Cloud Storage Client Application Evidence Analysis on UNIX/Linux

R. Malik[1], N. Shashidhar[1], and L. Chen[2]
[1]Department of Computer Science, Sam Houston State University, Huntsville, Texas, USA
[2]Department of Information Technology, Georgia Southern University, Statesboro, Georgia, USA

Abstract - *The research proposed in this paper focuses on gathering evidence from devices with UNIX/Linux systems (in particular on Ubuntu 14.04 and Android OS) in order to find artifacts left by cloud storage applications that suggests their use even after the deletion of the applications. The work performed aims to expand upon the prior work done by other researches in the field of cloud forensics and to show an example of analysis. We show where and what type of data remnants can be found using our analysis and that can be used as evidence in a digital forensic investigations.*

Keywords: cloud storage forensics, cloud application artifacts, data remnants, data carving, UNIX/Linux forensics, digital forensic investigations.

1 Introduction

The evolution of cloud computing [1] has certainly increased the importance of cloud forensics in the past few years. In particular, many businesses are now shifting their services from standalone computer devices to the cloud environment [2]. Cloud computing has changed the way by which digital data is stored, processed, and transmitted. The increased use of the cloud environment has brought forth many issues and challenges for digital forensics experts [3]. In fact, the National Institute of Standards and Technology (NIST) has identified 65 of these challenges that needs to be addressed [4]. Nowadays, most people download on their computer or other devices, such as smartphones or tablets, one or more of the popular cloud storage services applications, such as Google Drive, Dropbox, and Microsoft OneDrive. However, there are alternatives to these cloud services that are gaining greater attention by the public. When performing a digital investigation on these user devices, it is possible to find artifacts that suggest the use of any of the above mentioned applications. Previous research has confirmed that these applications do leave a trace even after they are removed on the client devices. These traces, which contain account information, settings, and user activities can be helpful when conducting a digital forensic investigation. However, prior work in this area has only been done on Windows systems, in particular on a Windows 7 operating system. No work so far has been done on UNIX/Linux, even though UNIX/Linux-based operating systems are commonly used as well, especially when providing applications on a server. The research proposed in this paper expands upon the work done by prior research. The chosen operating system is one of the most known Debian-based Linux distributions, Ubuntu. It is very likely that in the most of the cases, a user downloads the client application on multiple devices. For example, the client application can be downloaded on both a PC and a smartphone. Therefore, a second objective of this research was to perform the gathering of evidence on a smartphone, in particular on Android OS. There are multiple cloud storage services options on UNIX/Linux, however, not very surprisingly, Microsoft OneDrive does not provide a client, and very surprisingly enough, at the time of this writing, an official client of Google Drive has not been released for UNIX/Linux yet. Therefore, in order to perform this research, the best alternative to the mentioned client applications were used, such as Copy and OwnCloud. In addition, Dropbox was also analyzed since it represents the most used cloud storage service [5].

2 Prior Work

The following literature review, explores the procedures and approaches used by other researchers in this particular field. Three main prior and related researches were analyzed to discover the approach taken in order to collect artifacts. Artifacts collected can be files either accessed or modified by the cloud storage applications on the client devices, or artifacts related to web-based cloud storage services (which are accessed through a web browser). Two main approaches were identified. The first approach represents a presumption of where artifacts should be located on a device, and then perform a search in those specifics locati ons, based on the examiner's knowledge. Meanwhile, the second approach is based on the use of programs and tools, such as Process Monitor from the Sysinternals Suite [6] to determine the location, in a dynamic manner, of the artifacts and data remnants. All the prior work done in this field was performed on a Windows 7 system using virtual machines. The following is a brief discussion of the prior work.

The paper by H. Chung, J. Park, S. Lee and C. Kang [7] provides a procedure to investigate devices such as PCs and smartphones. According to this procedure, the investigator collects and analyzes data from all devices that a user has used to access a cloud storage service. Based on the type of the device that is being analyzed, the procedure can take a different approach. Simply put, if the device is a PC then it is very important to collect volatile data from physical memory (if live forensic analysis is possible) and nonvolatile data such as files, directories, internet history, and log files. Physical memory contains useful information about users and their activities. For example, physical memory can contain login attempts and login credentials used to access cloud storage accounts through a web

browser. If the device is instead a smartphone, and if the system is running Android OS, after rooting, it is possible to collect data from the main system folders. In the case of an iPhone, after connecting the device to a PC, important data of user activity related to the cloud can be found in backup files or in data synchronized with iTunes. Once all data is collected an analysis in order to find useful artifacts is performed. According to the paper, cloud-storage services can be web-based services accessed through a web browser or client applications installed on the device. In the first case, it is essential for an expert to analyze data such as web browser log and database files (cache, history, cookies, and downloaded files) that are stored in the user profile directory on a Windows system. Cache files include downloaded image files, text files, icons, HTML files, XML files, download times, and data sizes. History files contain visited URLs, web pages titles, visit times, and the number of visits. Cookie files store information about hosts, paths, cookies modification times, cookies expiration times, names, and values. Download lists include local paths of downloaded files, URLs, file size, download times, and whether downloaded files were successful. Through such web browser files an expert can identify a user's activity, including access or logins to a cloud storage service. However, when a client application is installed on a Windows system, traces of it are left in the registry, log files and database files. Mac systems have similar traces except registry files. These files are essential during a digital forensic analysis, since they provide proof of the use of a cloud storage service. These log files contain information such as logins attempts, if and when services were used, and times of synchronized files. Database files contain information about synchronized folders and files (creation times, last modified times, and whether files were deleted) on a PC. All this information can be used to create a timeline of the user activity. In a smartphone device traces are left in database files, XML files, and plist files (which contain information about a user account). Finally, the rest of the paper provides examples of forensic analysis and shows where data is found on a PC or a smartphone. The cloud services that were used in this work, are Amazon S3, Dropbox, Google Docs, and Evernote. In the research work done by M. Katz and R. Montelbano [8], to obtain the locations of artifacts, Process Monitor is used. The result is filtered to show the file system activity, and changes to the registry and files. The cloud storage applications used in this research are SkyDrive (now OneDrive), Dropbox, and Google Drive. When SkyDrive was installed 4959 artifacts were either created or modified. Presence of the files modified using the client, was found in unallocated space, $Recycle.Bin CSV files, pagefile.sys, and inside the AppData folder. In the case of Dropbox installation, 4163 artifacts were either created or modified. Evidence of deleted files was found in unallocated space and in pagefile.sys. During Google Drive installation, 9438 artifacts where either created or modified. Evidence of files modified or deleted was found in unallocated space, $Recycle.Bin CSV files, pagefile.sys, and configuration files. The result of this research proved that a large number of files are affected during the installation of the application, and a large number of files are left behind, once the uninstallation process is completed. Evidence of file manipulation was mainly found in the unallocated space, $Recycle.Bin CSV files, and pagefile.sys. The type and number of artifacts varied depending on the application, but evidence of the use of the cloud application was still present after uninstallation of the client in all the cases.

The following research, performed by D. Quick, B. Martini, and R. Choo [9], provides a well formatted methodology and a very exhaustive analysis of data remnants left by cloud storage applications. This research is performed on a Windows 7 machine and the cloud storage services analyzed are Microsoft SkyDrive (again, now OneDrive), Dropbox, and Google Drive. The objective of the research was to solve questions, such as, which data remains on the hard disk after a user used the client software? Which data is left once the user has had access to the cloud storage through a web browser? What is the location of the data remnants on the operating system and in the memory? Other questions that were attempted to answer relate to network traffic data and smartphones. Based on the work of this research, artifacts of files either access or modified, and data remnants left behind by the applications are found inside prefetch files (which are used to analyze the software activity, such as the number of times the software has run or the associated files used by the application), registry files (they can contain references, activities, settings or other information), link files (files' shortcuts), thumbnails pictures within the thumbcache, event logs (which contain information relating to system, software, and other events recorded by the operating system), and finally, directory lists file ($MFT files). Forensic analysis has been also performed on the memory, $Recycle.Bin (in order to find deleted files), client applications (analysis of installation path, sample files, synchronized files and folders), on the account accessed through a web browser (can contain information about the number and the type of devices used to access the storage space), and finally, on the files related to the browser. Network traffic was also captured and analyzed to find activity related to login sessions. To conclude, data carving was performed through allocated and unallocated data. Thumbnails icons and large size pictures were recovered. This research, used a dynamic approach: tools to dynamically find evidence that were used were Process Monitor, Wireshark, among others. Another research was performed by M. Epifani et al [10], on Microsoft SkyDrive (OneDrive), Google Drive, Dropbox, and iCloud. Again in this case, the collection of artifacts left behind by the applications was performed on a Windows 7 system. To track the disk usage DiskPulse was used (to determine information related to created, modified and deleted files), Regshot, and RegFromApp were used to track registry changes. By monitoring the registry changes, researchers were able to obtain installation locations and installed client applications versions. Other useful data was collected from configuration files present in the installation folder (inside the user profile), from online accounts (information about deleted files, devices connected to the account, version history for every file, and last browser sessions), from the memory (it can contain user email, display name, filecache.dbx path, server time, file list, deleted files, username and passwords in the case of a web-based storage access), from Hiberfil.sys and pagefile.sys, link files, browser history and cache, registry point, and volume shadow copies. As

we can see, this research collected evidence from the same locations as the previous researches.
To conclude this literature review, we can assert that the procedures and approaches taken in these prior research works are in concordance with each other. Even if the approaches were of two different types, the locations analyzed and the data remnants found were similar.

3 Methodology

To perform our research, different programs and tools were used. *VMWare Workstation 10* was used to create a virtual machine of *Ubuntu 14.04 ("Trusty Tahr")* 64-bit. Once the installation was complete, a snapshot was taken in order to revert to a clean state once the analysis of the client application was concluded. After the client application was installed, a keyword search on the system was used to find the locations of the files and other artifacts. Specifically, the command "*find / print | grep –i 'keyword'*" was executed on the terminal, and the output was filtered to gather paths, files, and other artifacts. Unfortunately, there is not a version of *Process Monitor* (from the *Sysinternals Suite*) for Linux. Nevertheless, the command *lsof* (list open file) can be executed to list files opened by the client process, by using its PID. The tool used to acquire memory is *LiME* [11]. In addition, a keyword search (using for example login credentials) on the memory was performed in order to find artifacts.
Ubuntu 14.04 64-bit uses *Ext4* as a default file system and to find where the locations of files on the file system, *TheSleuth Kit* [12] was used. In particular, the *istat* tool is useful to find *inode* information, status of directories, and status about files inside the main directory that are created by the cloud storage application in order to synchronize files with web-based the account. The *istat* tool was also used to determine the group to which the *inode* belongs, and to find the block address where the content of the data is stored. Using the *dd* command it is possible to create an image of the partition or, as explained later, to image ranges of blocks and groups. The images can be viewed with a *hex editor*. *Foremost* [13] is an example of a data carving tool that makes it possible to recover deleted files. Finally, *SQLite Browser* [14] was used to open database files created by a web browser, such as *Mozilla Firefox,* or by cloud storage applications. The *Android OS* analysis was performed on an *Android OS x86 RC1* virtual machine.

4 Main Research

4.1 Research

As pointed out in the previous work section, there are two means to connect to the cloud storage account. The first is through a web browser, while the second is through a client application. It is important, therefore, to gather information in both cases. This section contains many different subsections that explain where evidence can be found on both Mozilla Firefox (which is installed by default on Ubuntu operating systems), and in the main directories of the client applications. As previously seen, another source of useful information is found on a running system inside the physical memory. Therefore, a subsection is dedicated to the volatile information found in the physical memory. One subsection will also briefly describe where other artifacts can be found on the file system for all the applications and, finally, the last two subsections are dedicated to the possibility of recovering deleted files, and the gathering of data on Android OS.

4.2 Web Browser Analysis

All the web-based accounts for all the applications were accessed through Mozilla Firefox and, after operations such as uploading, deleting, and modifying files, the memory was captured and analyzed. Inside the memory it is possible to find in plain text users' credentials such as email addresses, usernames and passwords in plain text, user IDs, first names, last names, logins attempts, timestamps of logins, paths on the server, server names or addresses, references to files accessed, timestamps of creation times, data of last synchronization, files uploaded, files sizes, files modification times, files deletion times, messages and actions taken by the server. Accessing the hidden folder inside the user's home directory, by default named *.mozilla*, it is possible to find database files that store useful information. The database file *contentprefs.sqlite* contains preferred websites, such as the cloud storage websites accessed. The file *cookies.sqlite* stores cookies that are created when the cloud storage website was accessed. The *formhistory.sqlite* database file contains the history and tracks the websites visited by the user, the number of visits, along with the last visited time. The database file *places.sqlite* contains multiple tables where evidence of the use of the cloud storage services websites can be found. For example, the table *moz_places* stores the name of the server, paths, and files accessed on the server, while the table *moz_hosts* and *moz_favicon* store hosts names and icons paths. Finally, the file *login.json* stores logins attempts, hostnames, usernames, URLs of the forms in which credentials were inserted, encrypted passwords, and MAC times. Inside the user's home directory, the hidden directory *.cache* contains thumbnails that show pictures of websites and accounts accessed.

4.3 Client Application Analysis: Copy

When the Copy client application is executed it launches a process is called *CopyAgent*, which functions as a background process and enables the application to properly run and execute its operations. Once the location of the data remnants was found, by sorting through the artifacts, it is possible to determine that the main evidence resides in the Copy main folder and the hidden folder *.copy*, both present in the user's home directory. The Copy main folder contains the files that are synchronized with the account, a hidden folder called *.copycache*, which contains temporary files (references to these files were found during the memory analysis), and a hidden text file called *.user_info*, which does not contain any useful information. The hidden *.copy* folder contains two directories: *cache* and *resources*, however they are both empty. The most interesting information is contained in the following files: *config.ini* contains the host *UUID*, which is an alphanumeric string of 32

characters; the file *config.db* contains settings, such as the root cache path, the database version, the cloud server address, the root folder path, the authentication token, the client ID, the user ID, the user first name, last name, email address, the push token and push URL, among other settings. The file "*copy <user_email>.db*" (in this research the email address used is *cloudstorage.test.mail@gmail.com.db*) contains a list of files that are uploaded on the cloud storage space or the files that are synchronized with the local folder. The *file* table in this database, contains the files metadata such as the file path, name, parentID, volumeID, inode address, attributes, mtime, rstate, ctime, size, and child count. Other tables in this database contain more useful information such as the owner ID and the file fingerprint. The *synclog.txt* file, logs the application operations such as logins attempts, uploaded files, deleted files, and modified files, along with other timestamps. Finally, the file *trace.txt* stores information regarding the user, the process (*CopyAgent*), the operating system, and the server. Another Copy configuration file is contained inside */home/rakesh/.config/Barracuda Networks, Inc\Copy.conf*. Once the client application is removed, the *find* command revealed folders and files left on system. Since *CopyAgent* was never installed but simply run from the Copy client downloaded folder, once it is removed, it will not delete the Copy main folder; however, a user can simply delete the Copy main folder found in user's home profile. If the user is not aware of the hidden *.copy* folder, this will remain on the system, along with all the information contained, which represents a great deal of useful information during a forensic analysis. Even if the Copy main folder is deleted, both the inode entries and the data content are still present on the hard disk, and unless it is wiped or the data is overwritten, recovery of deleted files is possible (see *Recovering Deleted Files subsection*).

4.4 Client Application Analysis: ownCloud

Once the client application of ownCloud is installed, the main folder is by default present in the user's home directory. Unlike Copy, there is no hidden folder, but hidden files are present in the client main folder inside the user's home directory. This folder contains the files that are synchronized with the account, along with the hidden database file *.csync_journal.db*. The table *metadata* of the database file, stores valuable information regarding files, such as paths, inode addresses, UIDs, GIDs, MAC times, md5 hashes, among other information,. The table *version* contains the version of the client application. The hidden file *.owncloudsync.log* is a log file that stores timestamps, operations executed by the application (instructions), the length of the operations, the names of the files involved, MAC times, sizes, the file IDs, among other useful information. The configuration file found in the following path and named */home/<user_name>/.local/share/data/ownCloud/owncloud.cfg* contains the URL of the ownCloud server, the username, authentication type (http), and other settings. After removing the application, the files on the local system will still be present. Therefore a user has to manually delete the ownCloud main directory. Unfortunately, from a forensics point of view, the database containing the metadata and the log files, even if are hidden, are removed with the directory, that includes also the stored files. However, inode entries and data content are still present on the hard disk, so as for Copy, unless it is wiped or overwritten, recovery is still possible.

4.5 Client Application Analysis: Dropbox

Dropbox, unlike Copy and ownCloud, encrypts the database files and log files for security purposes. The encryption keys are not released even to the user. However, a tool exists for Windows versions that decrypts these files. Unfortunately, this tool does not have a counterpart for UNIX/Linux, nevertheless, according to the Magnet Forensic team, a version of the decryptor should be available for UNIX/Linux in the future [15]. Once the application is installed and the account is set, the Dropbox main folder is created in the same location as Copy and ownCloud, which is the user's home directory. The main Dropbox folder maintains a hidden cache directory that stores the files synchronized with the account. In addition to the main folder, there is a hidden folder named *.dropbox* inside user's home directory, which contains multiple interesting files and subdirectories: a file named *info.json* stores the path of the dropbox root directory and the file *dropbox_pid* that contains the Process ID, the sub-directory *instance1* stores all the encrypted files so it is not possible to collect evidence unless the files are first decrypted. One of these file, is *aggregation.dbx,* contains timestamp values, server paths, and a blocklist value. The encrypted files are *config.dbx* (contains configuration settings, user, host machine and server information), *deleted.dbx*, *filecache.dbx*, (contains files metadata), *notifications.dbx*, *sigstore.dbx*, *PENDING_cHDqrT*, *TO_HASH_3WP99a*, and *UPDATE_XyrFxy*. The folder *instance_db* stores another encrypted file named *instance.dbx*. When uninstalled, Dropbox does not delete the main Dropbox directory in the /home/rakesh/ folder. Even if the folder is deleted, a user might be unaware of the presence of the hidden folder *.dropbox*, and this will be helpful if the database files and logs can be decrypted. Like Copy and ownCloud deleted file are possible to recover.

4.6 Physical Memory Analysis

As mentioned in the Methodology section, physical memory was acquired using LiME and it was analyzed with keyword searches. Volatile evidence found includes names of files present in the application's main folder, recently accessed folders, name and client application main process information, process instructions (for example, in the case of Copy, *copy-sync, copy-update, copy-paused, or owncloud_init, owncloud_commit, owncloud_opendir* in the case of owncloud, etc.) icons paths, hosts names, loaded libraries and modules used by the process, libraries imported through the server, temporary files created by the client application main process in the respective application cache folders, database files accessed, log files accessed, log files entries, logins attempts and credentials (for example, during the memory analysis after the use of Copy, the string *–Loginsuccess U:*

'cloudstorage.test.mail@gmail.com' was found.

4.7 File System Analysis

Several artifacts are found in various locations of the file system. However, most of these files are not useful during a digital forensic investigation, but to only demonstrate the installation and use of the client applications by the user. The directories */usr/share/man* and */usr/share/doc* on Ubuntu contain the man pages and documentation of the applications, while the directories */bin/* and */usr/bin* store executables or daemons files used to launch the applications. A great number of icons related to the cloud storage applications are found inside */home/<user_name>/.icons*. Information can also be found inside systems logs: for example, the log auth.log inside */var/log* stores command issued on the terminal, along with the working directory and it can be used to analyze the user recent activity. The log file */home/<user_name>/.bash_history* contains a list of the recently commands issued by the user. The *dpkg.log* contains references to packages installed on the system, and it can reveal if any cloud storage client application was installed from the command line. Other possible locations that may contain useful evidence are the folders */home/<user_name>/.local/share/Trash,* which stores deleted files, and */home/rakesh/.cache/thumbnails*, which contains thumbnails associated with the client applications. If an application during its execution should crash, the *syslog* and *kern.log* files found inside */var/log* store references of the crash. The log file *history.log* contains the recently installed packages and the command issued to install them, while *term.log* contains libraries used to install the applications.

4.8 Deleted Files Analysis

We clearly established in the Prior Work section that it is possible to find artifacts of cloud applications inside the unallocated space. Therefore, it is possible to recover the data stored inside this space. This subsection will be dedicated to the recovery of deleted files, which is a process known as data carving. Recovering deleted files can be done using two different methods. The first is to recover the deleted files from the server, but it requires the application to be logged in, or the account to be accessed through a web browser. Copy and ownCloud applications both allow the user to recover deleted files with a feature known as '*undelete*'. Dropbox does not have a similar feature, but it is possible to recover deleted files or previous version of files on an account accessed through a web browser. However, unlike Copy and ownCloud, Dropbox allows a permanent deletion option, that will permanently delete a file from an account. The second way to recover deleted files is through the use of a data carving tool on the local device.

All the client applications analyzed in this research, when they first synchronized with the server, downloaded a copy of each file and stored them locally. As mentioned earlier, even if the files are deleted, the *Ext4* file systems does not delete the *inode entry*, so both the *inode entries* and the files' *data content* are still present on the hard drive. To see how the recovery is done, first we have to analyze how the file are allocated. Using the *istat* tool from the *TheSleuth Kit* an examiner is able to find the *inode addresses* of the files, the *group* to which they belong, and the *block address* of where the *data content* of the files is stored. The files are usually (but not necessarily) stored in contiguous *inode addresses*. Using the *blkstat* tool on the *block addresses* that contain the files' *data content*, the output shows that the *blocks* are in fact allocated. By executing the *fsstat* tool it is possible to find the *inode range address* of the *group* and the range of the *blocks* where the *data content* is present (usually stored in a different group). Using *dd* on these ranges it will allow a forensic expert to create a small image that can be further analyzed with a hex editor. Now, we can examine if the files are still present once they are deleted. After deleting the files, *blkstat* will return that the *blocks* that store the *data content* are not allocated anymore. However, by opening the images with a hex editor, it is possible to determine that the data is still present inside the *block*. In addition, the inode entries are still present (a character has been added before the file name to indicate that the files pointed by the inode have been deleted). At this point, using a simple data carving tool such as *Foremost* or *icat* from the *TheSleuth Kit*, recovering the deleted files is fairly easy.

4.9 Android OS Analysis

A common featre of cloud storage services is that they allow multiple devices to be synchronized. Therefore, it is very likely that one user will synchronize the same cloud storage account on multiple devices. Thus, in order to gather as much data as possible, it is important to analyze not only workstations, desktops, and laptops but also devices such as smartphones and tablets. An operating systems that comes as default on many smartphone is Android OS. Client applications were both available for Copy and Dropbox, however, an official client at the time of the writing of this paper is not available for ownCloud. Nevertheless, an unofficial but very efficient client application has been published by the BezKloboukuNos team [16]. In the case of Copy the main files are found inside the folder */data/media/0/Android/data/com.copy/files*. Among these files, the *configuration.ini* file contains different configuration and account settings, while the file *copy.db* stores names of the synchronized files along with the metadata of the files. The log file *trace.log* stores information regarding logins, synchronized files, timestamps, account information, and the client application's process operations such as uploading, deleting and/or modifying files. The *download* directory inside the *temp* subdirectory contains the files that were locally saved, while the directory *thumbnails-cache* contains thumbnails. The file *com.copy_preferences.xml* in the directory */data/data/com.copy/shared_prefs* stores other useful data such as the last opened file along with its timestamp. When the Copy application is deleted from Android OS, all the files are also deleted. However, information about the use of the application can be found in logs, such as *logcat*, which contains the system recent activity. Operations and actions executed by the Copy application, files and directory accessed were found inside *logcat*. ownCloud does not have an official client, but as

mentioned above, an unofficial client is available. Once the application is installed, the directory */data/media/0/owncloudApp* stores the subdirectory *owncloud_user@server_address*, which stores files that were downloaded locally on the device. The database file named *filelist* in the folder */data/data/com.owncloud.androidApp/databases/* stores the list of the files synchronized along with the type of the file, the username, server name and address. The file *com.owncloud.android/App_preferences.*xml stores as well, the username and server address. When the application is removed, the directory */data/media/0/owncloud* still contains the files that were locally stored, along with other temporary files. The rest of the files created by the application are deleted. Also in this case, recent activity can be found in logs such as *logcat.* The directory */data/data/com.dropbox.android/* stores local files, and the subdirectory *databases* contains database files. Unfortunately, also in the case of Android OS most of these files are encrypted. A file named *361753330-db.db* inside the directory, stores the names of the synchronized files, and the file *prefs-shared.db* stores account preferences. Finally, recent activity of the user, including the use of Dropbox, can be found using *logcat*. A lack of analysis tools for Android OS, did not let us perform a dynamic search to gather the useful data remnants. In fact, the files above mentioned and listed were found through a manual search.

5 Conclusions and Future Work

The research proposed in this paper highlighted the main locations and files on a client device with an Ubuntu 14.04, and more generally, a UNIX/Linux operating system. These files can be used as evidence related to the use of cloud storage service client applications. Although the operating system is different, as well as the file system, the results obtained by previous researches, and the results of this research are in accordance and similarities arise. Differences arise due to the different feature of the Windows 7 operating system and the UNIX/Linux OS, therefore locations and file types may vary. In fact, the registry, which is a classic feature of Windows is not present on UNIX/Linux systems so there will be no evidence related to it. Unfortunately, the amount of tools used to analyze dynamically a process, such as Process Monitor does not have a valid counterpart for UNIX/Linux, therefore a dynamical analysis to gather evidence is a bit harder. Nevertheless, it was still possible to find evidence in the hidden directories or hidden files created by the application, as well as in database or log files, inside web browser files, in the memory, and in both allocated and unallocated space. Many other locations and files on the file system, such as the cache, system logs, and configuration files stored useful data. Files stored on Android OS are similar to the files stored on laptops or desktops. However, one big difference is that, in the case of Ubuntu, the cloud application downloads and stores a copy of each file locally, while in the case of Android OS, a copy of a file is stored locally only if the file is accessed by the user. The fact that the files are stored locally, is really helpful during digital investigations since deleted files and their metadata are still present on the hard disk, unless the unallocated space is wiped or overwritten. Recently, most of the main cloud storage services, such as iCloud or Dropbox, started to encrypt their files, which makes it harder for digital experts to gather useful evidence. With a proper search warrant, the cloud service provider may be able to decrypt the files and allow forensics experts to gather evidence. Another key factor that can be seen in this analysis, is that even if the applications are different, the configuration files, database files, and log files (and use of hidden files and directories, cache directories and temporary files, database and log files) are similar. Finally, this research highlighted that a great deal of evidence can still be found on the Ubuntu system once the application is uninstalled or simply removed.

6 References

[1] Columbus, L. "Predicting Enterprise Cloud Computing Growth". Sepember 4, 2013. Available: http://www.forbes.com/sites/louiscolumbus/2013/09/04/predicting-enterprise-cloud-computing-growth/

[2] Cohen, R. "The Cloud Hits the Mainstream: More than Half of U.S. Businesses Now Use Cloud Computing". April 6, 2013. Avaialable: http://www.forbes.com/sites/reuvencohen/2013/04/16/thecloud-hits-the-mainstream-more-than-half-of-u-s-businessesnowuse-cloud-computing/

[3] NIST. "Drafts Cloud Forensics Standard". Information Management Journal, September, 2014.

[4] NIST. "Cloud Computing Forensic Science Challenges". Draft NISTIR 8006, NIST Cloud Computing Forensic Science Working Group Information Technology Laboratory, NIST, June, 2014. Available: http://csrc.nist.gov/publications/drafts/nistir8006/draft_nistir_8006.pdf

[5] Henry A. "Most Popular Cloud Storage Provider: Dropbox". Lifehacker, August 2, 2013. Available: http://lifehacker.com/most-popular-cloud-storageproviderdropbox-644624306

[6] Process Monitor v3.1, TechNet, Available: http://technet.microsoft.com/en-us/sysinternals/bb896645

[7] Chung, H., Park, J., Lee, S., & Kang, C. "Digital forensic investigation of cloud storage services". *Elsevier*, May 4, 2012. [8] Katz M., Montelbano R. "Cloud Forensics", The Senator Patrick Leahy Center for Digital Investigation, Champlain College, 4 November 2013.

[9] Quick, D., Martini, B., & Choo, R. "Cloud Storage Forensics", 1st ed., p. 208, Syngress, 2013.

[10] Epifani, M. "Cloud Storage Forensics", SANS European Digital Forensics Summit, Prague, 2013, Available: https://digital-forensics.sans.org/summitarchives/Prague_Summit/Cloud_Storage_Forensics_Mattia_Eppifani.pdf

[11] LiME, Linux Memory Extractor, Available: https://github.com/504ensicsLabs/LiME

[12] Carrier, B. The Sleuth Kit: Download. Available: http://www.sleuthkit.org/sleuthkit/download.php

[13] Foremost, SourceForge.net, Available: http://foremost.sourceforge.net

[14] DB Browser for SQLite. Available: http://sqlitebrowser.org/

[15] Dropbox Decryptor: A Free Digital Forensics Tool. Available: http://www.magnetforensics.com/dropbox-decryptor-a-free-digital-forensics-tool/

[16] Client for ownCloud, BezKloboukuNos, 2014, June 29 Available: https://play.google.com/store/apps/details?id=com.owncloud.androidApp&hl=en