Digital Investigation 24 (2018) S68–S75

Contents lists available at ScienceDirect

Digital Investigation

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

DFRWS 2018 Europe — Proceedings of the Fifth Annual DFRWS Europe

Forensic framework to identify local vs synced artefacts

Jacques Boucher [a, *], Nhien-An Le-Khac [b]

[a] Ottawa, Ontario, Canada
[b] University College Dublin, Belfield, Dublin 4, Ireland

Keywords:
Data synchronization
Forensic framework
Google chrome
Data synced forensics

A B S T R A C T

Today, application developers strive to make a user's experience seamless as they move from one device to the next by synchronizing the user's data between the devices. With the ever-increasing proliferation of Internet connected devices we can expect to see greater integration and synchronization between these devices. The end user benefits of this seamless synchronization of data between devices. The synchronization of data between devices translates to both a benefit and a challenge for computer forensic examiners. The benefit is that the device being analyzed may contain evidence that synced from another device that cannot be found. The challenge for a computer forensic examiner is that the device being analyzed may contain evidence that synced from another device. In most jurisdictions police must prove mens rea, the intention or knowledge of wrongdoing. It is a challenge for examiners if a user claims that the evidence found on their laptop was created by an unknown user on another device, and that activity synced to their laptop. There is very little research on synchronization of data between devices in literature. Therefore, in this paper, we propose a framework to guide computer forensic examiners in their quest to determine if data is local or synced. We also demonstrate the application of our framework on a known scenario to evaluate the confidence an analyst can attribute to each section of the framework, and caveats that need to be considered when forming an opinion on whether data is local or synced.

Introduction

There was a time when people owned one device to connect to the Internet, a PC (desktop or laptop). All user data relating to their Internet activity (whether it was browsing, email, or chatting) was stored locally on that single PC. All evidence relating to the user's online activity could be found on their PC.

As more devices penetrated our lives, such as mobile phones and tablets, it became cumbersome for a user to migrate their browser bookmarks for example from one device to the next with the support of Cloud computing platform (Dykstra, 2013; Faheem et al., 2015; Ryder and Le-Khac, 2016). As our online habits increased, the demand grew for seamless synchronization of a user's data between their devices. In the early days of syncing data, what did sync was relatively limited, and in some cases, it was not overly convenient. In the earlier days of Mozilla Firefox' sync feature a user had to create an account online and then authorize their devices that they wanted to allow to sync. It was a bit cumbersome and thereby discouraging mass adoption. Today syncing is near seamless. For instance, you can log into your Google account on Google Chrome browser and it will automatically sync data between your device and your data in the cloud. You can also surf on one device, and walk over to another device authenticated to the same Google account and continue where you left off. We can see the same seamless syncing of data in the Apple ecosystem (Carlson and Moren, 2015).

The sync feature commonly offered on modern end user applications benefits end users, and presents both benefits and challenges to a computer forensic analyst. One benefit to a computer forensic analyst is that if a user has one device securely locked down, you may still be able to see what they did via what synced to another device that the forensic examiner is able to access. Likewise, if a device is not found during a search, an examiner may still find activity from that device synced to another device found at the location being searched.

Despite its benefits, syncing also presents a challenge. How do you prove that the activity happened on the device being analyzed and that it did not sync from another device? Surprisingly this challenge does not appear to have been raised in the courts as a

* Corresponding author.
E-mail address: jjrboucher@gmail.com (J. Boucher).

https://doi.org/10.1016/j.diin.2018.01.009

defense, and therefore is not being addressed by examiners as part of their forensic process. With the ever-increasing number of end user devices syncing data between them, it is only a matter of time that a computer forensic analyst will face that defense in court. It may even be a legitimate defense. Imagine a computer forensic analyst analyzes a laptop computer and finds URLs in the browser history showing that the user surfed to websites that corroborate that the user is guilty of an offense. What if that user has a tablet at home that is automatically connecting to their browser account, and that someone at home surfed those sites? Would an examiner know this to be the case? Would they be able to defend their evidence against such a defense? Given the absence of research on this topic, it suggests that few computer forensic examiners are considering this during their examination. Therefore, in this paper, we present a framework that guides computer forensic examiners in their analysis to answer the question, "Is this a local artefact or did it sync from another device?". The lack of attention to this issue creates a risk that a computer forensic examiner will face this argument in court some day and will be unprepared and thus unable to address it. This is not something that will go away in time. With the Internet of Things, Cloud data storage finding their way into more aspects of our lives, the synchronization of data between devices will become more common, more seamless, and more fragmented across a larger number of devices (Hale, 2013; Alabdulsalam et al., 2018).

Our proposed framework is composed of four steps: application analysis, operating system analysis, timeline analysis and forming an opinion. The framework is independent of the operating system and the application. In the experiments to evaluate its performance, authors went with Microsoft Windows 10 because there are several mature computer forensic tools to process artefacts for that OS. Google Chrome was selected as the test application as it is also readily supported by our computer forensic tools.

The rest of this paper is organized as follows: Section Related work shows the related work of this research. We present the challenges of the framework in Section Problem statement. We describe our approach in Section Adopted approach. We evaluate the framework in Section Evaluation. We also discuss and analyze its performance in this section. Finally, we conclude and discuss on future work in Section Conclusion and future work.

Related work

Various applications have incorporated the ability to sync a user's data between devices. Browsers such as Google Chrome and Mozilla Firefox have provided the ability to sync data either natively or via an add-on for some time now. For instance, a Google Chrome release note advises users that Omnibox History has been added as a sync data type (Grunberg, 2011). Based on this we know that Google Chrome has natively supported syncing of data since sometime prior to that date. Yet even though this feature has existed for several years, attempts to find prior research on this topic has yielded negative results. There does not appear to be any writings online relating to how a computer forensic analyst should conduct their analysis to establish whether data found on a device originated from that device, or if it synced from another device.

There are a number of cloud services that allows data to sync between devices. iCloud is one such service designed to keep data synchronized between Apple devices. In (Friedman et al), authors looked at how a computer forensic examiner can determine if a device is configured to sync to an iCloud account. They also raised the challenge of trying to gather the evidence stored in an iCloud account. The study did not look at which Apple devices were responsible for the data in iCloud.

Today, the Internet of Things (IoT) adds a layer of complexity when it comes to analyzing devices with cloud evidence, especially as it relates to the application of the framework proposed in this paper. IoT devices have different operating systems and filesystems, and because of its ubiquitous syncing of data between devices, there is less certainty in where the data originated (Schatz et al., 2014; Lillis et al., 2016).

In (Shavers, 2013), Shavers talks about creating timelines with either *log2timeline* or with a forensic tool, and visualizing it with a visual tool or a spreadsheet application. Shavers also made an important statement about timelines. Indeed, he offered some helpful advice in how to organize timelines that come from multiple sources/devices, as well as adding events to the timeline from other sources (i.e. the time an employee checked in at work, when they accessed a room based on their access card, etc.). He focused on the issue of having to cull and analyze the volume of data on various storage devices. However, he fails to make any reference to the challenge that will result from attributing data to a specific device because of data synchronization across a user's multiple devices. Indeed, data duplication across multiple devices is a challenge as Shavers pointed out. But an even bigger challenge that we are tackling in this paper is on which device was the data created? What value is there in placing an individual behind the keyboard of device A if the data was created on device B and synced to device A? Shavers does not raise the challenge of trying to determine on which device the data was created is consistent with what we have seen in the computer forensic community. It is something that computer forensic examiners are not even considering in many cases.

In terms of Google Chrome investigation, Wright (2015) explored Google Chrome artefacts and how they can be attributed to a specific device. One notable difference in Wright's research and our experiments is that he ascertained how to attribute activity in Chrome to specific devices using the information in *SyncData.sqlite3*. Wright's research provides a computer forensic analysis with a possible template to use when mapping out application artefacts to apply the framework.

Problem statement

As outlined in the introduction, in today's connected world it is more likely that a computer forensic examiner will find evidence in user data originating from applications that can sync user data between devices. This presents a problem for the computer forensic examiner attempting to ascertain whether user artefacts recovered from a device were created by a user on that device, or if they were created on another device and synced to the device being analyzed.

The challenge that arises from trying to form an opinion is even more daunting because of the multitude of user applications out there, on numerous different platforms (i.e. desktops, laptops, mobile, wearable, gaming), and on a variety of consumer operating systems (i.e. versions of MS Windows, Mac OS, Android, iOS, Blackberry). The absence of a credible repository of forensic documentation means that in many cases, the computer forensic examiner will have to research an application that may have already been researched by another peer in the international computer forensic community.

The computer forensic examiner will have to be disciplined in identifying what research is needed to allow them to form a defensible opinion. It will be important for the computer forensic examiner to avoid falling in the trap of going too deep in their research and focusing unnecessarily on aspects of the OS or the application that does not help them form an opinion about the artefact in question found on a device being analyzed.

At its core, the question that needs to be answered is quite basic. Did the artefacts found during an analysis of a device originate from that device, or did it sync from another device? The difficulty in answering this question will depend on several factors. Are the OS artefacts and application artefacts needed to answer that question properly documented? Is there a reliable tool to create a timeline of activity on the device? Our proposed framework will guide a computer forensic examiner in attempting to answer the question, was it synced or not.

Adopted approach

In order to answer the question whether specific application user artefacts were produced locally on a device or if they were synced from another device, we propose a framework consisting of four steps: (i) application analysis; (ii) operating system analysis; (iii) timeline analysis; (iv) forming an opinion (Fig. 1).

Step 1 – application artefact analysis

A computer forensic analyst first needs to have a proper understanding of the application artefacts. If existing documentation exists, the computer forensic analyst can use that as a starting point and ensure it applies to the version of application on the device they are analyzing. In absence of such documentation, the computer forensic analyst will have to invest time researching the application – where artefacts are stored, and sync features.

The first step of the framework is to examine the application artefacts for any indicators that will assist the computer forensic examiner in forming an opinion (Fig. 2). For example, if a URL is found in a browser's history, the computer forensic analyst would analyze other browser artefacts relating to the URL in question that can serve as indicators of synchronization (i.e. absence of cookies or cache associated to the URL in question, or if the application specifically tracks which artefacts are local and which are synced) (Oh et al., 2011; Warren et al., 2017).

It is equally important to determine what syncs as well as what does not sync. If a user artefact of interest can sync (Google Chrome browsing history, for instance), the computer forensic examiner will need to work through the framework to attempt to determine if it's present on the device because it synced from another device, or if was created on the local device. If the user artefact of interest never syncs (Google Chrome Cache, for instance), this framework is not needed as the artefact can only be local.

Step 2 – OS artefact analysis

A computer forensic analyst must also have a good grasp of the forensic artefacts of the operating system. What is tracked by it, and where it stores that information? So, the second step of the framework is to examine OS artefacts for indicators that will help the computer forensic examiner form an opinion whether the URL of interest, for example, was browsed locally or if it synced from another device (Fig. 3). This part of the framework is particularly useful to help form an opinion that data is synced. But it is less helpful to reach a conclusive opinion that data is local.

If a URL of interest was browsed while the system was not powered on, or no user was logged in, it stands to reason that the data cannot be local as the device was off or no user was logged in at that time. But the reverse scenario is less conclusive. If the URL of interest was browsed while the system was powered on and a user was logged in, the computer forensic examiner cannot support a conclusive opinion either way as it's possible that the data is local, or synced.

Fig. 1. High level of proposed framework.

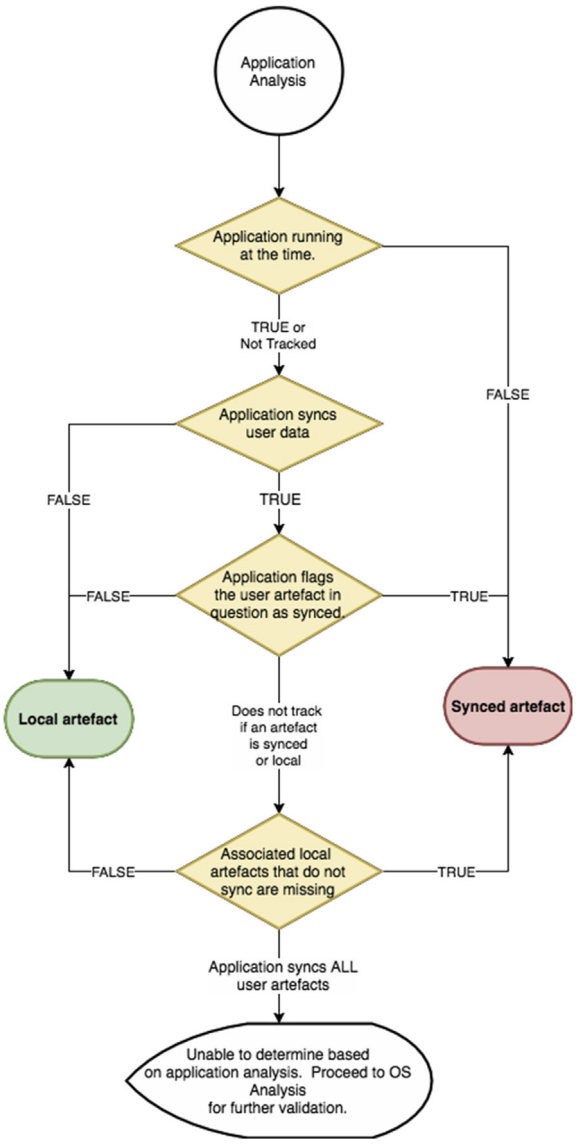

Fig. 2. Application analysis.

A computer forensic examiner will need to determine if the OS on the device tracks when a device powers on/off, when a user logs in/out, when a device goes into hibernation/sleep or comes out of it, when the screen saver/lock screen is engaged/disengaged, when connectivity is established/broken. These are important OS artefacts that can contribute to forming an opinion on whether an artefact is local or synced.

Step 3 – timeline analysis

The last piece of research which serves to bring the first two pieces of the framework together is timeline analysis. There are obvious scenarios that allows a computer forensic examiner to conclude that data was synced from another device (Fig. 4). For

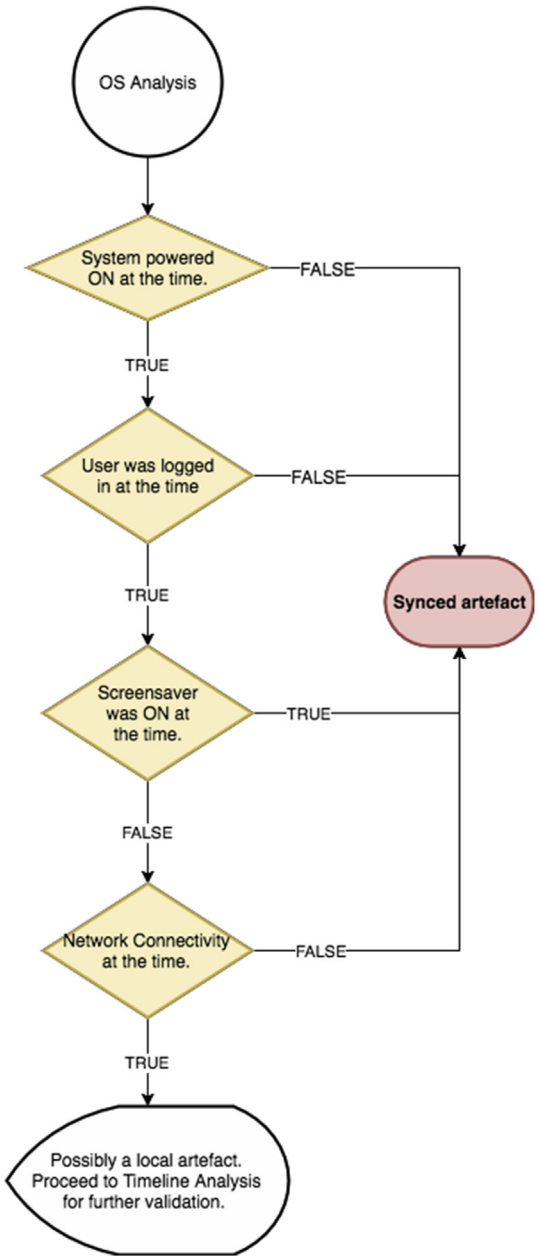

Fig. 3. OS analysis.

example, if the browsing activity took place during a time frame when the computer was not even powered on, a computer forensic analyst can form an opinion that the data was synced form another device. The most efficient way to identify this is through a visual timeline analysis. Therefore, the third step of the framework is a timeline analysis. A visual timeline analysis can allow a computer forensic examiner to see that at the time of an artefact of interest, the timeline activity was otherwise flat, suggesting that no other activity was happening on the system at the time. With an OS like MS Windows, there is always activity happening in the background.

The lack of activity in the timeline can corroborate the opinion that the activity was likely synced from another device. This is where the timeline complements the OS analysis. If the OS analysis revealed that the activity could have been local or browsed (i.e. system was powered on, user logged in, connectivity, etc.), the timeline analysis can be used to visually see how busy the system

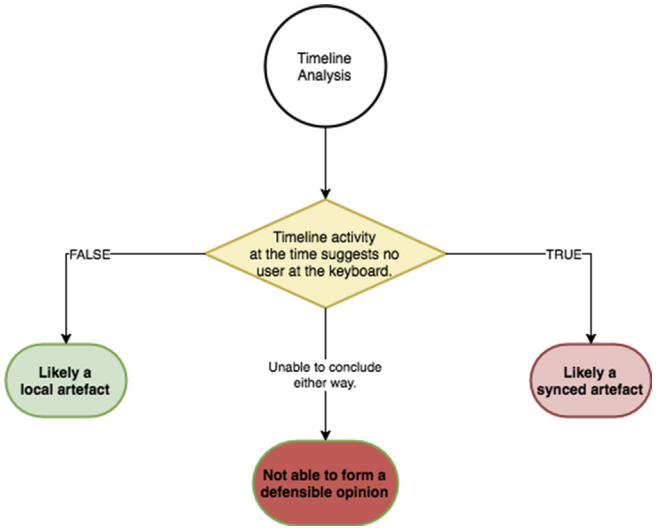

Fig. 4. Timeline analysis.

was at that time. From experience, we know that when a user is at the keyboard, there is a lot of noise in the timeline. Therefore, a relatively flat timeline around the time a URL was surfed would suggest no user was at the keyboard.

The timeline analysis is good to help form an opinion, but a computer forensic examiner should not rely solely on the timeline to reach an opinion. It's a great visual tool to quickly start forming an opinion, and may be the first one some examiners will use in the framework. There are caveats to be aware of when using timeline analysis. The timeline is only as good as the tool that created it. If a tool does not support certain artefacts (application artefacts, OS artefacts), those artefacts will be missing from the timeline. If at some point the time was off on the system, it will throw off the timeline. If a tool parses a time as local when it's UTC, it will affect the timeline.

Step 4 — forming an opinion

The goal of this framework is to form an opinion on whether a particular application user artefact on a device being analyzed was created on that device, or if it synced from another device. The forensic examiner needs to assess the confidence that they can attribute to their opinion and frame it accordingly. As was seen in the first three steps, in some instances a computer forensic examiner may be able to form a definitive opinion that something was synced. In other cases, they may be able to form a definitive opinion that something is local. Those are the easy ones. The challenge will arise when the forensic evidence available to the computer forensic examiner does not allow for a conclusive opinion either way. In these challenging circumstances, it will become incumbent on the computer forensic examiner to apply as many aspects of the framework as possible to increase the level of confidence in their opinion (and consequently, the defensibility of it in court).

An analyst may not need all three steps in the framework to form an opinion. For example, if the application tracks what is local and what is synced, an analyst will be able to rely on those facts to determine which artefacts are local and which ones are synced. In some cases, an analyst will not be able to conclusively state whether data is local or synced. In those cases, the framework will help the analyst attribute a level of confidence to their opinion that can be defended in court.

Discussion

The framework proposed in Fig. 1 is straight forward on the surface, and can be used by a computer forensic examiner to form an opinion whether a particular artefact of evidence was produced on the device being examined or if it was synced from another device. Depending on the specific circumstances encountered by a computer forensic examiner, it's possible that all three steps of the framework will be required to form a defensible opinion whether an artefact is local or if it synced from another device. In other cases, a computer forensic examiner may only need to apply one of the steps of the framework to form a defensible opinion.

A computer forensic examiner may need to use this framework to validate key pieces of digital evidence if the application that created it has a feature that syncs user data between devices. If the application is not designed to sync data, there is no need to conduct any further analysis to determine if the user artefacts from that application are local or synced from another device. It stands to reason that in absence of an application's ability to sync data, the application data is local.

If the application requires a user to sign in each time to use it (i.e. it does not auto-login), that may be sufficient to support the findings that the owner of that account is the author of the digital evidence that was found. In this case, it may be immaterial on which device the activity took place. The fact that it was done by a specific user may be sufficient evidence in court to associate that activity to a specific individual.

The application of the framework is by no means a trivial task if dealing with an OS or an application that lacks documentation. Different OS' track different user activity, at different locations in the filesystem, and in different formats. The same applies to applications. To add to that complexity, some applications can drastically alter features or how it stores data from one version to the next. Mozilla Firefox for example moved from a cryptic file system to SQLite for user data when it went from version 2 to version 3, rendering all prior research obsolete. Both Google Chrome and Mozilla Firefox use a rapid release cycle with updates being released every 6 weeks. Features within an application can change from one release to the next. As the work in this field matures, computer forensic examiners will be able to draw on prior research to help facilitate the application of the framework to different operating systems, and different applications.

Evaluation

Application analysis

We use Google Chrome to demonstrate the process an examiner will need to undertake in analyzing the application.

Through a series of test scenarios, we confirmed that Chrome can sync user data between devices providing that Chrome is logged into a Google Account (accessible via Settings menu). This is important as a user must log in to allow data to sync. It is important to point out that a user can surf without being logged in, log in to allow syncing, and then log back out.

Google Chrome tracks this status in a JSON file called 'Local State' that can be found at *%localappdata%|Google|Chrome|User Data|* on a MS Windows 7/10 system (Fig. 5).

Google Chrome uses several SQLite files to store user data. They can be found in a user's profile folder found at this same location. The SQLite file containing browsed URLs is stored in a file called 'history'. This SQLite file has multiple tables, including one called URLS where the actual visited URLs are stored. A second table called VISITS tracks the details of the visits (i.e. date/time, the previous URL in the navigation sequence, the URL visited, the transition (i.e. typed URL, clicked on a link)). No actual URLs are stored in this table. Rather a record in visits will point to a record in URLS. This avoids having to store the same URL several times. A unique URL is stored only once in URLS, and referenced as needed in VISITS.

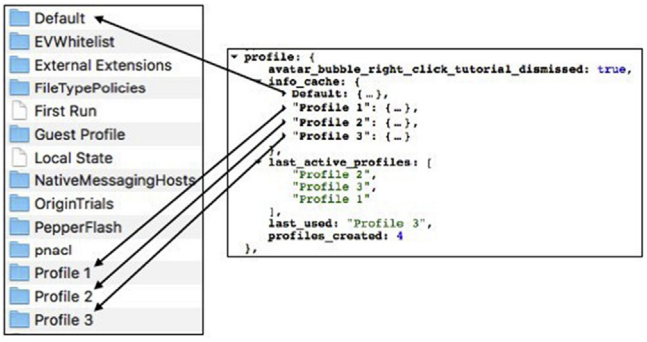

Fig. 5. Local state.

A third table called VISIT_SOURCE tracks whether a URL is local, or if it synced from another device. This table contains only two fields, the ID which corresponds to the ID in the URLS table, and SOURCE. Only synced URLS had an entry in VISIT_SOURCE. Using a SQLite statement with joins in it to link these three tables together, a computer forensics examiner can easily determine if a URL is synced, or local.

As we see in Fig. 6, Chrome makes it easy for an examiner. But what if it didn't track that? How might we form an opinion? We'd have to dive deeper into its artefacts. Chrome will only sync typed URLs, and URLs that a user clicked on if that URL was also typed. It tracks this information in the transition field in the URLS table. The transitions field is a four-byte value. The right most byte decodes to the transition type (i.e. typed URL, clicked on a link, form submit, etc.) The next three bytes decode to a transition qualifier (i.e. navigated via the forward/back arrow, redirect, etc.).

Through testing, we know that Chrome does not sync cookies or cache. So even without Chrome tracking what syncs and what does not, if an examiner finds a URL of interest in Chrome with corresponding cookies and cache entries, that URL is a local URL. Conversely if a URL exists but no corresponding cache or cookies exists, there is a very good chance it's a synced URL (we have to recognize that a user can selectively delete cookies, and could clear their cache).

For this scenario, we will use URL #10, http://www.bmw.ca, as our artefact of interest. When we examine Chrome's cookies for that period of time, no entries are found for bmw.ca. Likewise, when we examine Cache, we note that there is no corresponding browser cache activity for that time frame for bmw.ca.

Because of the level of detail tracked by Google Chrome, a computer forensic analyst can successfully form an opinion using only the "application analysis" portion of the framework. Google Chrome tracks which browser history entries are synced (and therefore which ones are local, Fig. 3).

This part of the framework is especially reliable when the application that created the user artefacts of interest meticulously tracks what data has synced and what has not (as Chrome does), or if certain artefacts never synchronize (as was demonstrated using Cookies and Cache) allowing the computer forensic examiner to be able to form the opinion that the data had to be local given testing showed it never syncs.

OS analysis

We use Microsoft Windows 10 to walk through this part of the framework. Microsoft Windows 10 is Microsoft's current flagship

visit	url	Visit Source	transition	Transition Type	visit_time	Decoded visit_time (UTC)
1	http://www.cnn.com/	Synced	838860801	Typed URL	13134336170050200	2017-03-18 18:42:50
3	https://www.google.com/	Synced	268435457	Typed URL	13134336385628800	2017-03-18 18:46:25
2	https://www.google.ca/?gfe_	Synced	-1610612735	Typed URL	13134336385628800	2017-03-18 18:46:25
6	https://www.google.ca/?gfe_	Synced	805306368	Clicked on a link	13134336556735900	2017-03-18 18:49:16
4	https://www.google.com/	Synced	268435457	Typed URL	13134336556537400	2017-03-18 18:49:16
5	https://www.google.ca/?gfe_	Synced	-1610612735	Typed URL	13134336556537400	2017-03-18 18:49:16
7	http://www.youtube.com/	Synced	268435457	Typed URL	13134336668854300	2017-03-18 18:51:08
8	https://www.youtube.com/	Synced	-1610612735	Typed URL	13134336668854300	2017-03-18 18:51:08
9	http://www.bmw.ca/en/hom	Synced	-1610612735	Typed URL	13134337619102600	2017-03-18 19:06:59
10	http://www.bmw.ca/	Synced	268435457	Typed URL	13134337619102600	2017-03-18 19:06:59

Fig. 6. Browser history.

consumer desktop operating system. This won't be a deep dive into MS Windows artefacts. But it will be sufficient to provide an overview of how the framework is applied.

In the case of Windows 7/10, it tracks OS events in the various Windows event logs. An analyst can query those event logs to determine whether a user could have been using the device at the time an application artefact of interest was created (i.e. when a site was browsed). If the device had no connectivity at the time of the browsed URL, it stands to reason that the URL had to have been browsed on another device and synced to this one once connectivity was established. If the logs reveal that a user was at the keyboard and could have surfed that URL, an analyst can only conclude that the URL could have been browsed locally, or it could have been synced.

In Fig. 7 the Windows event logs reveal that the system was powered on at the time, but no user was logged in. The Internet history activity in the timeline in Fig. 8 has no corresponding cache activity. This further corroborates that the artefact in question had to be synced.

The strength of the "Operating System Analysis" portion of the framework is to provide the computer forensic examiner with defensible evidence when something was synced. If the device is powered off, or if no user is logged in at the time a user artefact was created, the artefact has to have been created on another device and synced to the one being analyzed. This part of the framework can also help corroborate an opinion that an artefact was local by showing that the system was powered on and a user was logged in at the time of the artefact. But a computer forensic examiner would not be able to rely solely on this fact to form an opinion that the artefact was created on the local device. A device can be powered on with a user logged in and actively browsing at the same time as browsing is taking place on a second device that is syncing to the first, co-mingling the synced data with the local data.

Attempting to form the opinion that all the artefacts in that scenario are local because the OS was powered on and the user was logged on at the time would clearly be incorrect.

Timeline analysis

The authors created different scenarios for timeline analysis. All timeline scenarios were created on a Windows 10 Virtual Machine (running in VMWare Fusion on an iMac computer), with the time zone set to EST. All times referenced in scenarios will be in EST unless otherwise noted. The authors used X-Ways Forensics (Shavers and Zimmerman, 2014) version 18.9. A refined volume snapshot was applied that included extracting browsing activity (given that is the application we examined earlier). X-Ways has a timeline feature that was used against the refined snapshot. The timeline activity was filtered by date/time to provide only activity for the time frame of the scenario plus a short time frame before and after the scenario. Due to the limitation of number of pages, we only describe one scenario in this section.

In this scenario, the surfing was done on the Mac on 2017-03-18 between 14:35 h and 15:09 h. The VM was powered on at that time, but no user was logged in. We actively surfed on the Mac, and subsequently logged in the VM, launched the browser and allowed browsing activity to sync to the VM. In this scenario, XWays extracted 180,329 events in the timeline for the slice of time being examined.

Of the 180,329 events, there were 11,778 unique time stamps. As was done in the other scenarios, the time stamps were rounded, resulting in 29 unique clustered time stamps to plot along the X axis (vs trying to plot 11,778 events along the X axis). The surfing was done on the Mac on 2017-03-18 between 14:35 h and 15:09 h. The VM was powered on at that time, but no user was logged in.

A number of graphical timelines were created. Because we are dealing with a scenario where the system was powered on, it resulted in a lot of noise in the timeline. That made it challenging to reach an opinion on whether a user was at the keyboard or not. Advanced analytics (outside the scope of this paper) might provide insight to support an opinion.

In this scenario, the authors excluded some of the events in order to zoom in on Internet activity as noted in Fig. 9. By doing so, it yielded a cleaner graphical timeline that showed very minimal Internet activity during the period. This is contrary to what was observed during normal browsing in other scenarios where local Internet activity typically yielded 50 + events for a data point on the X axis.

Graphical timeline analysis can be very effective in quickly identifying scenarios where synced activity had an activity time-stamp when the system was powered off. In some of the more

(128)	2017-03-18 15:24:17	Kernel-General	13	(2)
Audit Success	2017-03-18 15:18:46	Microsoft Windows secu...	4647	Logoff
Audit Success	2017-03-18 15:09:41	Microsoft Windows secu...	4624	Logon
Audit Success	2017-03-18 15:09:41	Microsoft Windows secu...	4624	Logon
Audit Success	2017-03-18 15:09:41	Microsoft Windows secu...	4648	Logon
(128)	2017-03-18 14:44:07	Kernel-General	12	(1)
(128)	2017-03-18 14:43:49	Kernel-General	13	(2)
(128)	2017-03-18 14:39:23	Kernel-General	12	(1)

Fig. 7. Windows event logs.

Timestamp	Category	Description	Name
2017-03-18 14:06:58	File system		amd64_microsoft-wi
2017-03-18 14:06:58	File system		wow64_microsoft-wi
2017-03-18 14:06:58	File system		aepic.dll
2017-03-18 14:06:58	File system		aepic.dll
2017-03-18 14:06:59	File system		twinui.dll
2017-03-18 14:06:59	File system		twinui.dll
2017-03-18 14:06:59	File system		amd64_microsoft-wi
2017-03-18 14:06:59	File system		dbgeng.dll
2017-03-18 14:06:59	File system		dbgeng.dll
2017-03-18 14:06:59	File system		amd64_microsoft-hy
2017-03-18 14:06:59	File system		vmusrv.dll
2017-03-18 14:06:59	Internet	http://www.bmw.ca/en/home.html (title: Hc	History
2017-03-18 14:06:59	Internet	http://www.bmw.ca/ (title: Home)	History
2017-03-18 14:06:59	File system		StartUI.dll
2017-03-18 14:06:59	File system		StartUI.dll
2017-03-18 14:06:59	File system		77b0f2e54a52a84bb
2017-03-18 14:06:59	File system		e49c953bbc1a8845b
2017-03-18 14:06:59	File system		77b0f2e54a52a84bb
2017-03-18 14:06:59	File system		77b0f2e54a52a84bb
2017-03-18 14:06:59	File system		e49c953bbc1a8845b
2017-03-18 14:06:59	File system		e49c953bbc1a8845b
2017-03-18 14:06:59	File system		amd64_microsoft-hy
2017-03-18 14:06:59	File system		vmswitch.sys

Fig. 8. Timeline activity.

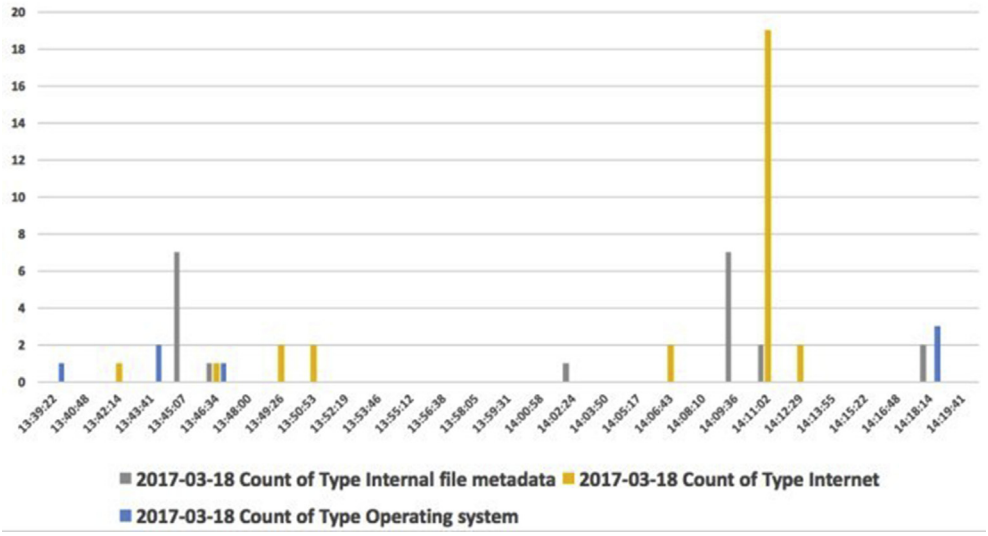

Fig. 9. Graphical timeline–internet activity (yellow). (For interpretation of the references to color/colour in this figure legend, the reader is referred to the Web version of this article.)

active scenarios as presented herein, its effectiveness is reduced. In some cases, looking at the raw timeline is sufficient to form a defensible opinion based on the knowledge gained about the application artefacts in the previous part of the framework (i.e. lack of Google Chrome cache activity in the raw timeline around the time when a site is visited).

All three parts of the framework (Google Chrome artefacts, Windows 10 event log artefacts, and Timeline analysis) all corroborate that the device was powered on at the time the browsing took place but no user was logged in, thus it could not have been browsed on the device being examined. It is synced artefacts that came from another device (the Google Chrome session on the Mac in this case).

Discussion

The sync defense is not automatically fatal to a case even where a computer forensic examiner did not consider it at the time of the analysis. It will allow however defense to raise an issue in court that will be harder to address if the computer forensic examiner did not consider it during the analysis of the digital evidence. In addition, the sync defense could also be used to attack the credibility of the computer forensic examiner by arguing that the examiner ignored possible exculpatory evidence by neglecting to look at whether the evidence was local, or synced from another device.

Each of these three steps of the framework contributes to forming an opinion on whether a particular application user

artefact is local or synced. As was demonstrated in this paper, the application analysis has the most potential to provide evidence to defend one's opinion on whether an application artefact is local or synced. If an application is tracking what is local and what is synced, a defensible opinion can be formed without any further application of the framework.

The OS analysis' strength is in forming a defensible opinion that an artefact is synced. It doesn't afford the same degree of confidence if an artefact is believed to be local. The timeline analysis' strength is visually identifying the volume of activity around the same time as when an application artefact of interest was created. This helps form an initial opinion that should be validated using the application analysis and OS analysis parts of the framework.

The highest confidence is obtained when the three analysis steps of the framework (steps 1–3) are applied to form an opinion.

There are countless applications that can sync data that could be researched and added to a repository for computer forensic examiners to use to assist them with applying this framework. Apps that synchronizes user data are especially prevalent in the mobile device market. There is excellent computer forensic documentation for a variety of applications being produced by computer forensic examiners. But the current practice does not include documenting how application artefacts differ when local vs synced.

Apps on mobile devices may track more granular usage of the app that could allow a computer forensic examiner to cross reference user artefacts to whether the app was launched at that point in time. Much like the OS artefacts, logging of historical app launches would be useful to say with confidence that a user artefact is synced because the app was not launched at that time. But it wouldn't afford the same confidence to say that it must be local because the app was launched. If two devices are being used at the same time for the same user account, the local and synced data will both be present on the device, making it impossible to form the opinion that it must be local based solely on the fact that the app was launched at that time. Likewise, there are operating systems that lack the documentation necessary to assist a computer forensic examiner in applying the OS analysis portion of this framework.

Conclusion and future work

Synchronization of data between devices has been around since at least 2011. The era of ubiquitous syncing of user data arrived with smart phones and is expanding with the Internet of Things and wearable technology. But there is a void in research, best practices, and tools to deal with synced evidence between this expanding universe of connected devices. Computer forensic examiners are not yet facing challenges in court that synced data did it, but it's only a matter of time. It has the potential to become the go to defense in coming years much like the malware defense of the past decade.

The framework presented in this paper is the first step in raising awareness of the sync defense within the computer forensic community and providing a possible solution. The need to consider the impact of sync data on evidence is important in order to carry out an impartial analysis of electronic evidence, and identify both inculpatory and exculpatory evidence. The framework provides computer forensic examiners with the guidance needed to address the sync defense should it be raised in court.

There are likely other potential OS artefacts on a Windows 10 system that could help with applying this framework, such as evidence in the registry (i.e. *UserAssist*), that were not explored in this research paper. Research could be undertaken to leverage machine learning such as collective detection (Le-Khac et al., 2010) to help form an opinion whether a user was at the keyboard or not. Recently, we are also studying the feasibility of using this framework to investigate financial crimes (van Banerveld et al., 2014).

References

Alabdulsalam, S., Schaefer, K., Le-Khac, N.-A., January 2018. Internet of things forensics: Challenges and case study. In: 14th Annual IFIP WG11.9 International Conference on Digital Forensics, New Delhi, India.

Carlson, J., Moren, D., 2015. Discover the Rich Apple Ecosystem of the Mac, iPhone, iPad, and AppleTV. Peachpit Press.

Dykstra, J., 2013. Seizing electronic evidence from cloud computing environments. In: Cybercrime and Cloud Forensics: Applications for Investigation Processes. Hershey, IGI Global, pp. 156–185.

Faheem, M., Kechadi, M.T., Le-Khac, N.-A., 2015. The state of the art forensic techniques in mobile cloud environment: a survey, challenges and current trends. Int. J. Digit. Crime Forensics (IJDCF) 7 (2), 1–19.

Friedman R., et al. A digital forensic analysis on the iCloud® and its synchronization to Apple® devices: Research Report. Marshall University Forensic Science Center, August 2012.

Grunberg, K., 2011. Chrome Beta Release [Online]. Available: https://chromereleases.googleblog.com/2011/09/chrome-beta-release.html.

Hale, J.S., 2013. Amazon cloud drive forensic analysis. Digit. Invest. 10 (3), 259–265.

Lillis, D., Becker, B.A., O'Sullivan, T., Scanlon, M., April 2016. Current challenges and future research areas for digital forensic investigations. In: 11th ADFSL Conference on Digital Forensics, Security and Law (CDFSL 2016), Florida, USA.

Le-Khac, N.-A., Markos, S., Kechadi, M.-T., June 2010. A data mining-based solution for detecting suspicious money laundering cases in an investment bank. In: 2nd International Conference on Advances in Databases Knowledge and Data Applications (DBKDA), Menuires, France.

Oh, J., Lee, S., Lee, S., August 2011. Advanced evidence collection and analysis of web browser activity. In: The Digital Forensic Research Conference, DFRWS 2011 USA, New Orleans, LA, USA.

Ryder, S., Le-Khac, N.-A., June 2016. The end of effective law enforcement in the cloud? – to encrypt, or not to encrypt. In: 9th IEEE International Conference on Cloud Computing, Sanfrancisco, CA, USA.

Schatz, B., Gladyshev, P., van der Knijff, R., 2014. The Internet of things: interconnected digital dust. Digit. Invest. 11 (3), 141–142.

Shavers, B., 2013. Placing the Suspect behind the Keyboard. Elsevier, Waltham, MA.

Shavers, B., Zimmerman, E., 2014. X-ways Forensics Practitioner's Guide. Syngress, Elsevier.

van Banerveld, M., Le-Khac, N.-A., Kechadi, M.T., 2014. Performance evaluation of a natural language processing approach applied in white collar crime investigation. In: Dang, T.K., Wagner, R., Neuhold, E., Takizawa, M., Küng, J., Thoai, N. (Eds.), Future Data and Security Engineering. Lecture Notes in Computer Science, vol. 8860. Springer, Cham.

Warren, C., El-Sheikh, E., Le-Khac, N.-A., 2017. Privacy preserving internet browsers: forensic analysis of browzar. In: Daimi, K. (Ed.), Computer and Network Security Essentials. Springer, Cham.

Wright, K., 2015. A Study of the Forensic Artefacts Related to Google's Chrome Synchronisation Feature, Their Interpretation and Device Attribution (MSc Thesis). Cranfield University.