

Identifying Remnants of Evidence in the Cloud^{*}

Jeremy Koppen, Gerald Gent, Kevin Bryan, Lisa DiPippo, Jillian Kramer^{**},
Marquita Moreland^{***}, and Victor Fay-Wolfe

University of Rhode Island,
Kingston, RI USA
{bryank,dipippo,wolfe}@cs.uri.edu

Abstract. With the advent of cloud computing, law enforcement investigators are facing the challenge that instead of the evidence being on a device that they can seize, the evidence is likely located in remote data centers operated by a service provider; and may even be in multiple locations (and jurisdictions) across the world. The most practical approach for an investigator when cloud computing has been used is to execute a warrant that requires the service provider to deliver the evidence. However, to do this, the investigator must be able to determine that a cloud application was used, and then must issue a warrant with reasonable scope (e.g. the subject's username at the cloud provider, the name of the documents, the dates accessed, etc). Fortunately, most cloud applications leave remnants (e.g. cached web sites, cookies, registry entries, installed files, etc) on the client devices. This paper describes the process for identifying those remnants and parsing them to generate the data required by law enforcement to form warrants to cloud service providers. It illustrates the process by obtaining remnants from: Google Docs accessed by Internet Explorer, Dropbox, and Windows Live Mesh.

Keywords: cloud computing, cloud forensics, digital forensics.

1 Introduction

Cloud computing, where applications and data storage are provided as services to users via the Internet, is becoming more and more prevalent - and because of it, law enforcement investigators are facing new challenges in obtaining evidence. Instead of the evidence being on a device that they can seize, the evidence is likely located in a data center at a service provider that is often not geographically easily accessible. In fact, the data may be stored in multiple physical locations (and jurisdictions) across

^{*} This work was supported by a grant from the U.S. Department of Justice's National Institute of Justice Electronic Crimes Research and Development program – Grant # 2011-FD-CX-K011.

^{**} Supported as a National Science Foundation Research Experience For Undergraduate student researcher from Villanova university under grant NSF 1004409.

^{***} Supported as a National Science Foundation Research Experience For Undergraduate student researcher from Purdue university under grant NSF 1004409.

the world. The problem is particularly acute for law enforcement investigators from smaller organizations, where extensive traveling to obtain evidence is not feasible. Furthermore, the volume of data kept by these service providers is so vast and the data is so complex that it is often impractical for an investigator armed with a warrant to extract the evidence from the data centers of most service providers, even if he/she were physically present.

The most practical approach for State and local law enforcement is to execute a warrant through the service provider's Keeper of Records that requires the service provider to deliver the evidence. This mitigates the issues of having to travel to remote and multiple physical locations, and issues of needing to understand data formats to find the evidence in vast data storage centers. Although there are other potential problems with this approach, such as uncooperative service providers, a warrant to a service provider to acquire evidence is the best means available to law enforcement when cloud applications have been used by a suspect.

However, there are several substantial barriers to an investigator obtaining evidence from cloud service providers. First, the investigator must be able to determine that a cloud application was used. Typically all they have to work from are seized devices (computers, phones, etc); and determining that the suspect was using a cloud application by examining a seized device is often very difficult. Second, even if law enforcement seizes the suspect's devices, the suspect may use other devices/means to access his/her cloud data to modify or delete it. This makes it essential that law enforcement *quickly* deduce that cloud computing was used so that they can issue preservation orders to the service provider. Third, to meet practicality considerations and restrictions on scope, preservation orders and warrants must be specific and indicate details such as the cloud application used, the user account, the dates it was used, and files of interest - information that can be even more difficult for the investigator to obtain from the suspect's devices. Finally, cloud computing is in its "Wild West" stage where new cloud applications are coming and going daily. Keeping up with which cloud applications the suspect might have used and where on devices that the applications keep the data necessary for a preservation order and warrant is impractical for State and local law enforcement investigators.

This paper reports our research to determine what remnants are left on devices (computers, phones, iPads, etc.) and how to collect and present those remnants necessary for law enforcement to meet restrictions on scope in warrants and preservation orders served to the cloud service provider. These remnants of cloud applications that are left on devices include data found in file system data structures, cached web sites, cookies, index.dat entries, registry entries, and several other places on devices used by the suspect. This includes information such as the cloud applications used, usernames at the service provider, dates and times that the cloud applications were used, and cloud application document names involved.

Section 2 provides background on cloud computing and related forensics tools. Section 3 describes the process used to identify and find remnants. Section 4 provides details on the initial cloud applications that we used for proof of concept. Section 5 summarizes and describes the next steps in creating a full, robust, tool to support law enforcement in investigations that involve evidence in The Cloud.

2 Background

This section presents background on cloud computing and on related digital forensics tools that have established the paradigm of focusing on specific classes of evidence – the paradigm on which we base the notion of searching specifically for cloud remnants.

2.1 Cloud Computing

According to the latest definition from NIST, *cloud computing* is "a model for enabling convenient, on-demand network access to a shared pool of configurable resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction." [1].

NIST's cloud model promotes availability and is composed of five essential characteristics:

- On-demand self-service
- Broad network access
- Resource pooling
- Rapid elasticity
- Measured service

Most cloud computing infrastructures consist of services delivered through common centers and are built on servers. Clouds often appear as single points of access for users' computing needs. The major cloud service providers include Amazon [2], Apple [3], Dropbox [4], Rackspace Cloud [5], Salesforce [6], Skytap [7], Microsoft Windows Live [8] and Google [9]. Some of the larger IT firms that are actively involved in cloud computing are Huawei [10], Cisco [11], Fujitsu [12], Dell [13], Hewlett Packard [14], IBM [15], VMWare [16], Hitachi [17] and NetApp [18]. A more complete list is provided in Section 4.1.

The fundamental concept of cloud computing is that the computing is "in the cloud" i.e. the processing (and the related data) is not in a specified, known or static place(s). In fact, data may be stored in many different locations. This is contrary to what law enforcement investigators are more used to: where processing takes place on a client device or in one or more specific servers that are known. Although an empowering and "freeing" concept for users, the removal of data to the cloud presents problems for law enforcement who often need to find the place(s) of the evidence.

Cloud Architecture. NIST's cloud general architecture that describes the delivery of cloud computing, typically involves multiple cloud components communicating with each other over application programming interfaces, usually web services.

The two most significant components of cloud computing architecture are known as the *front end* and the *back end*. The front end, also called the *cloud client*, is the

part seen by the user. This includes the user's device (computer, phone, etc) and the applications used to access the cloud, such as a web browser. The back end of the cloud computing architecture is the 'cloud' itself, comprising various computers, servers and data storage devices networked together.

Cloud services can be delivered at three levels: Service, Platform and Infrastructure, and any particular cloud application can provide one or more of these services.

Cloud application services, also known as "*Software as a Service (SaaS)*," deliver software over the Internet, eliminating the need to install and run the application on the customer's own devices and simplifying maintenance and support. People tend to use the terms 'SaaS' and 'cloud' interchangeably, when in fact SaaS is just one type of cloud service available. Key characteristics of SaaS include:

- Network-based access to, and management of, commercially available (i.e., not custom) software;
- Activities that are managed from central locations rather than at each customer's site, enabling customers to access applications remotely via the Web;
- Application delivery that typically is closer to a one-to-many model (single instance, multi-tenant architecture) than to a one-to-one model, including architecture, pricing, partnering, and management characteristics;
- Centralized feature updating, which obviates the need for downloadable patches and upgrades.

Cloud platform services or "*Platform as a Service (PaaS)*" deliver a computing platform and/or solution stack as a service, often consuming *cloud infrastructure* and sustaining *cloud applications*. It facilitates deployment of applications without the cost and complexity of buying and managing the underlying hardware and software layers.

Cloud infrastructure services, also known as "*Infrastructure as a Service (IaaS)*", delivers computer infrastructure - typically a platform virtualization environment - as a service. Rather than purchasing servers, software, data-center space or network equipment, clients instead buy those resources as a fully outsourced service. Suppliers typically bill such services on a utility computing basis and amount of resources consumed (and therefore the cost) will typically reflect what the client uses.

Cloud Data. The cloud model has been criticized by privacy advocates for the greater ease with which the companies hosting the cloud services can control and monitor the communication and data stored between the user and the host company [19, 20]. Regulations governing cloud data storage include FISMA [21], HIPAA [22] and SOX [23], the credit card industry's PCI DSS [24], and SAS 70 Type II certification [25]. In fact many research projects on "cloud forensics" tend to focus on privacy issues – not the pragmatic issues of how to perform forensics with a valid warrant when the suspect uses cloud computing.

Significance of the Cloud. According to an April 2011 forecast by Forrester Research, the volume of the global Cloud computing market will reach \$241 billion by the year

2020, from just \$40.7 billion in 2009 [26]. Similarly, a report from 451 Market Monitor predicts a 24% compound annual growth rate (CAGR) of Cloud computing revenue between 2010 and 2013 [27]. Cisco reported in 2011 that global Cloud IP traffic will increase twelvefold in the five years from 2010 to 2015 [27]. This prediction indicates an overall CAGR of 66% over that time period. According to the IDC, the revenue of worldwide public Cloud services has a growth rate which exceeds that of the global IT market as a whole by a factor of four [28]. This rapidly increasing popularity is becoming a considerable contribution to the IT market's overall growth.

Law Enforcement Investigations and Cloud Computing. Given the rapid growth in the use of Cloud Computing, the Cloud will likely be the next evolution in the history of computing, following in the footsteps of mainframes, minicomputers PCs, servers, smart phones, and so on. It could radically change the way enterprises manage information technology. As such, cloud computing has the potential to be the next disruptive technology that prevents law enforcement from performing effective digital forensics [29].

2.2 Digital Forensics Tools

The concept of identifying specific classes of remnants that this project uses is consistent with a trend in digital forensics tools - the use of focused special-purpose tools that integrate with an overall forensics tool suite. These special-purpose tools focus on a specific class of evidence, gather it, and present it either as a report or as data suitable to be imported into a forensic analysis tool such as FTK [30], EnCase[31], or X-Ways[32]. Some of these focused tools include:

- *P2P Marshal* - ATC-NY developed P2P Marshal [33] to automatically analyze peer-to-peer (P2P) usage on disk images (Forensic Edition) and live systems (Field Edition). It detects what P2P client programs are, or were, present, extracts configuration and log information, and displays the shared (uploaded) and downloaded files. It also includes extensive search capabilities and a thumbnail browser and image viewer. The tool produces reports in RTF, PDF, and HTML formats and runs on Windows machines.
- *Mac Marshal* - ATC-NY developed Mac Marshal [34] to analyze Mac OS X file system images. It scans a Macintosh disk image, automatically detects and displays Macintosh and Windows operating systems and virtual machine images, then runs a number of analysis tools on the image to extract Mac OS X-specific forensic evidence written by the OS and common applications. Mac Marshal Forensic Edition runs on an investigator's Mac workstation to analyze a disk image. Mac Marshal Field Edition runs on a Mac target machine from a USB drive. It extracts volatile system state data, including a snapshot of physical RAM. Mac Marshal follows forensic best practices and maintains a detailed log file of all activities it performs. It produces reports in RTF, PDF, and HTML formats, and runs on Mac OS X-based analysis machines.

- *Cyber Marshal Dropbox Reader* – ATC-NY released this collection of command line tools to parse Dropbox configuration and cache files stored as SQLite databases. The reader works with Dropbox Cloud storage software on Windows, Macintosh, or Linux systems [35].
- *Gargoyle* - Wetstone Technologies developed Gargoyle [36] to detect malware on a computer. It has data sets of remnants indicative of over 10,000 programs, most of them being malware. It can mount disk images, presents reports of the likely software it finds, and exports to formats for import to EnCase, FTK, and spreadsheets.
- *NetAnalysis* - Digital Detective developed the NetAnalysis tool [37] to collect and report browser remnants (cached web sites, history, cookies, etc) from disks and disk images in an easy to understand and easy to use format.
- *Internet Evidence Finder* – JADsoftware Incorporated created this tool to scan a computational device and identify web browser artifacts. It is designed to find existing and deleted data that has been left behind by Internet communications. The reporting style allows the user to search, filter, and bookmark results [38].
- *RedLight* - Our group at the URI Digital Forensics Center developed RedLight [39] to detect pornography in files on mounted drive or drive image. RedLight is based on how law enforcement investigates a case - by finding likely pornography in images very quickly, allowing visual confirmation by the investigator through a display of thumbnails, and then exporting selected images, reports, and hash sets suitable for importing into EnCase, FTK, and X-Ways.

All of these tools, and many other useful digital forensics tools, are designed to search computers and/or disks or disk images, find a particular class of evidence (e.g. peer-to-peer application remnants, malware remnants, browser remnants, etc), and report it to the investigator and/or allow its import into a major analysis tool.

There are no forensics tools specifically meant to collect and report remnants of Cloud applications – this research is the first step towards that goal.

3 Finding Cloud Remnants

This section presents our approach to determining what remnants cloud applications leave. There are two primary classes of cloud applications: *browser-based*, and *installed components*. Browser-based cloud applications execute exclusively in the browser and thus their remnants are found in browser remnants. Installed component cloud applications require software to be installed on the device.

3.1 Cloud Remnant Locations

The remnant data sets vary based on the operating system (e.g. Windows, MacOS, Linux, Android, iOS, etc.) and for browser-based application they also depend on the browser. We have identified several key places that cloud application remnants can be found.

- *Cached web sites* - Cached web sites are the files that the browser downloads and stores on the client device when a web site is requested. Cached web sites will often indicate the cloud application used, the dates it was used (file modified, created, access times), the username (which often appears in a "logged on" message on the cloud application web pages), account names and kind of account, and more. Most cloud applications display this information in known areas of the web page using known HTML-based formatting tags that the resulting cloud analysis tool can search for. How browsers cache web sites varies based on the browser, so the remnant data set will have to be developed differently for all prominent browsers, but the web page content that it looks for will typically be the same.
- *Web history* - Web sites visited are often tracked in web history files, `index.dat`, files and places like the Windows registry. The URLs in the web history indicate that the user likely visited the web site (e.g. to use a cloud application), and occasionally show the HTTP parameters in the URL which can indicate things like user IDs and actions to be performed.
- *Cookies* - Most cloud applications use cookies to track user activity and facilitate ease of re-connection. The format of most cookies requires some decoding, but the formats are standard. Even encoded proprietary information such as user identifiers that may be meaningless when pulled off a device can be provided to the service provider in the warrant for interpretation and decoding to useful evidence.
- *Installed files and registry entries* - Installed component cloud applications, such as Dropbox [4], require the installed client software to communicate with the cloud application via the Internet. Presence of these installed components indicate that the cloud application was possibly used, and often contain configuration files and registry entries with specifics such as usernames, account names and types, IP addresses and port numbers of the service, and history of use. Some of these configuration file formats and registry keys may be encoded and proprietary, but they can be obtained and presented to the service provider in the warrant for decoding to useful evidence.
- *Modified files and registry entries* - Some web applications, like online storage applications, download files as part of their on-going operation. The modified, accessed and created dates of these files and how they were created (e.g. by a service) can be important in some investigations.

3.2 Identifying Cloud Signature Remnants

To identify cloud application remnant data sets we first identify which files the cloud applications typically install and/or access, then we use software to parse these files to extract specific data that is of relevance for forming warrants.

Determining Modified Files. We use two primary techniques for compiling a list of files that are modified by cloud applications: *hash sets* and *monitoring tools*.

Hash sets are lists of MD5 hash values of files that can be created by a software tool, such as AccessData's Forensic Toolkit (FTK) [30], scanning each file on the device and recording the MD5 hash value of that file. We used FTK to create a hash set of all files on a VMWare virtual machine (VM) that uses the target operating system (e.g. Windows 7). We then launch a cloud application on that virtual machine. We again take hash sets of the entire system after performing various tasks in the application including, but not limited to: connection, logon, using the application, and saving data on the cloud storage. Differences in the hash sets indicate the files that have been changed in the respective steps of using the cloud application. Although useful, this technique can yield a great deal of changed files, all of which have to be further inspected to ascertain their relevance, if any, to the signature of the application.

To refine the results of hash set monitoring, we used several special-purpose commercial monitoring tools. We used two categories of monitoring tools – *dynamic monitoring tools* and *static monitoring tools*.

Dynamic monitoring tools monitor an executing system and report live results. Our primary dynamic monitoring tool was SysInternals/Microsoft's Process Monitor [40], which is a tool that analyzes a process and reports on the CPU utilization, file I/O, Registry operations, network operations, and memory statistics. By observing its live reports during the various phases of using a cloud application, we were able to determine what system resources the cloud application used. For instance, we were able to observe the registry keys locked by Windows Live Mesh while it executed – the presence of these registry keys then being an element of the remnant data set for that application.

The primary static monitoring tool that we used for Windows systems is InCtrl5 [41] that monitors the state of the system before and after installation of software. We used InCtrl5 to establish what files installed-component software placed on the system, what registry entries they inserted, and what existing files they modified. Other tools such as Total Uninstall [42], and Spy Me [43] provide similar insight.

Parsing Files. The above techniques yield which files are added, and modified, by cloud applications. These files then need to be parsed to extract the specific data required by law enforcement.

Parsing requires searching the files for known keywords or substrings and then further processing of the text around those substrings for known parameters. For instance, when Google Docs is used on a Windows 7 computer via Internet Explorer, the substring *docs.google.com/document/create?* appears in an index.dat file and possibly the web history. The text following that string can contain the file name of the created document. In some circumstances a key string is first found and then subsequent searching and parsing is necessary. For instance, the presence of the string *docs.google.com/* in the index.dat file can indicate that the HTML code of the cached web sites on the device should be parsed for the specific HTML that Google Docs uses to display the username on all Google Docs pages. The parser then extracts the

username from the HTML code as a use remnant for law enforcement to use in their warrant.

Note that there is a substantial manual process that involves using these tools in a laboratory setting to pinpoint which changes to the system are made by the use of a cloud application (and which are not), and what relevant information those changes might yield. This research is the first step in establishing a process by which cloud remnant data sets can be added to a tool that will support law enforcement investigations.

4 Results

We constructed partial data sets for some important and representative cloud applications as a proof of concept. This section describes data sets for *Google Docs* as an example of browser-based applications, and of *Dropbox* and *Windows Live Mesh* as examples of installed component applications.

4.1 Browser-Based Cloud Applications

To demonstrate our techniques for searching browser-based cloud applications, we investigated Microsoft's Internet Explorer web browser versions 6, 7, 8, and 9 on Windows XP and Windows 7. All of these versions of Internet Explorer implement browser data in the *index.dat* file [44] structure from version 5. Overall, all of the Internet Explorer versions share the same *index.dat* structure, but the location of the temporary Internet files may change. Using the hash set technique of Section 3.2 we were able to determine the *index.dat* file used for Google Docs.

We wrote a parsing program to search the *index.dat* file to extract the last accessed time, the URL visited, and the filename based on keywords. The last accessed time indicates when the website was visited by the user. The URL visited stores the ASCII string representation of the URL that was visited, and the filename stores the name of the file that was accessed and temporarily stored by the browser on the physical device. The filename may not be set in all of the entries of an *index.dat* file because not all entries correspond to a file that is being temporarily stored on the computing device. As noted before, Internet Explorer versions 6, 7, 8, and 9 all implement the same data structure to store temporary Internet files. These applications implement Client URL Cache Version 5.

In every *index.dat* file there exists a 32-bit integer value at offset 0x20, which indicates where the entries are being stored on the file. Once the parsing program moves to the beginning of the entry storage, it starts to search for any valid entries.

An entry can be one of three types:

- REDR - a browser redirect.
- LEAK - an error that was generated. Usually this is generated due to an error occurring during the deletion operation of a URL entry.
- URL – URL that the user visited.

The parsing program only scans for entries marked LEAK or URL. It will not search for REDR entries because the final URL that the user would arrive at would still be indicated as a URL entry. After parsing the data, the parsing program moves to the URL and Filename offsets to parse the ASCII String representation of the bytes located in these data fields. After all of the data for the entry has been parsed, the parsing program determined if the entry has any evidentiary value. If the entry does contain evidentiary value, it is added to the set of data to report. The tool then continues searching for another iteration of a URL or LEAK entry.

Below in Figures 1 and 2, a URL entry is shown after it has been parsed by a custom WinHex template.

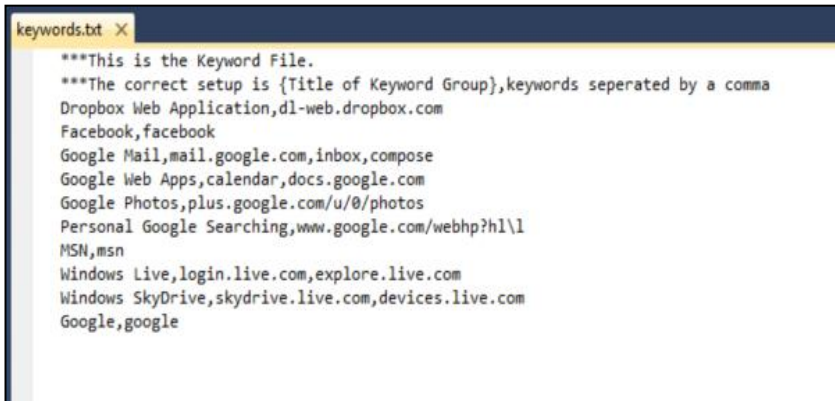
Offset	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	
00005100	55	52	4C	20	02	00	00	00	A0	78	09	E5	D2	ED	CC	01	URL x àÔiï
00005110	A0	78	09	E5	D2	ED	CC	01	51	42	70	02	00	00	00	00	x àÔiï QBP
00005120	97	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	
00005130	60	00	00	00	68	00	00	00	FE	00	10	10	8C	00	00	00	h p
00005140	01	00	10	00	00	00	00	00	00	00	00	00	00	00	00	00	
00005150	52	40	5B	02	04	00	00	00	00	00	00	00	52	40	5B	02	R@[R@[
00005160	00	00	00	00	EF	BE	AD	DE	43	6F	6F	6B	69	65	3A	6B	i%-bCookie:k
00005170	6F	70	70	65	6E	40	77	77	77	2E	6D	69	63	72	6F	73	oppen@www.micros
00005180	6F	66	74	2E	63	6F	6D	2F	00	BE	AD	DE	4D	4C	54	4C	oft.com/ %-pMLTL
00005190	30	46	37	4E	2E	74	78	74	00	BE	AD	DE	EF	BE	AD	DE	0F7N.txt %-pi%-p
000051A0	EF	BE	AD	DE	EF	BE	AD	DE	EF	BE	AD	DE	EF	BE	AD	DE	i%-pi%-pi%-pi%-p

Fig. 1. Hex view of Index.dat File URL Entry



Fig. 2. Parsed URL Bytes

To determine if a parsed entry is of evidentiary value, it is compared against a list of user-specified keywords, such as the following: *Google Documents, Google Mail, Google Plus, Personal Google Web Searches*. Figure 3 shows an example keyword file used by the parsing program.



```

keywords.txt X
***This is the Keyword File.
***The correct setup is {Title of Keyword Group},keywords seperated by a comma
Dropbox Web Application,dl-web.dropbox.com
Facebook,facebook
Google Mail,mail.google.com,inbox,compose
Google Web Apps,calendar,docs.google.com
Google Photos,plus.google.com/u/0/photos
Personal Google Searching,www.google.com/webhp?hl\l
MSN,msn
Windows Live,login.live.com,explore.live.com
Windows SkyDrive,skydrive.live.com,devices.live.com
Google,google

```

Fig. 3. Example Investigator Keyword File

The parsing program then searches each of the entries found in an `index.dat` file to determine if any of the keywords from the user generated `keyword.txt` file is present. If any keyword matches, then the entry is added as a result for the corresponding keyword group. Once the parsing of an entire `index.dat` file is finished, the entries that contained keywords are then written out to a Keyword Group in a report file formatted in HTML. The same steps are repeated for each `index.dat` file found, with a separate HTML report for each `index.dat` file.

When we applied this technique to the remnants left by the use of Google Docs, we have found that Google Docs leaves cached web sites that can determine the application and dates/times of use:

- *Start page:* -URL: <https://docs.google.com/> -Title: Google Docs
- *File listing:* -URL: <https://docs.google.com/#all> -Title: Google Docs - All items
- *Create:* -URL: <https://docs.google.com/document/create?hl=en> -Title: create
- *New document:* -URL: https://docs.google.com/document/d/1CjOwcXrET-uaFGPzalbNPS_P6kgQNhB1whAMprwNHXs/edit?hl=en -Title: Untitled document - Google Docs
- *Save as new name (test):* -URL: https://docs.google.com/document/d/1CjOwcXrET-uaFGPzalbNPS_P6kgQNhB1whAMprwNHXs/edit?hl=en# -Title: test - Google Docs

(where the long string in the URL parameter list is constant and likely an identifier that can be part of the warrant to Google.) These are just some sample cached files. Furthermore, the Google username is stored in an `index.dat` file as well as being available by subsequent parsing of some cached web sites.

We have monitored other browser-based applications such as iCloud and Zoho. Those two, for instance, leave cookies as well as Internet history.

4.2 Installed Applications

As noted before, a client-based cloud-computing application requires the user to install a piece of client software on a physical computing device. This paper reports our results on collecting artifacts from the *Dropbox* and *Windows Live Mesh* applications. Dropbox was chosen due to its immense popularity. In April 2011, Dropbox announced they had over 25 million users (Arrington, 2011). Windows Live Mesh was chosen due to the fact that it is supported by Microsoft, and is expected to be included in future operating systems (“BUILD, 2011”). Both of these allow for a user to upload files and folders via a client application. Then these files and folders can be downloaded on any other computer (“Dropbox”, “Windows Live Mesh 2011”).

Dropbox. The main focus of our investigation of Dropbox was on Dropbox SQLite database files that are typically present and contain data that would be of evidentiary value. These two database files are named *config.db* and *filecache.db*. The *config.db* database file contains the following information regarding the user’s Dropbox account:

- Dropbox Version - The version of Dropbox that is being used
- Unique Dropbox Host ID - A unique 128 bit key pertaining to a user account
- Dropbox Path - The path on the computational device where Dropbox has mounted its virtual folder
- Dropbox Username - A string username that is specified by the user
- Recently Changed Files - List of files that were most recently changed on the Dropbox account

	key	value	
1	config_schema_version		1
2	show_bubbles		1
3	fixed_dropbox_perms		1
4	recently_changed3	(lp1	
5	host_id	f1a8a2e7b15b0ba60d8474a55a2d6c3f	
6	dropbox_path	C:\Documents and Settings\Jeremy\My Documents\Dropbox	
7	root_ns		4827460
8	email	jeremy.koppen1@gmail.com	
9	stats_dont_send_until_upgrade		
10	stats_next_report_time		1332213286.078
11	stats_next_report_id		33916433
12	stats_build	S'Dropbox-win-1.1.35'	
13	ns_p2p_key_map	(dp1...	
14	sandboxes	(lp1	
15	last_update	(F1331003687.6400001...	

Fig. 4. Dropbox Config.db

The filecache.db database file contains a table called file_journal, which contains the following information for any files stored on the Dropbox account:

- Server Path - Stores the path of the file with the server identification
- Local Filename - Stores the local filename of the file that was added to the Dropbox account
- SHA-256 Hash - Stores the SHA-256 Hash in a Base-64 encoded string
- Local Size (MB) - Stores the Local Size of the file in MB
- Modified Time (UTC) - Stores the time that the file was modified
- Created Time (UTC) - Stores the time that the file was created

id	server path	local filename	local_blocklist	local size	local_mtime	local_ctime
61	4827460:/packetfilt...	Test Code Output.txt	Cq-EOPDpjhmPPG...	1685	1258497754	1329590553
62	4827460:/packetfilt...	Sample output.txt	rJ3DnT0C8ZRsoxi...	3632	1258497768	1329590553
63	4827460:/packetfilt...	PacketFilter1.vcproj	ORUNeRmC2GJQQ...	4540	1256241354	1329590553
64	4827460:/packetfilt...	stdafx.obj	KmSDgccE1f80Ze...	12601	1258413050	1329590553
65	4827460:/packetfilt...	PacketFilter1.pch	jAIOpC4rqJbe0s3...	3211264	1258413050	1329590578
66	4827460:/packetfilt...	mt.dep	3bbkxXd6JvUWaP...	66	1258497836	1329590553
67	4827460:/packetfilt...	PacketFilter1.suo	qjf4x1kFRgZdPn1...	8704	1258497840	1329590553
68	4827460:/packetfilt...	BuildLog.htm	TvqI0TSLK:kn9tig...	46118	1258497836	1329590563
69	4827460:/packetfilt...	PacketFilter1.cpp	SFccZh06NvyZxmi...	110602	1258497826	1329590564
70	4827460:/packetfilt...	PacketFilter1.exe	1L1HjBSduUQVAa...	159744	1258497836	1329590564
71	4827460:/packetfilt...	vc90.idb	_jHwVpaikDl7w1F...	183296	1258497836	1329590564
72	4827460:/packetfilt...	vc90.pdb	t53p7iIN7BMORQH...	282624	1258497834	1329590564

Fig. 5. File_Journal Table From File_Cache.db

Our parsing program scanned every user account described in the investigator's keyword list to determine the presence of a Dropbox directory. In Windows 7 the tool scans the user's AppData directory, and in Windows XP the tool scans the user's Application Data directory. If the Dropbox directory is found, then the software parses the config.db database file by using an instance of an SQLReader class that we programmed to parse SQLite databases. After the data in the SQLite database that matches keywords from the investigator's list has been determined, it is written out to a Dropbox Results HTML webpage report. Next, the software parses the filecache.db database file and sends all of the parsed data to the same Dropbox Results HTML webpage report. This information provides law enforcement unique user identification information, along with all of the files that the user had placed on the Dropbox account for storage.

Windows Lives Mesh. Windows Live Mesh is an application that allows a user to sync multiple folders, and subsequent files, within a supported Windows operating system to a cloud storage device, which is maintained by Microsoft.

Windows Live Mesh installs programs in the folder %PROGRAMFILES%\Windows Live. and in registry entries: HKCU\Software\Microsoft\Windows Live and HKLM\Software\Microsoft\Windows Live.

We found that there are several .edb database files used by Windows Live Mesh. These files are Extensible Storage Engine, JET Blue, database files. JET Blue was created by Microsoft and implements an Indexed Sequential Access Method (ISAM), data storage approach. In order to parse this database file the parsing program will use

the ManagedEsent .NET library. This library provides the ability to load the database and extract data.

Inside of this Windows Live Mesh directory there exists a directory titled “DB”. The “DB” directory contains one subdirectory called “Device” which contains Device.edb database file. In this file there is a User table that corresponds to the account information. The User table contains all of the user specific information such as: the email address of the account used, a unique ID, the last time the account was updated, the published date of the account, and the Name that corresponds to the user’s account. This table provides a wealth of information about the user’s Windows Live Mesh account. Once all of this table’s information is parsed, it will be printed out to a table inside of an .HTML webpage for a report to the law enforcement investigator.

The “DB” directory also contains subdirectories for each user account that was accessed, which are named from a unique user GUID. This GUID can be found in the User table from the Device.edb file. This directory contains an .edb database file that is also named based off of the unique user GUID. This file contains information corresponding to the user’s files.

Inside of this user GUID database file there are several tables. Our parsing software first accesses the MeshObject table, which contains fields called “Id”. These unique Ids correspond to directories that were synced with Windows Live Mesh. Windows Live Mesh creates tables for each directory that the user synced. The tool then parses each Id found in the Mesh Object in order to detect the names of the tables that contain information regarding the files that were synced. Once this Id has been parsed, the tool opens another table that is titled “{MeshObject Id}_DataEntity_Enclosure”.

Inside of this table there exists the following information for each file or directory added:

- Filename
- Parent History
- Creation Time (UTC)
- File Last Write Time (UTC).

5 Conclusion

Many informed predictions believe that cloud computing will become the predominant way that digital data is processed and stored. As such, it will necessitate changes in law enforcement policy and practice when performing investigations with digital evidence. The results reported here in the application of techniques to determine cloud remnants, and to parse those remnants for data that is required by law enforcement investigators is the first step in developing a tool to arm investigators against the looming threat that cloud computing poses - the threat of vast amounts of digital evidence not being available in the form that investigators have been trained to handle.

Based on the work reported here, we are developing a tool, called *Cloud Signature*, that performs the parsing described in Section 4 and generates reports that are easy for

law enforcement to use to fashion preservation letters and warrants. The tool will use the remnants described in this paper as well as remnants from other browsers (we are currently integrating the parsing of the Chrome browser remnants) and eventually remnants from mobile devices, specifically iOS and Android devices. Cloud Signature is being designed so that as the data sets for more cloud applications are determined, they can easily be added helping ensure that Cloud Signature keeps pace with the rapidly evolving landscape of the Cloud.

References

1. Mell, P., Grance, T.: The NIST Definition of Cloud Computing. Information Technology Laboratory (2009), <http://www.nist.gov/itl/cloud/upload/cloud-def-v15.pdf>
2. Amazon, Inc., <http://www.amazon.com/>
3. Apple iCloud, <http://www.apple.com/icloud/>
4. Dropbox, <http://www.dropbox.com/>
5. Rackspace Cloud, <http://www.rackspace.com/cloud/>
6. Salesforce, Inc., <http://www.salesforce.com/>
7. Skytap, Inc., <http://www.skytap.com/>
8. Microsoft Corporation, Windows Live, <http://explore.live.com/>
9. Google, <http://www.google.com/>
10. Huawei Technologies Co., Ltd., <http://www.huawei.com/>
11. Cisco Systems, Inc., <http://www.cisco.com/>
12. Fujitsu, Ltd., <http://www.fujitsu.com/global/>
13. Dell, Inc., <http://www.dell.com/>
14. Hewlett-Packard Development Company, L.P., <http://www.hp.com/>
15. IBM, <http://www.ibm.com/>
16. VMware, Inc., <http://www.vmware.com/>
17. Hitachi, Ltd., <http://www.hitachi.com/>
18. NetApp, Inc., <http://www.netapp.com/>
19. Security Guidance for Critical Areas of Focus in Cloud Computing V2.1. Cloud Security Alliance (2009)
20. Mather, T., Kumaraswamy, S., Latif, S.: Cloud Security and Privacy: An Enterprise Perspective on Risk and Compliance, p. 239. O'Reilly Media (2009)
21. Federal Information Security Management ACT (FISMA) Implementation Project, <http://csrc.nist.gov/groups/SMA/fisma/index.html>
22. Health Information Privacy: The Health Insurance Portability and Accountability Act (HIPPA), <http://www.hhs.gov/ocr/privacy/>
23. The Sarbanes-Oxley Act, <http://www.soxlaw.com/>
24. PCI SSC Data Security Standards Overview, https://www.pcisecuritystandards.org/security_standards/index.php
25. SAS 70 Definition: Type II, <http://www.sas70.us.com/what-is/definition-of-sas70.php>
26. Kirilov, K.: Cloud Computing Market Will Top \$241 Billion in 2020. Cloud Tweaks (2011), <http://www.cloudtweaks.com/2011/04/cloud-computing-market-will-top-241-billion-in-2020/>

27. Columbus, L.: Roundup of Cloud Computing Forecasts and Market Estimates. A Passion for Research (2012),
<http://softwarestrategiesblog.com/2012/01/17/roundup-of-cloud-computing-forecasts-and-market-estimates-2012/>
28. IDC Cloud Research, http://www.idc.com/prodserv/idc_cloud.jsp
29. Bigsey, Cloud Computing and the Impact on Digital Forensic Investigations. ZDNet (2009), <http://www.zdnet.co.uk/blogs/cloud-computing-and-the-impact-on-digital-forensic-investigations-10012285/>
30. Access Data FTK, <http://accessdata.com/products/computer-forensics/ftk>
31. EnCase Forensic v7, <http://www.guidancesoftware.com/encase-forensic.htm>
32. X-Ways, <http://www.x-ways.net/>
33. ATC P2P Marshal, <http://p2pmarshal.atc-nycorp.com/>
34. ATC Mac Marshal, <http://macmarshal.atc-nycorp.com/>
35. ATC Cyber Marshall Dropbox Reader, <http://www.cybermarshal.com/index.php/cyber-marshal-utilities/dropbox-reader>
36. Gargoyle Investigator, <http://www.wetstonetech.com/cgi-bin/shop.cgi?view,2>
37. NetAnalysis, <http://www.digital-detective.co.uk/netanalysis.asp>
38. JADsoftware's Internet Evidence Finder, <http://www.jadsoftware.com/internet-evidence-finder/>
39. RedLight, <http://www.dfc.cs.uri.edu/redlight.php>
40. Process Monitor v3.01, <http://technet.microsoft.com/en-us/sysinternals/bb896645>
41. PCMag InCtrl5, <http://www.pcmag.com/article2/0,2817,25126,00.asp>
42. Total Uninstall, <http://www.martau.com/>
43. Spy Me, <http://www.ghacks.net/2009/03/01/software-installation-monitor/>
44. Jones, K.J.: Forensic Analysis of Internet Explorer Activity Files (2003), <http://www.mcafee.com/us/resources/white-papers/foundstone/wp-pasco.pdf>