including those hosted in the public cloud environment to determine the best practices for forensic extraction and analysis on these platforms. Further work on the potential for network interception as a method of forensic collection should be pursued especially as a method of identification of potential evidence sources.

Acknowledgements

The first author is supported by scholarships from both the University of South Australia and the Defence Systems Innovation Centre. The views and opinions expressed in this article are those of the authors alone and not the organisations with whom the authors are or have been associated/supported. We thank the editor and the anonymous reviewers for providing constructive and generous feedback. Despite their invaluable assistance, any errors remaining in this paper are solely attributed to the author.

References

AARNET. CloudStor+ viewed 10 April 2013, <http://www.aarnet.edu.au/communities/research/eresearch-overview/cloudstorplus-information>; 2013.

Australian Government Attorney-General's Department. Australian Government Policy and Risk management guidelines for the storage and processing of Australian Government information in outsourced or offshore ICT arrangements. In: <http://www.protectivesecurity.gov.au/informationsecurity/Documents/PolicyandRiskmanagementguidelinesforthestorageandprocessingofAusGovinfoinoutsourcedoroffshoreICTarrangements.pdf>; 2013.

Australian Government Department of Broadband, Communications and the Digital Economy. The national cloud computing strategy. <http://www.dbcde.gov.au/__data/assets/pdf_file/0008/163844/2013-292_National_Cloud_Computing_Strategy_Accessible_FA.pdf>; 2013.

Australian Government Parliamentary Joint Committee on Intelligence and Security. Proof Committee Hansard (Public) Wednesday, 5 September 2012. <http://www.aph.gov.au/Parliamentary_Business/Committees/House_of_Representatives_Committees?url=pjcis/nsl2012/hearings/program01.pdf>; 2012.

Badger L, Grance T, Patt-Corner R, Voas J. Cloud computing synopsis and recommendations. Gaithersburg, MD: SP 800-146, U.S. Department of Commerce; 2012.

Birk D, Wegener C. Technical issues of forensic investigations in cloud computing environments. In: 6th international workshop on systematic approaches to digital forensic engineering – IEEE/SADFE 2011, Oakland, CA, USA 2011. p. 1–10.

Chung H, Park J, Lee S, Kang C. Digital forensic investigation of cloud storage services. Digital Investigation 2012;9(2):81–95.

Cuthbertson S. Mutual assistance in criminal matters: cyberworld realities. In: Hufnagel S, Harfield C, Bronitt S, editors. Cross-border law enforcement regional law enforcement cooperation – European, Australian and Asia-Pacific Perspective, Routledge Research in Transnational Crime and Criminal Law 2012. p. 127–42.

Dykstra J, Sherman AT. Acquiring forensic evidence from infrastructure-as-a-service cloud computing: exploring and evaluating tools, trust, and techniques. Digital Investigation 2012;9(1):S90–8.

European Network and Information Security Agency (ENISA). Procure secure: a guide to monitoring of security service levels in cloud contracts. Heraklion, Greece: European Network and Information Security Agency; 2012.

Gartner. Gartner says that consumers will store more than a third of their digital content in the cloud by 2016 viewed 15 October 2012, <http://www.gartner.com/it/page.jsp?id=2060215>; 2012.

Google. Google trends – web search interest: owncloud – worldwide, 2004–present viewed 12 July 2013, <http://www.google.com.au/trends/explore#q=owncloud&cmpt=q>; 2013.

Hooper C, Martini B, Choo K-KR. Cloud computing and its implications for cybercrime investigations in Australia. Computer Law & Security Review 2013;29(2):152–63.

Jenkins RJ. Hash functions for hash table lookup viewed 22 August 2013, <http://www.burtleburtle.net/bob/hash/evahash.html>; 1997.

Junod P. OwnCloud 4.0 and encryption viewed 24 September 2012, <http://crypto.junod.info/2012/05/24/owncloud-4-0-and-encryption/>; 2012.

Kent K, Chevalier S, Grance T, Dang H. Guide to integrating forensic techniques into incident response. Gaithersburg: SP 800-86, U.S. Department of Commerce; 2006.

Klosowski T. How to set up your own private cloud storage service with owncloud. Lifehacker Australia, viewed 5 April 2013, <http://www.lifehacker.com.au/2013/04/how-to-set-up-your-own-private-cloud-storage-service-with-owncloud/>; 2013.

Kovar J. Storage-as-a-service poised for big boost. CRN 2009;1291:14–8.

Martini B, Choo K-KR. An integrated conceptual digital forensic framework for cloud computing. Digital Investigation 2012;9(2):71–80.

Marty R. Cloud application logging for forensics. In: 2011 ACM symposium on applied computing, Taichung, Taiwan 2011. p. 178–84.

McKemmish R. What is forensic computing? Trends & Issues in Crime and Criminal Justice 1999;118:1–6.

Meky M, Ali A. A novel and secure data sharing model with full owner control in the cloud environment. International Journal of Computer Science and Information Security 2011;9(6):12–7.

Mell P, Grance T. The NIST definition of cloud computing. U.S. Department of Commerce; 2009.

Microsoft. Monitored file name extensions (Windows) viewed 15 July 2013, <http://msdn.microsoft.com/en-us/library/windows/desktop/aa378870%28v=vs.85%29.aspx>; 2013.

National Institute of Standards and Technology. Challenging security requirements for US government cloud computing adoption (draft). Gaithersburg: U.S. Department of Commerce; 2011.

ownCloud. Sync clients | ownCloud.org viewed 12 July 2013, <http://owncloud.org/sync-clients/>; 2013.

Quick D, Choo K-KR. Dropbox analysis: Data remnants on user machines. Digital Investigation 2013a;10(1):3–18.

Quick D, Choo K-KR. Digital droplets: Microsoft SkyDrive forensic data remnants. Future Generation Computer Systems 2013b;29(6):1378–94.

Ruan K, Carthy J, Kechadi T, Crosbie M. 'Cloud forensics: an overview. In: 7th IFIP International Conference on Digital Forensics, Orlando, Florida, USA 2011.

Tassone C, Martini B, Choo K-KR, Slay J. Mobile device forensics: a snapshot. Trends & Issues in Crime and Criminal Justice 2013;460:1–7.

United Nations Office on Drugs and Crime (UNODC). The use of the Internet for terrorist purposes. New York, US: United Nations; 2012.

United States National Institute of Justice (US NIJ). Forensic examination of digital evidence: a guide for law enforcement. Washington, DC: U.S. Department of Justice; 2004.

Waters J. Cloud-based data storage. The Education Digest 2011;76(8):28–34.

Wipperfeld M. How to choose a storage as a service provider. eWeek 2009. 18 February 2009.

Zatyko K, Bay J. The digital forensics cyber exchange principle. Forensic Magazine 2012:13–5. December 2011–January 2012.

Please cite this article in press as: Martini B, Choo K-KR, Cloud storage forensics: ownCloud as a case study, Digital Investigation (2013), http://dx.doi.org/10.1016/j.diin.2013.08.005