518.4

Application Data Analysis

SANS | GIAC CERTIFICATIONS

THE MOST TRUSTED SOURCE FOR INFORMATION SECURITY TRAINING, CERTIFICATION, AND RESEARCH | sans.org

© 2022 Sarah Edwards. All rights reserved to Sarah Edwards and/or SANS Institute.

PLEASE READ THE TERMS AND CONDITIONS OF THIS COURSEWARE LICENSE AGREEMENT ("CLA") CAREFULLY BEFORE USING ANY OF THE COURSEWARE ASSOCIATED WITH THE SANS COURSE. THIS IS A LEGAL AND ENFORCEABLE CONTRACT BETWEEN YOU (THE "USER") AND SANS INSTITUTE FOR THE COURSEWARE. YOU AGREE THAT THIS AGREEMENT IS ENFORCEABLE LIKE ANY WRITTEN NEGOTIATED AGREEMENT SIGNED BY YOU.

With this CLA, SANS Institute hereby grants User a personal, non-exclusive license to use the Courseware subject to the terms of this agreement. Courseware includes all printed materials, including course books and lab workbooks, as well as any digital or other media, virtual machines, and/or data sets distributed by SANS Institute to User for use in the SANS class associated with the Courseware. User agrees that the CLA is the complete and exclusive statement of agreement between SANS Institute and you and that this CLA supersedes any oral or written proposal, agreement or other communication relating to the subject matter of this CLA.

BY ACCEPTING THIS COURSEWARE, USER AGREES TO BE BOUND BY THE TERMS OF THIS CLA. BY ACCEPTING THIS SOFTWARE, USER AGREES THAT ANY BREACH OF THE TERMS OF THIS CLA MAY CAUSE IRREPARABLE HARM AND SIGNIFICANT INJURY TO SANS INSTITUTE, AND THAT SANS INSTITUTE MAY ENFORCE THESE PROVISIONS BY INJUNCTION (WITHOUT THE NECESSITY OF POSTING BOND) SPECIFIC PERFORMANCE, OR OTHER EQUITABLE RELIEF.

If User does not agree, User may return the Courseware to SANS Institute for a full refund, if applicable.

User may not copy, reproduce, re-publish, distribute, display, modify or create derivative works based upon all or any portion of the Courseware, in any medium whether printed, electronic or otherwise, for any purpose, without the express prior written consent of SANS Institute. Additionally, User may not sell, rent, lease, trade, or otherwise transfer the Courseware in any way, shape, or form without the express written consent of SANS Institute.

If any provision of this CLA is declared unenforceable in any jurisdiction, then such provision shall be deemed to be severable from this CLA and shall not affect the remainder thereof. An amendment or addendum to this CLA may accompany this Courseware.

SANS acknowledges that any and all software and/or tools, graphics, images, tables, charts or graphs presented in this Courseware are the sole property of their respective trademark/registered/copyright owners, including:

AirDrop, AirPort, AirPort Time Capsule, Apple, Apple Remote Desktop, Apple TV, App Nap, Back to My Mac, Boot Camp, Cocoa, FaceTime, FileVault, Finder, FireWire, FireWire logo, iCal, iChat, iLife, iMac, iMessage, iPad, iPad Air, iPad Mini, iPhone, iPhoto, iPod, iPod classic, iPod shuffle, iPod nano, iPod touch, iTunes, iTunes logo, iWork, Keychain, Keynote, Mac, Mac Logo, MacBook, MacBook Air, MacBook Pro, Macintosh, Mac OS, Mac Pro, Numbers, OS X, Pages, Passbook, Retina, Safari, Siri, Spaces, Spotlight, There's an app for that, Time Capsule, Time Machine, Touch ID, Xcode, Xserve, App Store, and iCloud are registered trademarks of Apple Inc.

PMP® and PMBOK® are registered trademarks of PMI.

SOF-ELK® is a registered trademark of Lewes Technology Consulting, LLC. Used with permission.

SIFT® is a registered trademark of Harbingers, LLC. Used with permission.

Governing Law: This Agreement shall be governed by the laws of the State of Maryland, USA.

All reference links are operational in the browser-based delivery of the electronic workbook.

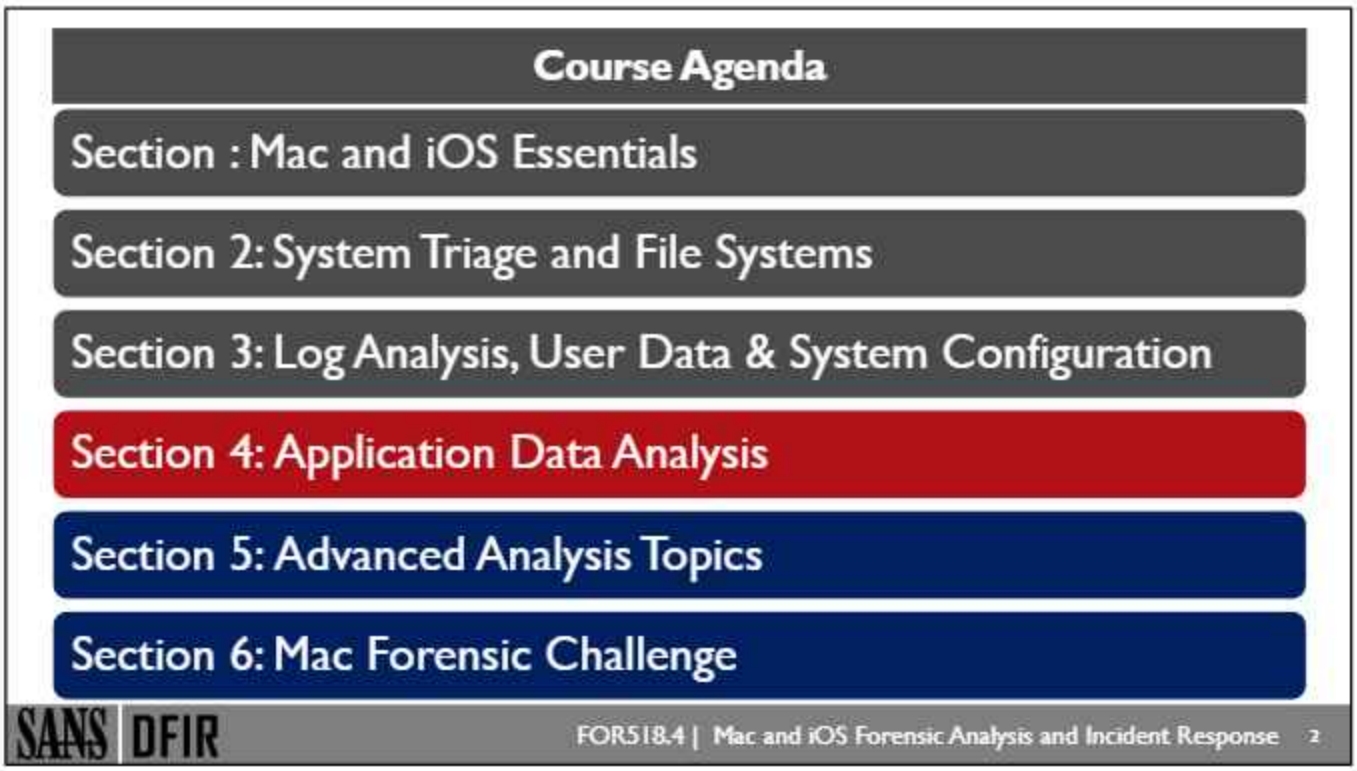
FOR518.4 | Mac and iOS Forensic Analysis and Incident Response

Application Data Analysis

© 2022 Sarah Edwards | All Rights Reserved | Version H02_01

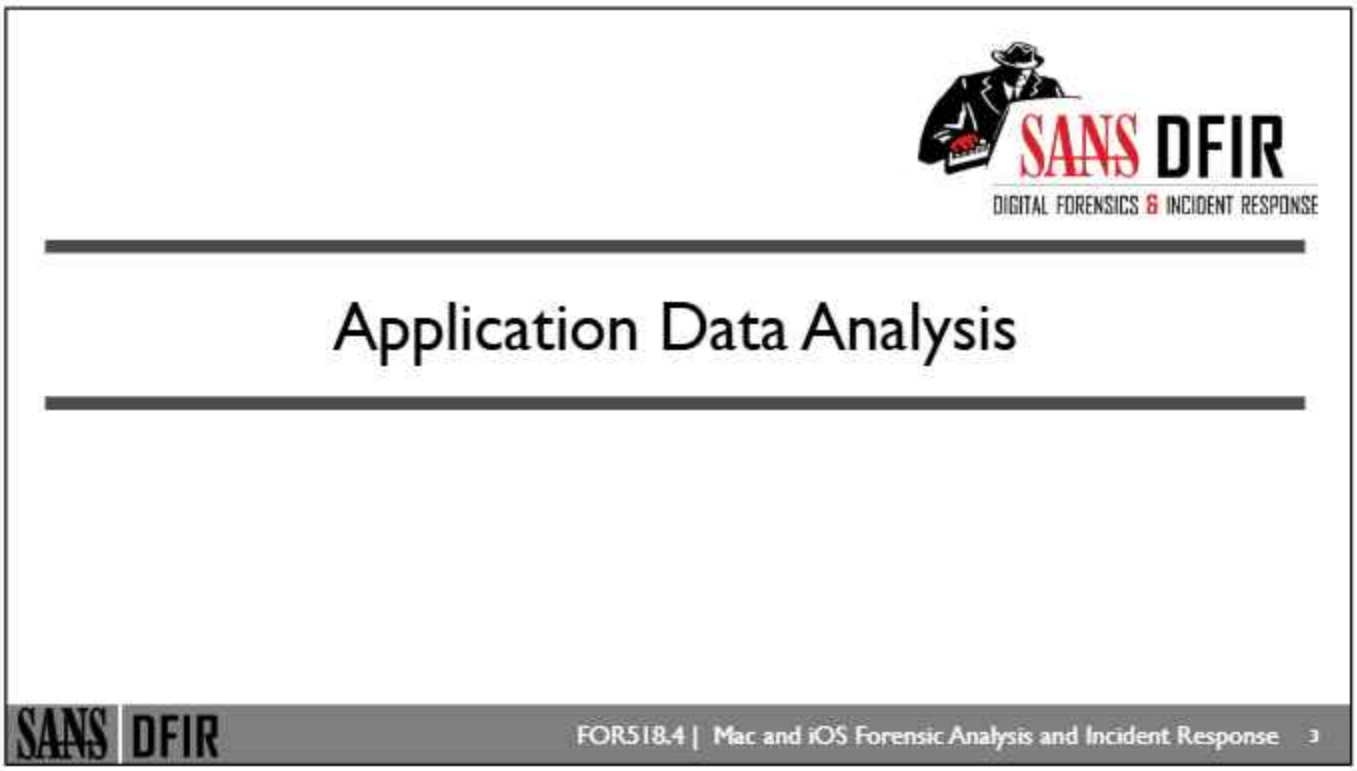
Author: Sarah Edwards
sedwards@sans.edu
mac4n6.com
https://twitter.com/iamevltwin

https://digital-forensics.sans.org/
https://twitter.com/sansforensics



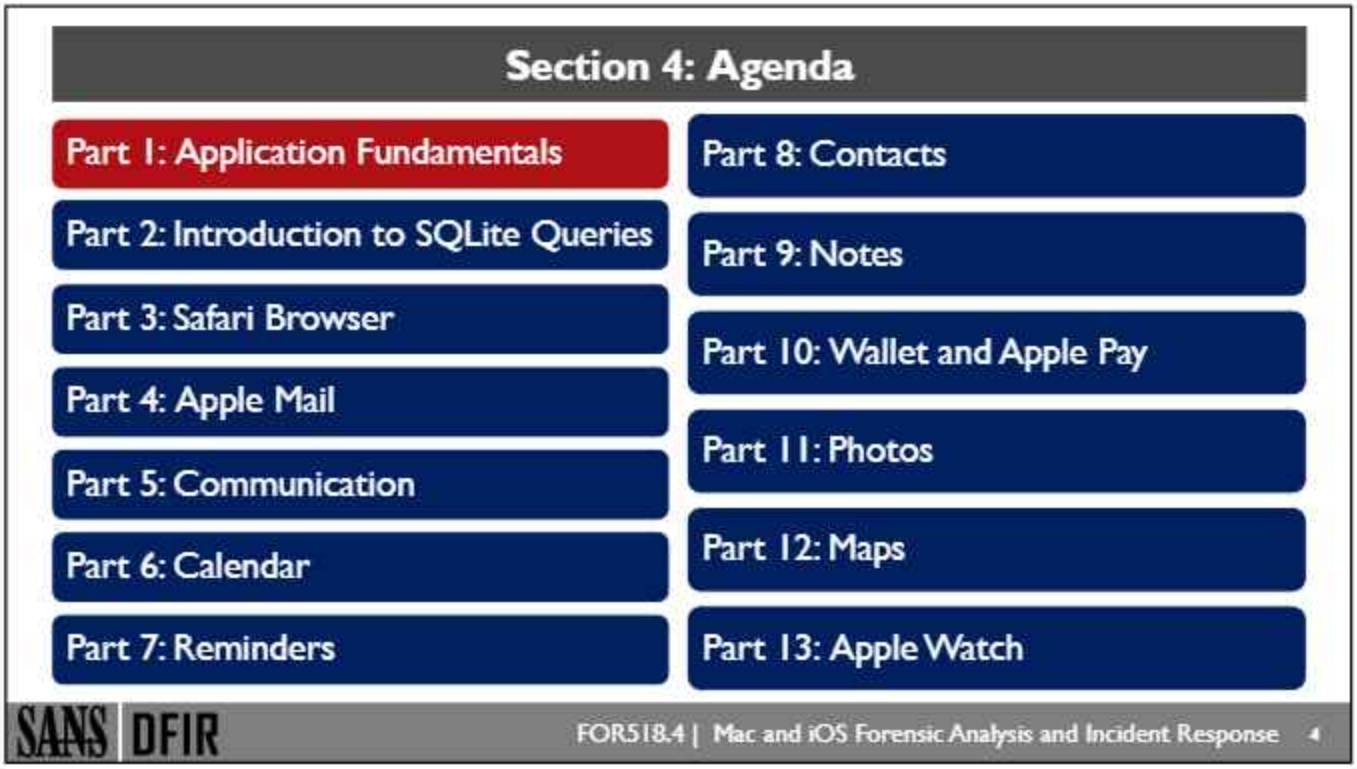
This page intentionally left blank.

© 2022 Sarah Edwards



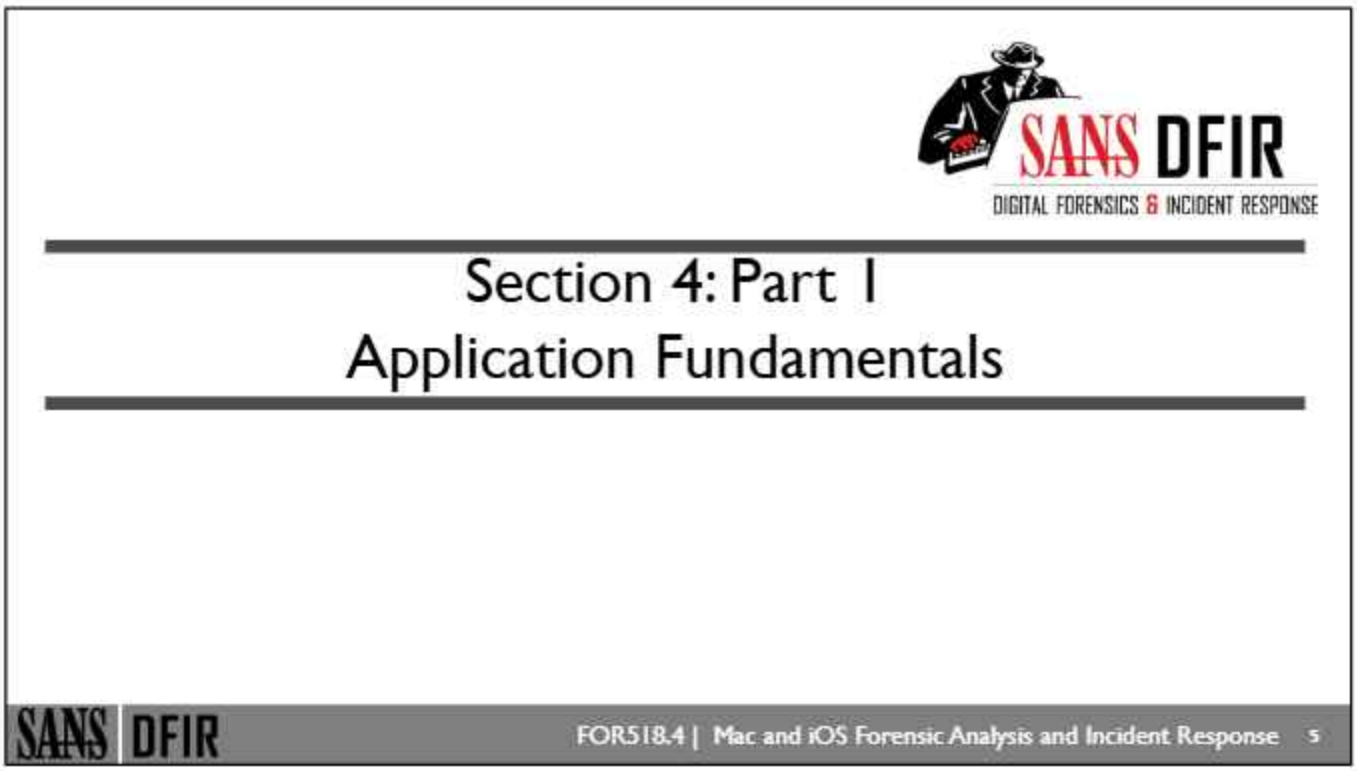
This page intentionally left blank.

© 2022 Sarah Edwards



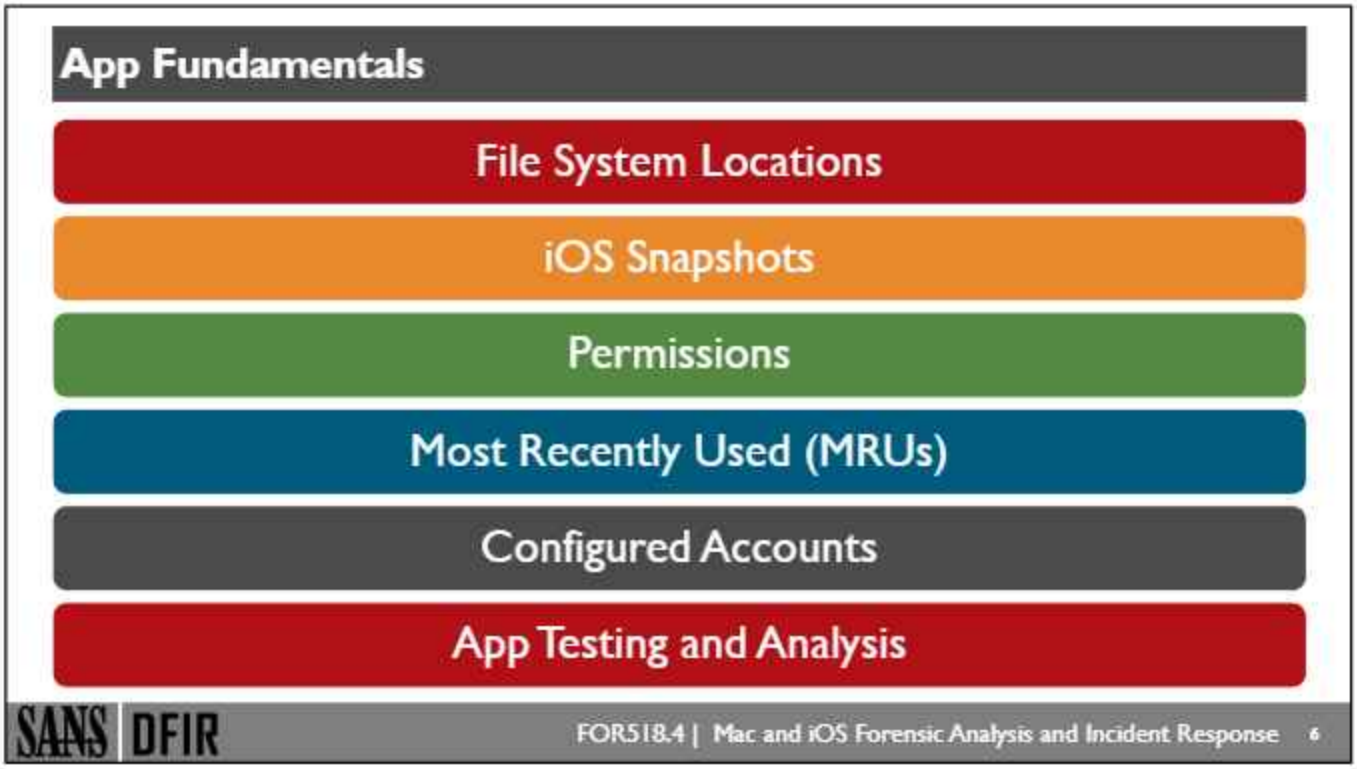
This page intentionally left blank.

© 2022 Sarah Edwards



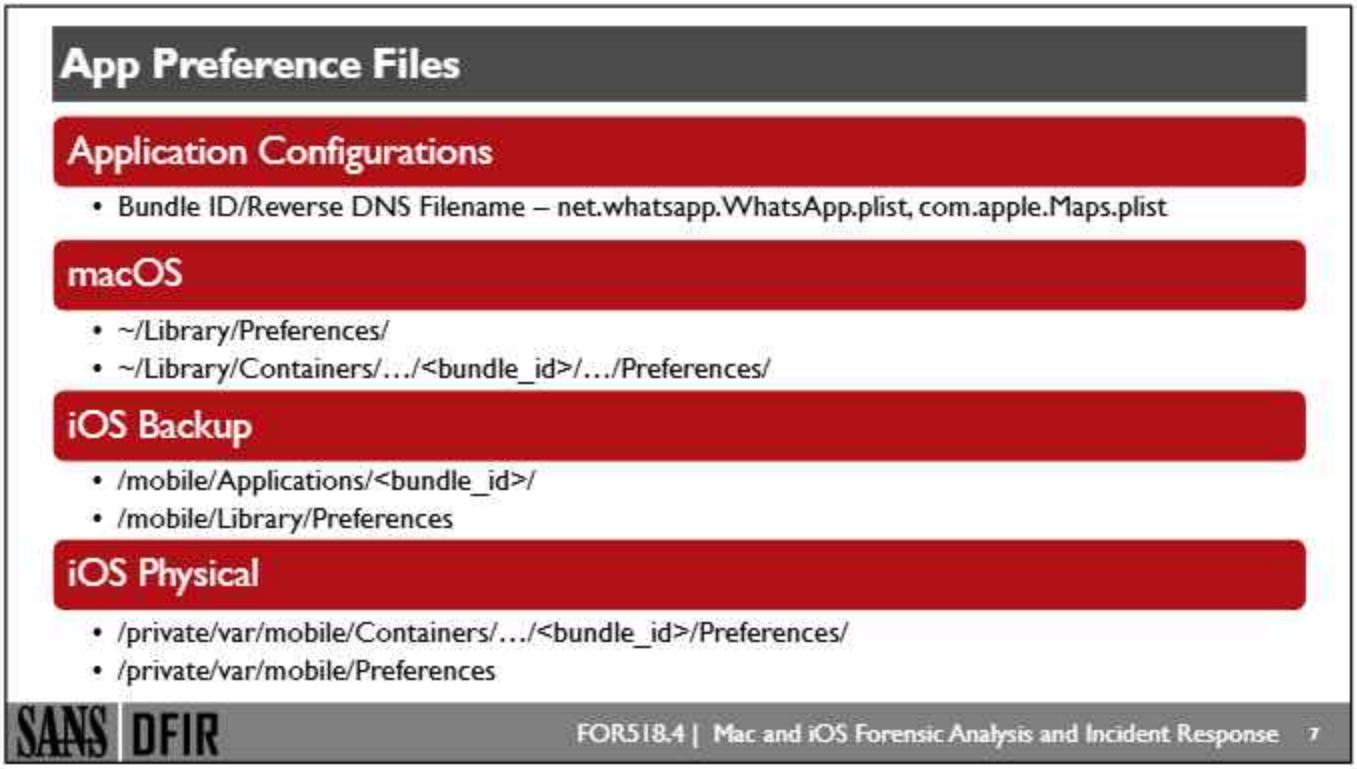
This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards



Preference files are usually plist files that contain configuration data for a particular application. This data can include a wide variety of items.

These preference files are plist files that are named using the bundle ID, the reverse DNS format.

© 2022 Sarah Edwards

Example Preference File: Hyatt Mobile Entry App

Root	Dictionary	(20 items)
ADMS_LifecycleData	Dictionary	(14 items)
ADMS_PAUSE	Date	2015-05-03 23:59:06
ADMS_START	Date	2015-05-03 23:51:31
ADOBEMOBILE_STOREDDEFAULTS_AID	String	2A9D72208507A683-4000010A200860D2
CurrentProperty	Number	2
OMCK1	Date	2015-04-25 00:48:01
OMCK2	String	1
OMCK5	Date	2015-05-03 23:48:26
OMCK6	Number	3
OMCK7	Boolean	True
WebDatabaseDirectory	String	/var/mobile/Containers/Data/Application/A977
WebKitDiskImageCacheSavedCacheDirectory	String	
WebKitLocalStorageDatabasePathPreferenceKey	String	/var/mobile/Containers/Data/Ap
WebKitOfflineWebApplicationCacheEnabled	Boolean	True
WebKitShrinksStandaloneImagesToFit	Boolean	True
com.hyatt.mobile_key.goldPassportId	String	[redacted]
com.hyatt.mobile_key.guestName	String	SARAH EDWARDS
com.hyatt.mobile_key.rememberMe	Boolean	True
com.hyatt.mobile_key.savedGoldPassportNumberOrUsername	String	[redacted]
com.hyatt.mobile_key.savedGoldPassportPassword	String	[redacted]

Yep, password in plaintext ☹

SANS DFIR FOR518.4 | Mac and iOS Forensic Analysis and Incident Response

In addition to application data such as messages or contact lists, an analyst may need to look for user IDs such as email addresses and account numbers. Analysts may also find passwords in plaintext associated with various accounts.

The example above is from the Hyatt Mobile Entry application (`com.hyatt.MobileEntry.plist`). The app configuration plist files may contain some interesting information and are always worth quickly reviewing. If an analyst is trying to find passwords used to attempt to break into other encrypted containers, seemingly random applications on that user's device may contain this information in an easy-to-get way! Why use brute force when you can get passwords in plaintext!

© 2022 Sarah Edwards

App Databases and Other Files

SQLite Databases, Proprietary Files, Logs – Anything Goes!

macOS

- ~/Library/
- ~/Library/Application Support/
- ~/Library/Containers/…

iOS Backup

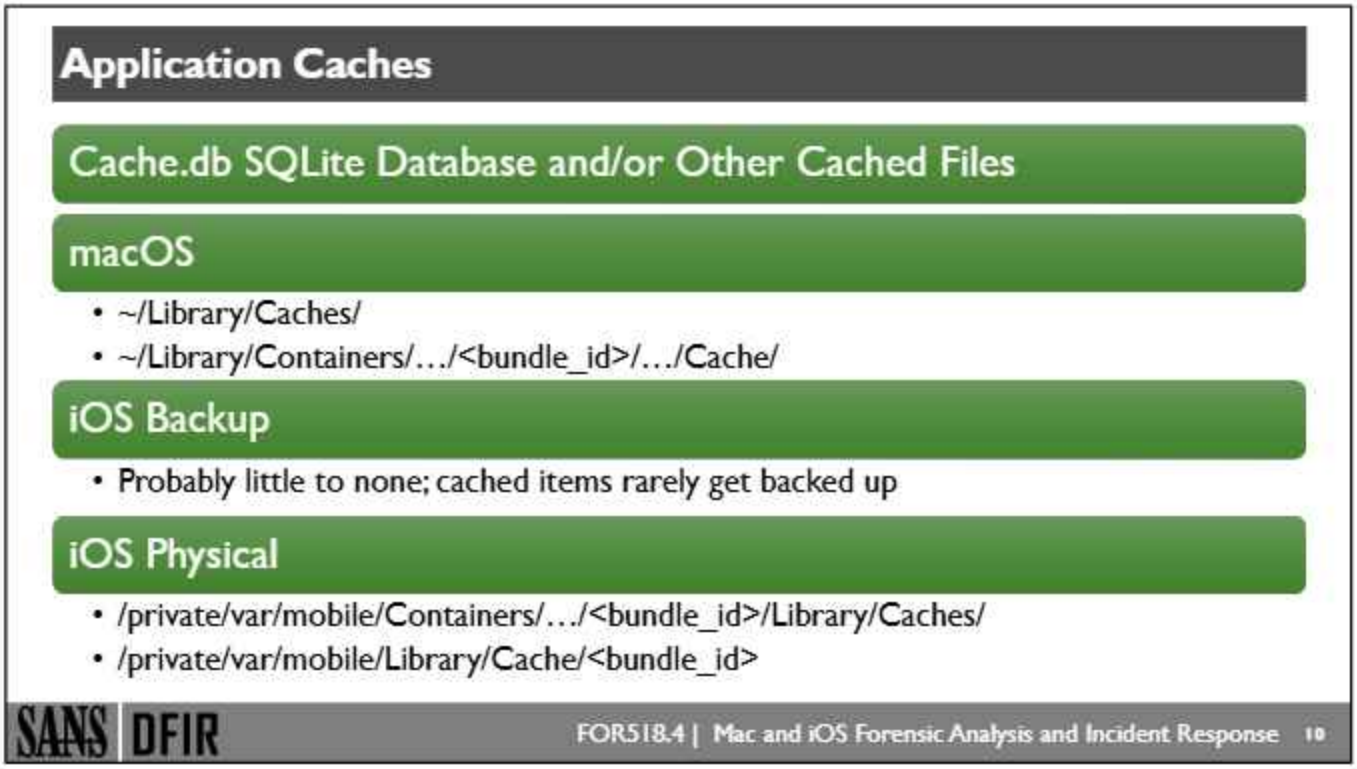
- /mobile/Applications/<bundle_id>/
- /mobile/Library/

iOS Physical

- /private/var/mobile/Containers/… /<bundle_id>/Library/
- /private/var/mobile/Library/

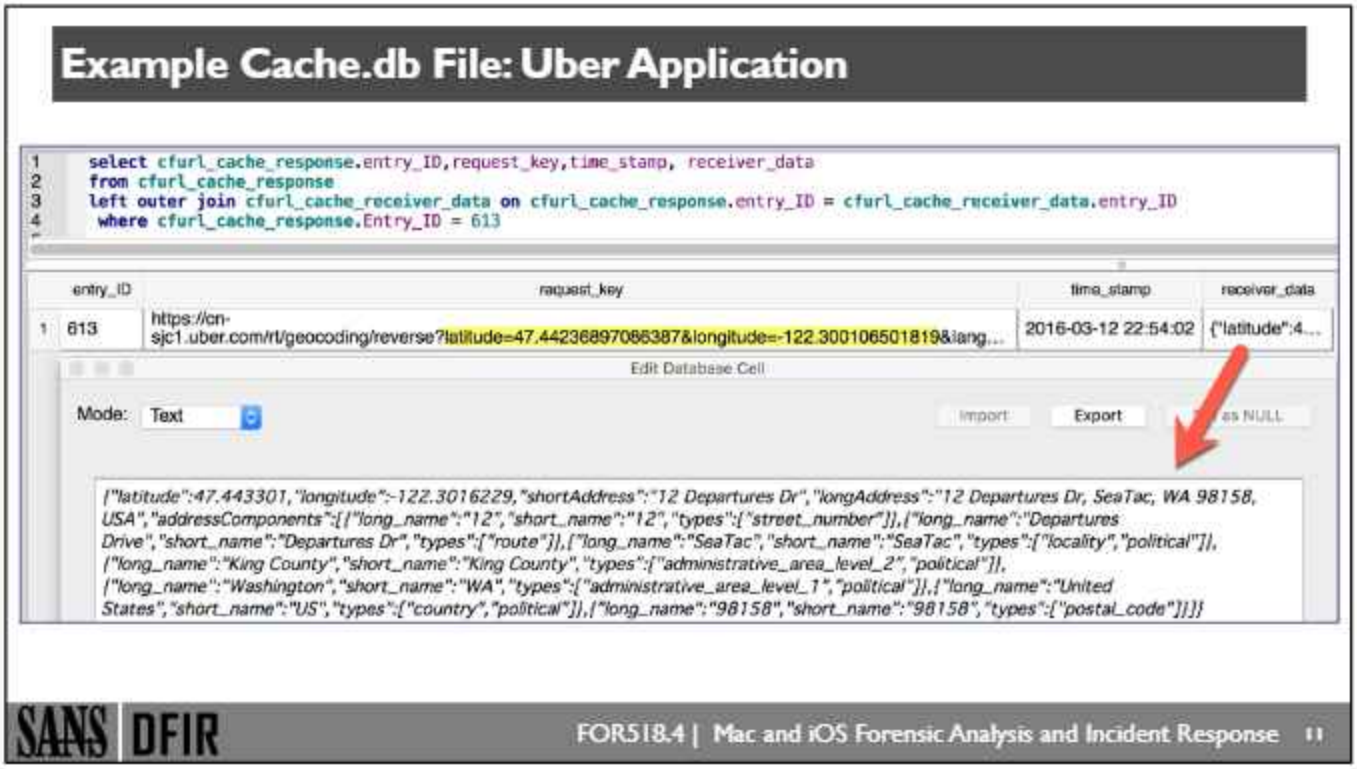
Databases and other application files can be found in these locations. Each application will have its own files representing the data used for the apps themselves. Common files seen are SQLite database files, proprietary data, and maybe some log files.

© 2022 Sarah Edwards



Application caches are always worth a review. Browsers are often the most common for an analyst to review; however, other apps may retain cached data that could be of importance in an investigation.

© 2022 Sarah Edwards



Another type of data an analyst may be searching for is location data. Depending on the acquisition type, the analyst may not have access to the location databases on iOS. An examiner can get creative and look in many types of applications for this data.

The example above shows the `Cache.db` (`com.ubercab.UberClient/Cache.db`) file for the Uber application. Any applications that use location services may be good candidates for location data. The Uber `cache.db` database contains HTTP URL queries containing location data for where the user was requesting a car. The response (shown in the "Edit Database Cell" window) contains the return information from this query, which also contains location information with more granular detail.

iOS Backup Structure: Third-Party App Data

/mobile/Applications/<App_Bundle_ID>

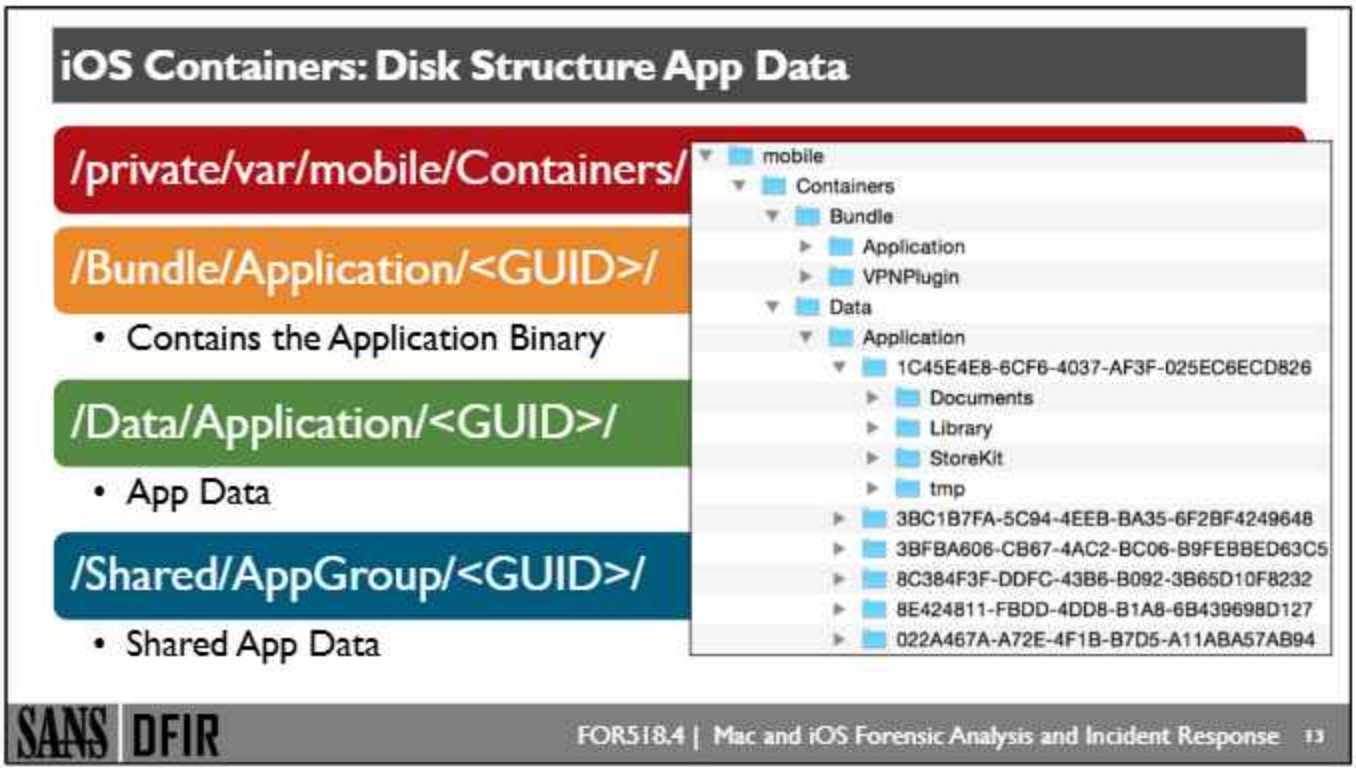
All Third-Party App Data under Apple Bundle ID Directory

Normalized Structure

The iOS backup structure when shown in Inspector (shown) and other tools will show a normalized data structure. This is not exactly how it is being stored on disk.

All application data is stored under the directory associated with that application's Bundle ID.

© 2022 Sarah Edwards



Application data is stored in containers in the file system. Application data may be stored in multiple areas on the file system due to this sandboxing implementation.

The application bundle is stored in the /Bundle directory while application data may be split into the /Data/Application/ and /Shared/AppGroup directories.

The /Shared/AppGroup data can be shared among apps by the same developer, such as the Microsoft Word, PowerPoint, and Excel applications.

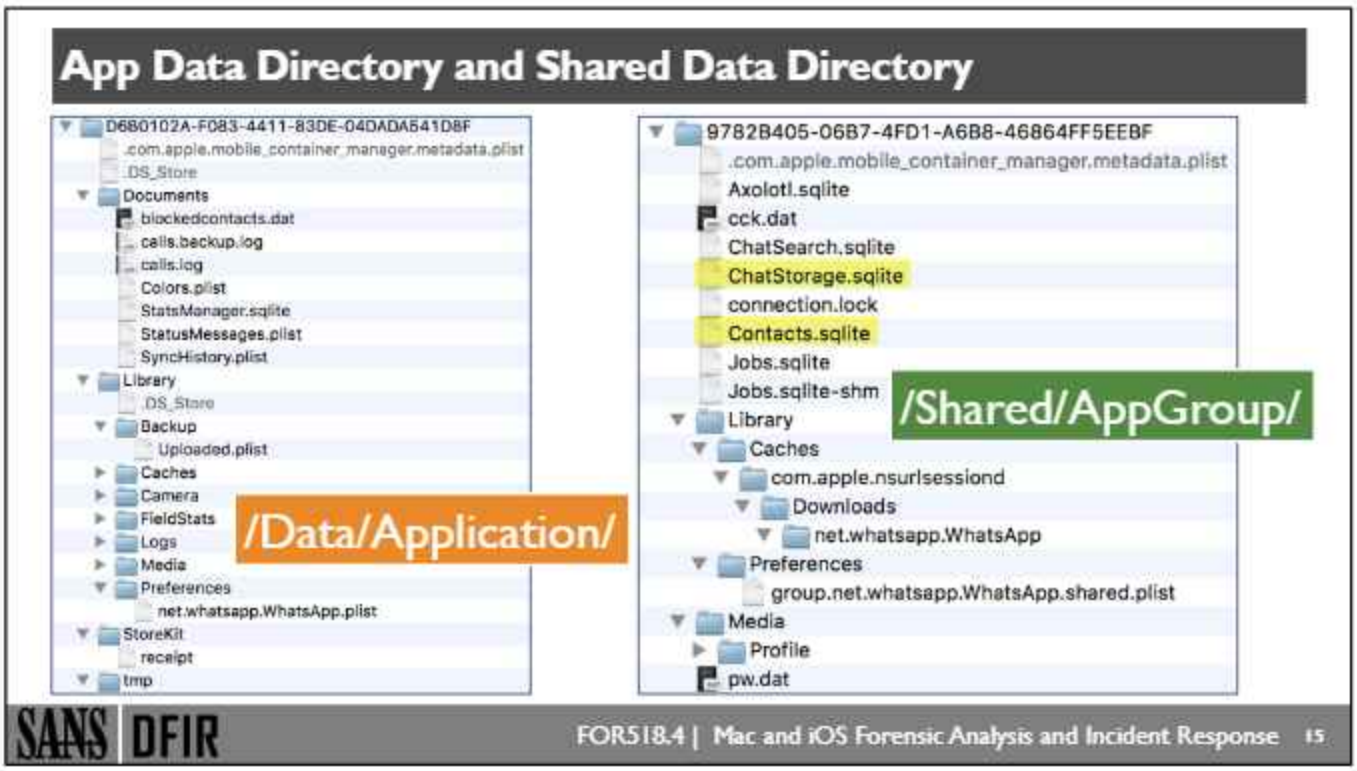
Reference:
https://developer.apple.com/library/archive/documentation/FileManagement/Conceptual/FileSystemProgrammingGuide/FileSystemOverview/FileSystemOverview.html

Each third-party application directory contains a standard set of folders which has certain structure.

The Documents directory may contain documents, but also application databases. The Library directory may contain cookie data, cached data in the cache.db databases, and configuration plists, as well as application analytics information.

Each application will have a hidden plist file (.com_apple.mobile_container_manager.metadata.plist) in its root directory containing the Bundle ID of the application as well as other metadata, such as protection classes and group bundle ID information.

© 2022 Sarah Edwards



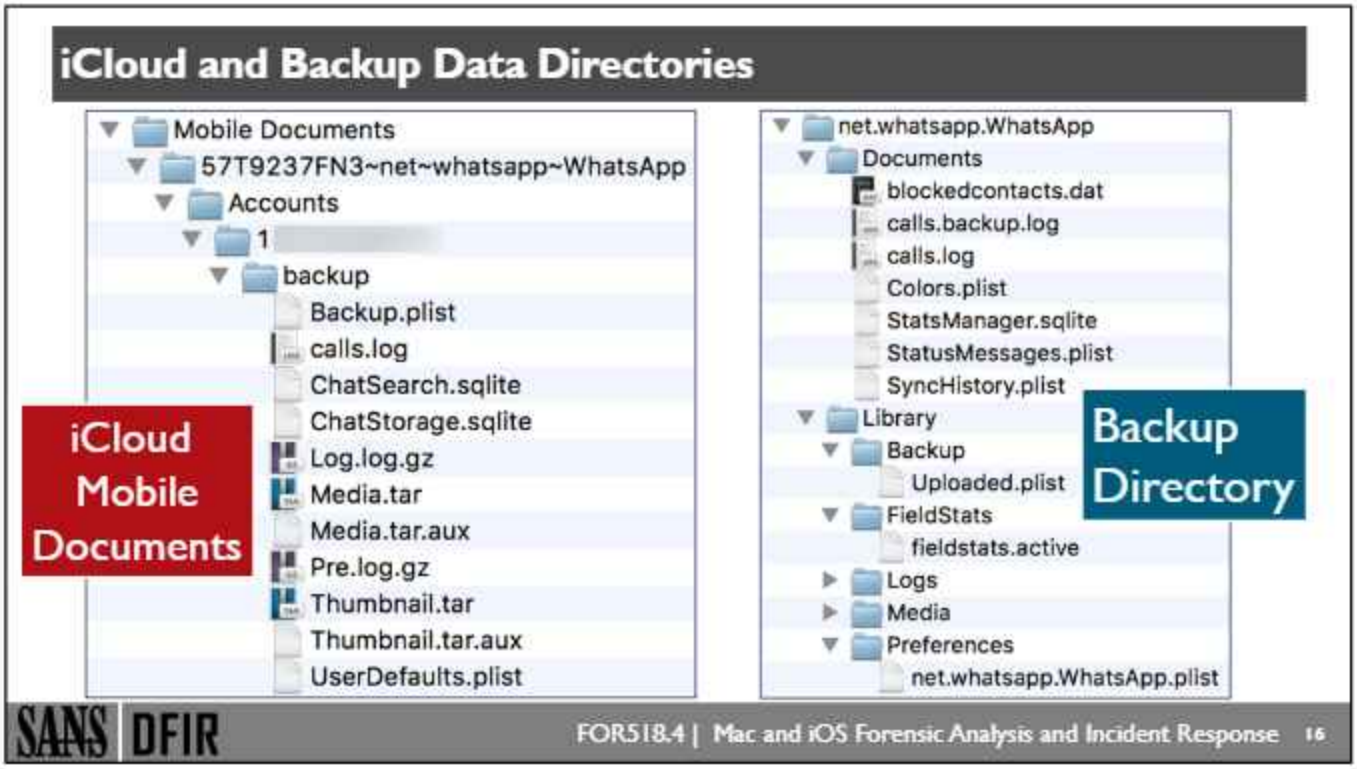
The differences between the `/Data/Application/` and the `/Shared/AppGroup/` directories in the app containers can be quite different and it's always worth looking at both to find the information you are seeking.

On the left is a screenshot of the `/Data/Application` directory for the WhatsApp app (`net.whatsapp.whatsapp`). This shows some information, but if you are looking for the Messages and Contacts databases for the WhatsApp app, you will have to look in the `/Shared/AppGroup/` directory instead.

Not shown in the screenshots above, the Media directory in the `/Data/Application/` directory contains the media (photos/videos) sent back and forth in conversations—this media is not in the `/Shared/AppGroup/` directory for WhatsApp.

These examples come from FFS dumps.

© 2022 Sarah Edwards



Along with the `/Data/Application/` and `/Shared/AppGroup/` directories, the analyst should also look at the iCloud Mobile Documents directory (shown left), as well as any backups that may be found on the device.

Each directory structure will choose to store only a specific directory. If the app does not want to back up anything to iCloud, you will likely not find much in the `Mobile Documents` directory. If the app developer chooses to back up specific items in local backups, you will find these in the normalized data structure, such as the example shown on the right for WhatsApp.

The `Mobile Documents` directory example comes from an FFS dump.

© 2022 Sarah Edwards

iOS Application Snapshots

Native and third-party applications

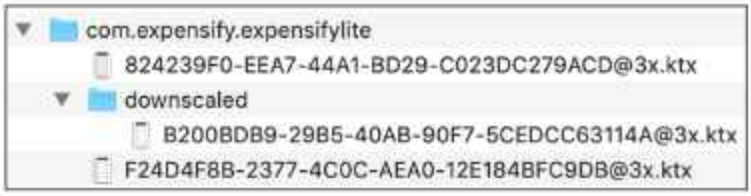
Screenshot taken when application is backgrounded

- Device locked, app switch, app interruption, etc.

Application may not allow screenshots to be taken

Available on physical acquisitions (some in backups in iOS 13+)

- <app_dir>/Library/Caches/Snapshots/<bundle_id>/



Whenever an application is backgrounded, a screenshot is taken of what the user is currently viewing on the screen and is saved to the file system. This allows for smoother transitions and the appearance of faster switching between applications.

A screenshot is taken when an app is switched to another application, when the device is locked, or when an application is interrupted by another application (i.e., receiving a phone call).

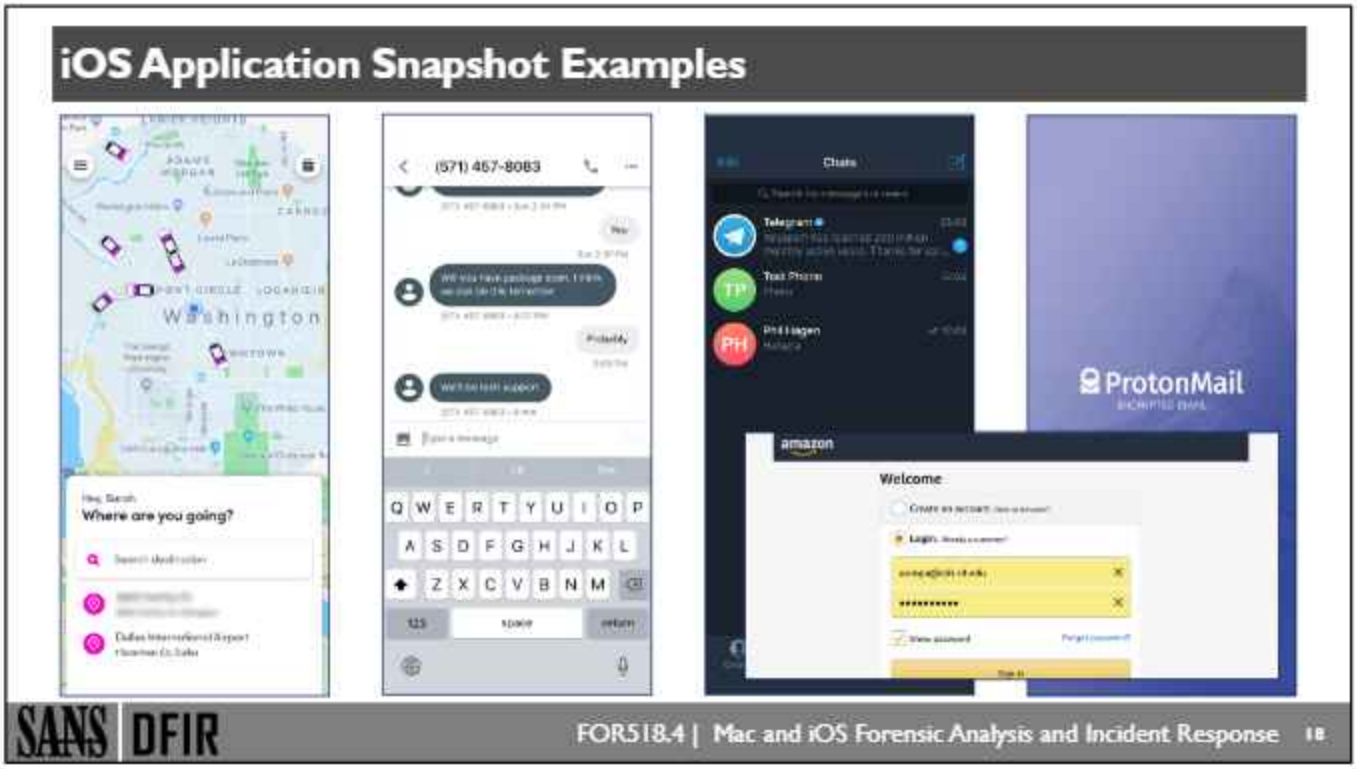
These screenshots may be available for native as well as third-party applications. Depending on the application, developers may choose to limit this screen capture by explicitly telling their code to do so. This may be done for "security", as sensitive data may be saved in these applications. Many messaging or banking applications do not use this feature, or they use a decoy image instead.

These screen captures are only available on physical acquisitions.

The files use the format—KTX. These files can be viewed in the macOS Preview application. (These screenshots were PNG files on iOS 9-.)

Reference:
https://github.com/KhronosGroup/KTX-Software/wiki

© 2022 Sarah Edwards



Snapshots can be used in many ways to help an investigation. You are viewing a snapshot in time of exactly what the user was seeing. These pictures have the potential to contain information that may not necessarily be stored in the applications database or plist files.

Snapshots may be different for applications, depending upon whether the user was using the application in portrait or landscape modes.

Contacts / Address Book: Last contact viewed
Camera: Last "thing" viewed, perhaps without having an actual photo taken
Calendar: Schedule the last day viewed
Messages: List of messaged contacts with last message to them—could potentially be captured before message is deleted
Third-Party Messaging Apps: Messages viewed, typed
Mapping Applications: Last viewed location
Web Browsers: Last viewed webpage

The examples above from left to right are the following apps:

- Lyft
- Google Voice
- Telegram ("Secure" Messenger)
- Proton Mail – Secure Email
- Safari Browser (horizontal)

© 2022 Sarah Edwards

Application Transparency, Consent, Control (TCC)

Application Permissions

SQLite Database

macOS

- User: ~/Library/Application Support/com.apple.TCC/TCC.db
- System: /Library/Application Support/com.apple.TCC/TCC.db

iOS

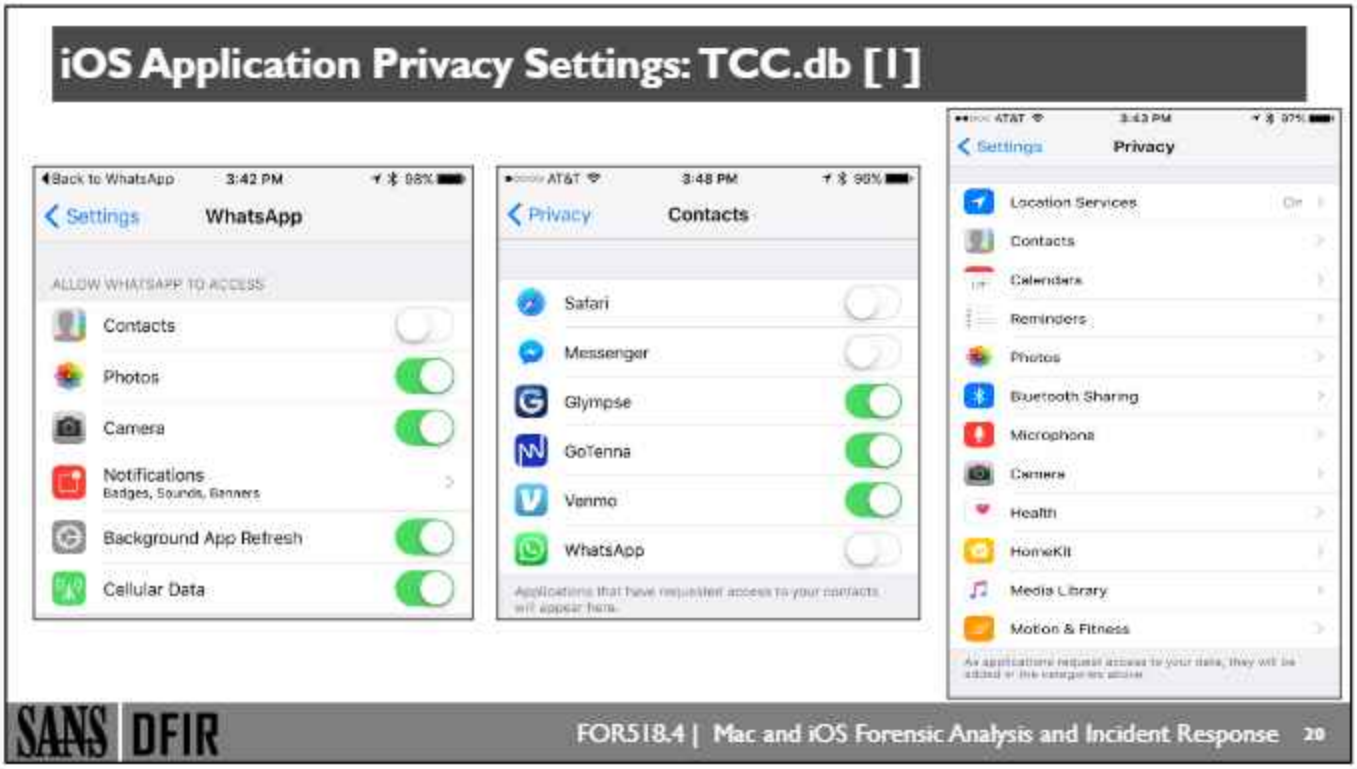
- Backup/FS: /mobile/Library/TCC/TCC.db
- Physical: /private/var/mobile/Library/TCC/TCC.db

Applications on macOS and iOS ask users which permissions they may have for different capabilities available on the system. This database is named `TCC.db` and is a SQLite database.

On macOS, they are recorded in a couple of different databases: One for the system, and one for each user.

On iOS, they are stored in one database and are available in a backup or physical acquisition.

© 2022 Sarah Edwards



The `TCC.db` database contains application privacy settings. This database gets its name from Transparency, Consent, and Control.

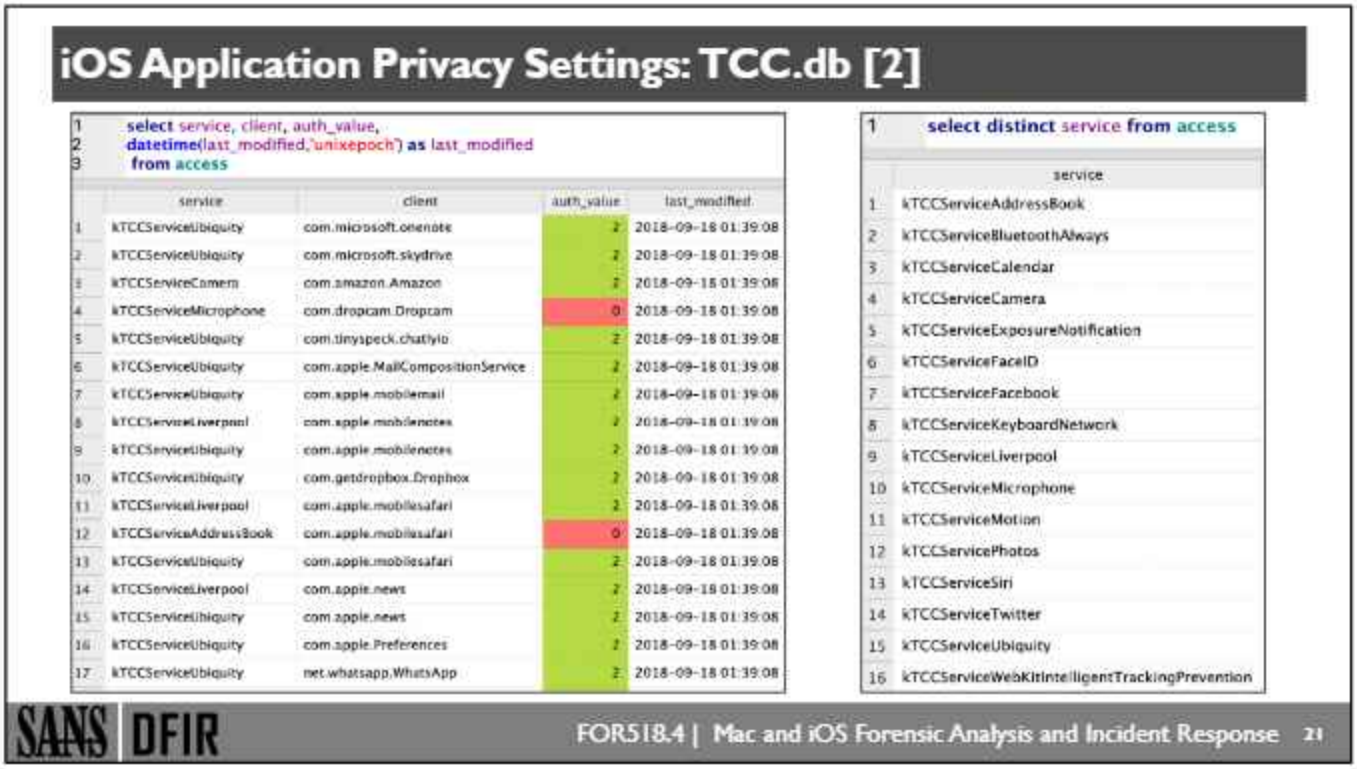
This database can be viewed to determine what a specific application has access to. In the screenshot to the right, an application may have access to some of the following items: Location, Contacts, Calendars, Reminders, Photos, Bluetooth, Microphone, Camera, Health/Fitness, Media, HomeKit, etc.

Each application's settings can be viewed in that particular configuration in the Settings area on the device, as shown in the screenshot to the left. The WhatsApp application on this device has access to Photos and Camera, but it does not have access to Contacts.

A certain type of permission can also be viewed to determine which applications have access. The middle screenshot shows an example of this. The Contacts database is only available to the Glypmse, GoTenna, and Venmo applications but not Safari, Facebook Messenger, or WhatsApp.

References:
https://developer.apple.com/videos/play/wwdc2016/705/
https://developer.apple.com/videos/play/wwdc2016/709/

© 2022 Sarah Edwards



The `TCC.db` SQLite database file keeps track of what app has which permissions or "services" as it is labeled in the database. The `last_modified` timestamp for that permission is also tracked.

The screenshot above shows the following:

- `com.amazon.Amazon` has access to the camera
- `com.dropcam.Dropcam` does not have access to the microphone

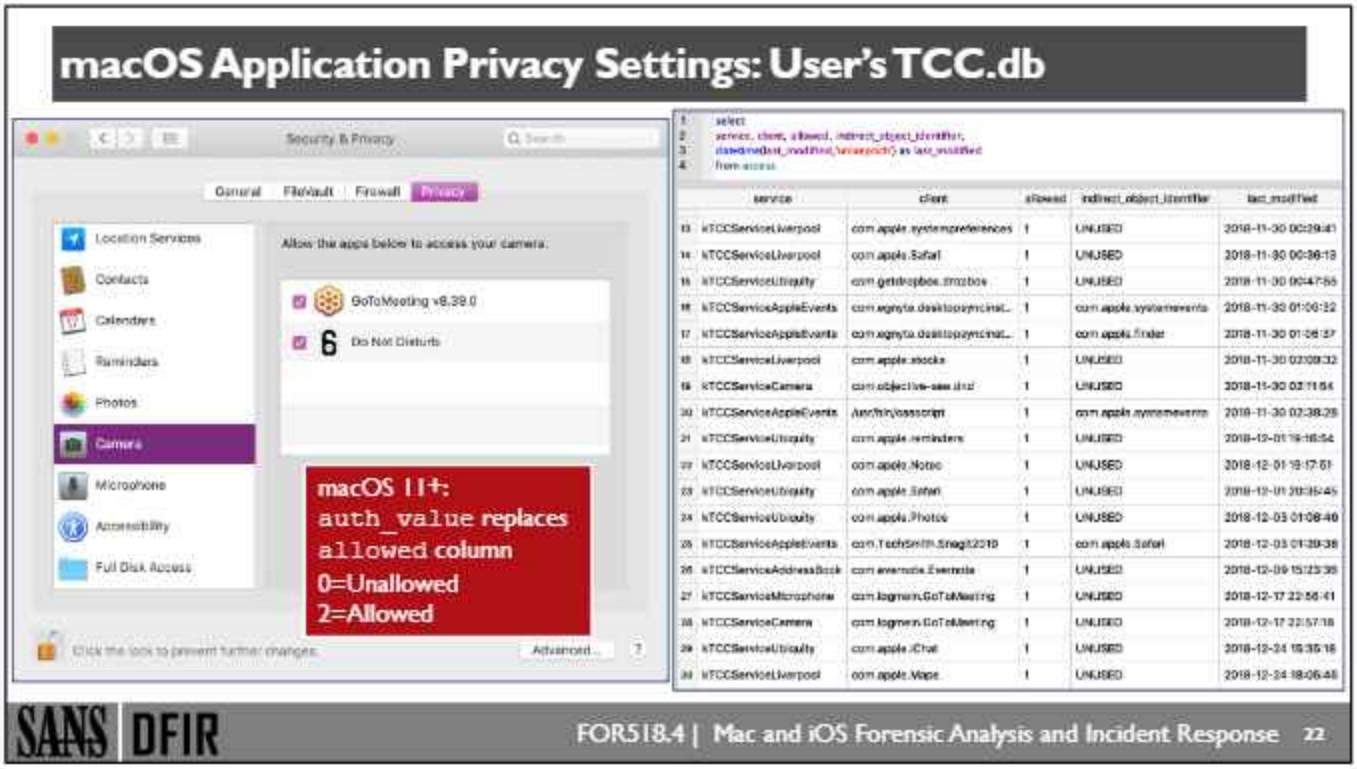
The `auth_value` column was changed from value in iOS 14 (where 1=Allowed, 0=Unallowed). Now 0=Unallowed, while 2=Allowed.

Services may include the following:

- `kTCCServiceAddressBookk`
- `TCCServiceBluetoothAlways`
- `kTCCServiceCalendar`
- `kTCCServiceCamera`
- `kTCCServiceExposureNotification`
- `kTCCServiceFaceID`
- `kTCCServiceFacebook`
- `kTCCServiceKeyboardNetwork`
- `kTCCServiceLiverpool`
- `kTCCServiceMicrophone`
- `kTCCServiceMotion`
- `kTCCServicePhotos`
- `kTCCServiceSiri`
- `kTCCServiceTwitter`
- `kTCCServiceUbiquity`
- `kTCCServiceWebKitIntelligentTrackingPrevention`

References:
https://developer.apple.com/videos/play/wwdc2016/705/
https://developer.apple.com/videos/play/wwdc2016/709/

© 2022 Sarah Edwards



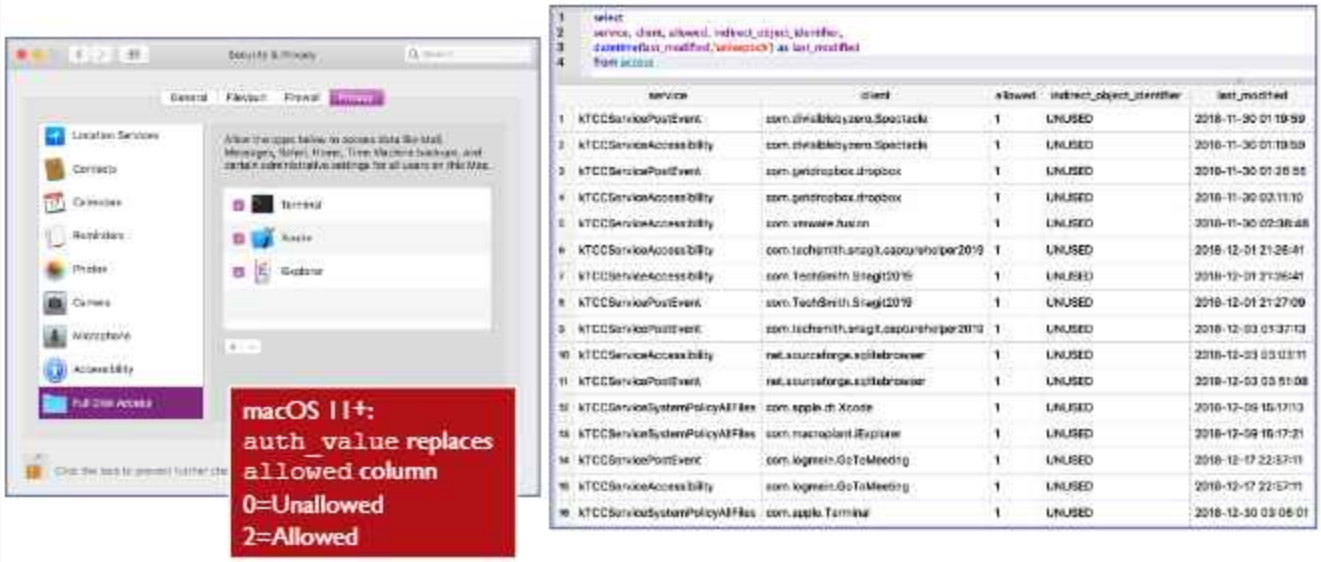
The macOS privacy settings found in `TCC.db` are taken from `System Preferences | Security & Privacy | Privacy`.

For this macOS user example, two applications are using the Camera permission, GoToMeeting and Objective See's Do Not Disturb. In 10.14, the last modification time was added.

Other privacy settings may use the Address Book, Calendar, Reminders, Diagnostics, and/or various social media credentials such as Facebook and Twitter.

© 2022 Sarah Edwards

macOS Application Privacy Settings: System TCC.db



SANS DFIR | FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 23

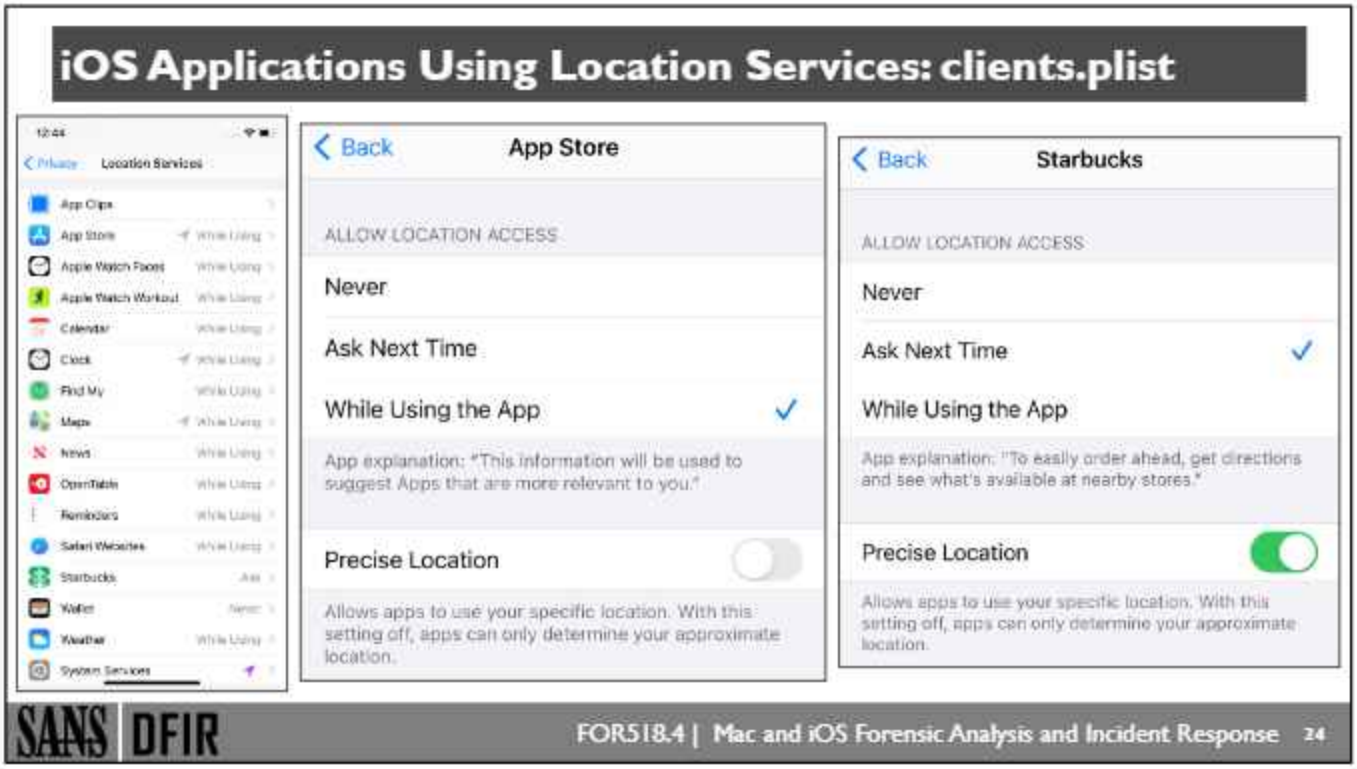
The macOS privacy settings found in `TCC.db` are taken from `System Preferences | Security & Privacy | Privacy`.

For this macOS System example, 10.14 added more permissions-based configurations. The one forensic analysts may run into the most often is the 'Full Disk Access' permission. This will limit access to some file system directories and files unless the application is provided access. In the example above, three apps have been given access:

- Terminal.app
- Xcode.app
- iExplorer.app

The database shows this access with the service "`KTCCServiceSystemPolicyAllFiles`".

© 2022 Sarah Edwards



Backup: `/root/Library/Caches/locationd/clients.plist`
FFS: `/private/var/root/Library/Caches/locationd/clients.plist`

The locationd directory contains several files related to location information stored on an iOS device. Each iOS app may request different levels of location service access; Never, While Using, Ask, and Always.

iOS 14 introduced the Precise Location option for each application.

© 2022 Sarah Edwards

iOS App Location Service Authorizations: clients.plist

com.apple.AppStore	Dictionary	(12 items)
BundleId	String	com.apple.AppStore
CorrectiveCompensationEnabled	Number	2
SignificantTimeStopped	Number	644,076,935.722987
SupportedAuthorizationMask	Number	3
RemoteUsage	Dictionary	(0 items)
ReceivingLocationInformationTim...	Number	644,076,787.06667
Whitelisted	Boolean	0
Authorization	Number	2
Registered	String	/Applications/AppStore.app/AppStore
Executable	String	/Applications/AppStore.app/AppStore
LocationTimeStopped	Number	644,076,926.904817
SLC	Dictionary	(2 items)

Authorization:
1: Never
2: While Using
4: Always
None: Ask

com.starbucks.mystarbucks	Dictionary	(7 items)
Whitelisted	Boolean	0
BundleId	String	com.starbucks.mystarbucks
SupportedAuthorizationMask	Number	3
RemoteUsage	Dictionary	(0 items)
Registered	String	
CorrectiveCompensationEnabled	Number	1
Executable	String	

CorrectiveCompensationEnabled:
'Precise Location' (iOS14):
1: Enabled (or no key listed)
2: Disabled

SANS DFIR | FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 25

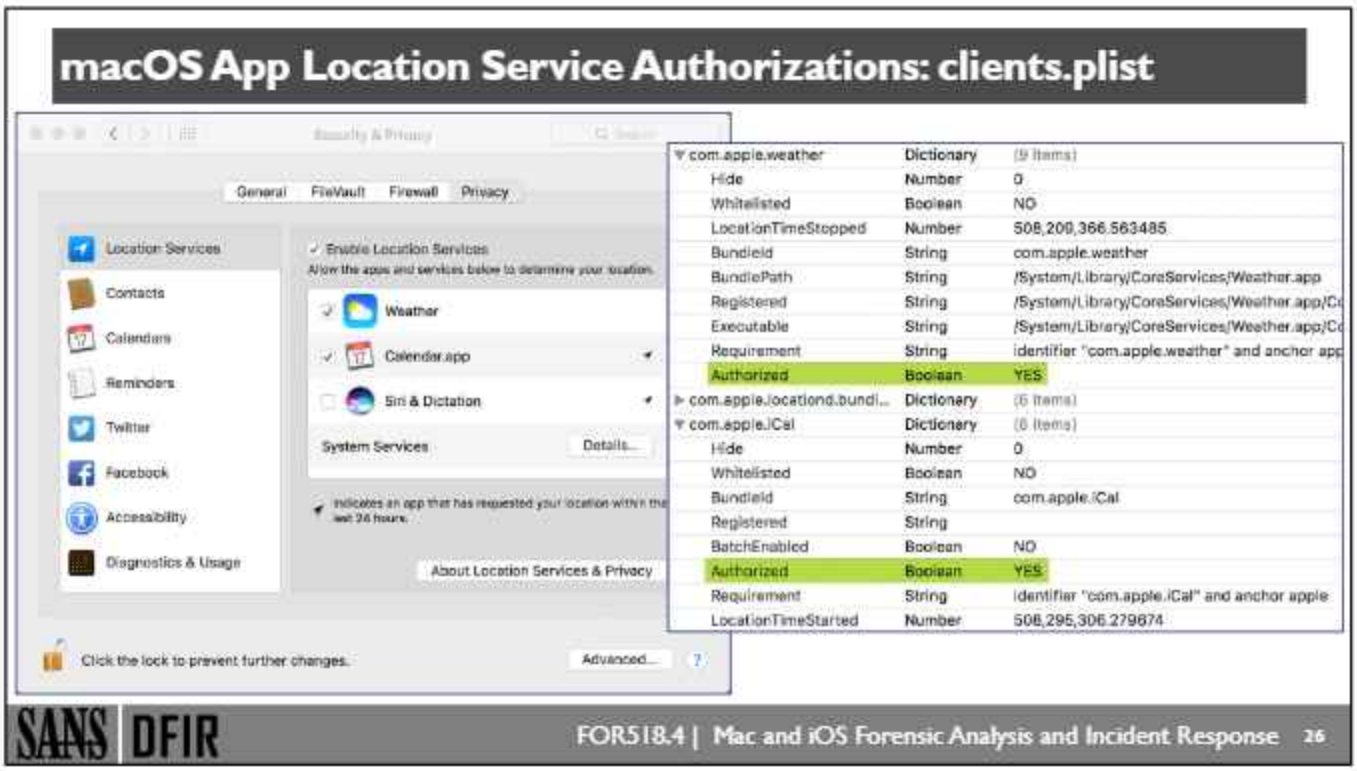
The `clients.plist` file maintains a list of all the applications that have been granted Location Services permissions. This is important because it may clue in the examiner to pay extra attention to those applications listed in the file, as GPS coordinates may be stored along with other relevant user information. Along with applications, OS services can also use Location Services to notify the user of traffic conditions or find their lost iPhone.

Each application's location services information is detailed in the `clients.plist` file. This plist file contains the "`LocationTimeStopped`" key, which keeps track of the time in Mac Epoch of when the application last used Location Services to get the location of the device. Similar timestamps may also exist for each application. The key "`Authorization`" shows the level of permission for location services for that application. If no 'Authorized' key exists, it is set to 'Ask Next Time'.

1: Never authorized (no Location Services)
2: Only use Location Services when using the application
4: Always use Location Services (even when the application is backgrounded)

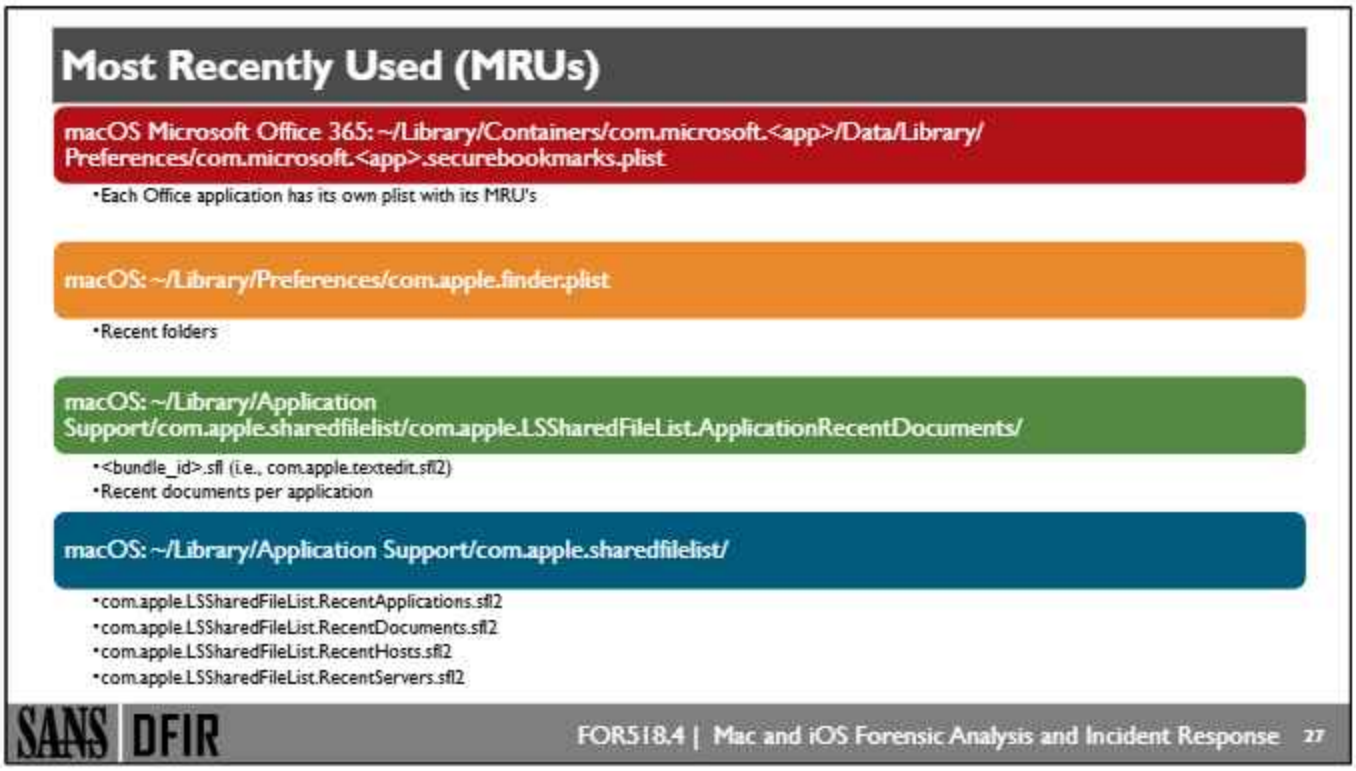
Precise Location is stored in the 'CorrectiveCompensationEnabled' key where 1=Enabled and 2=Disabled. If this key does not exist it is assumed enabled. The user can toggle the configuration on/off to create the key.

© 2022 Sarah Edwards



Just like with iOS, the location information is stored in the `clients.plist` file on macOS. This file is located in the `/private/var/db/locationd/` directory.

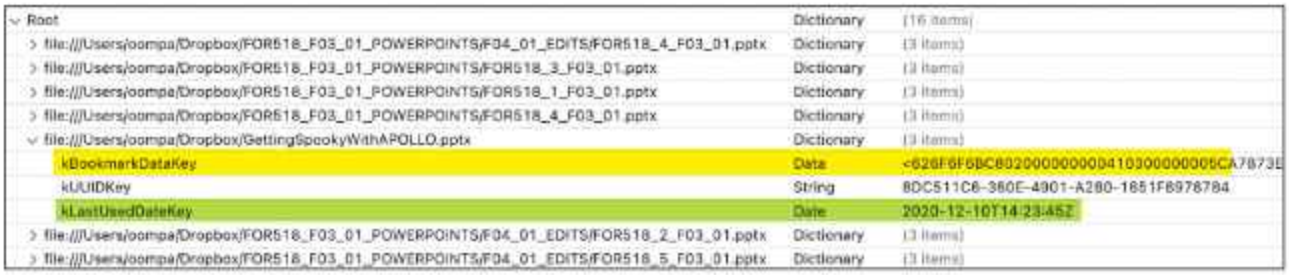
© 2022 Sarah Edwards



Microsoft Office 365 (and 2016) use the com.microsoft.<app>.securebookmarks.plist for each application in the Office suite.

The Shared File List (SFL) MRU format to store the recent documents for each application and other recent items. This format uses the NSKeyedArchiver format. These are binary plist files that have the file extension "`.sfl`". In 10.13, this format changed slightly with the *.sfl2 file extension.

Microsoft Office 365: MRUs
~/Library/Containers/com.microsoft.<app>/Data/Library/Preferences/com.microsoft.<app>.securebookmarks.plist

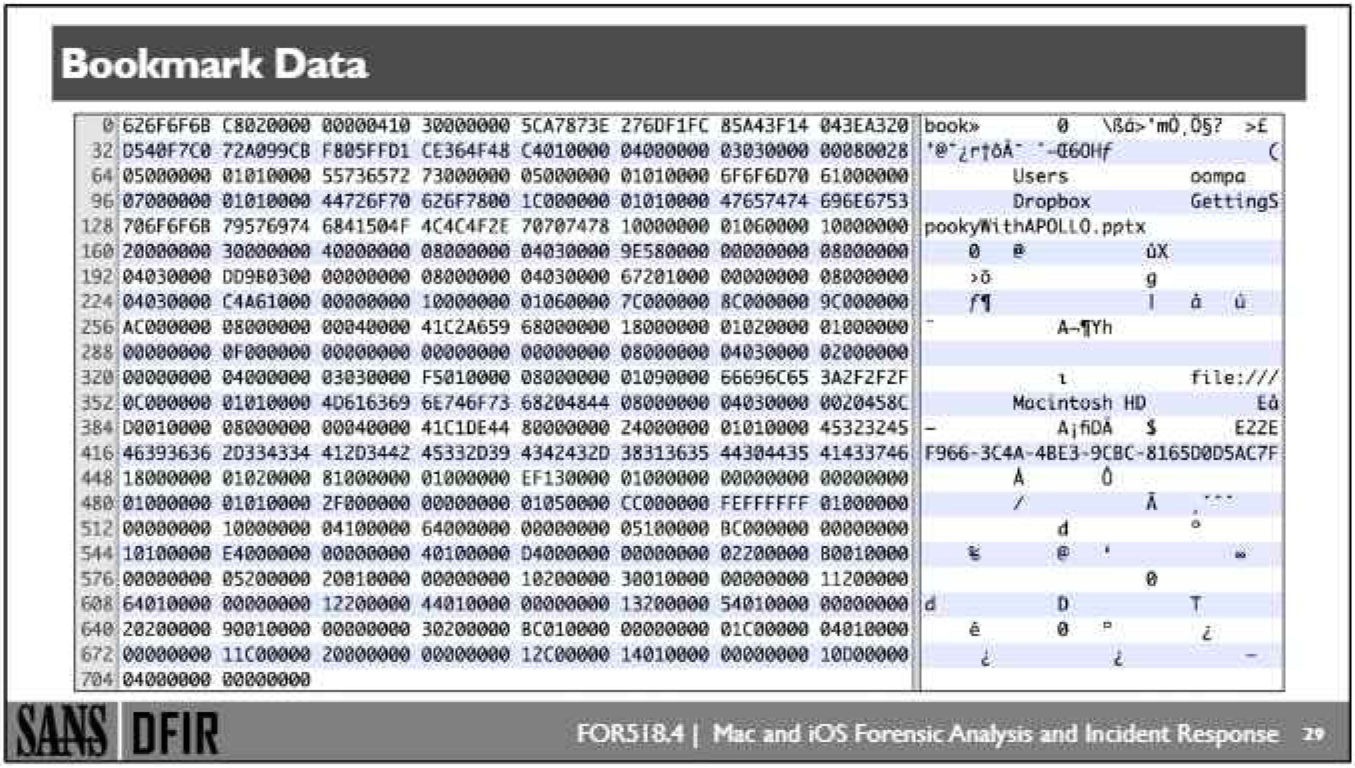


Microsoft Office 365 (and 2016) stored its MRUs in the com.microsoft.<app>.securebookmarks.plist. Each application in the Office suite will have its own set of MRUs.

Each key in these plists is the document path and stores the bookmark data BLOB, a UUID, and a last used timestamp. Older versions did not include the kLastUsedDateKey timestamp.

While most native MacOS, MRUs will only keep the last 10 items (by default); Microsoft Office keeps more!

© 2022 Sarah Edwards



The bookmark data can be extracted and viewed in a hex editor. A bookmark is a reference mechanism that is used to find a file or directory, somewhat similar to Windows Link files.

Each bookmark starts off with the header "`book`" (0x626F6F6B).

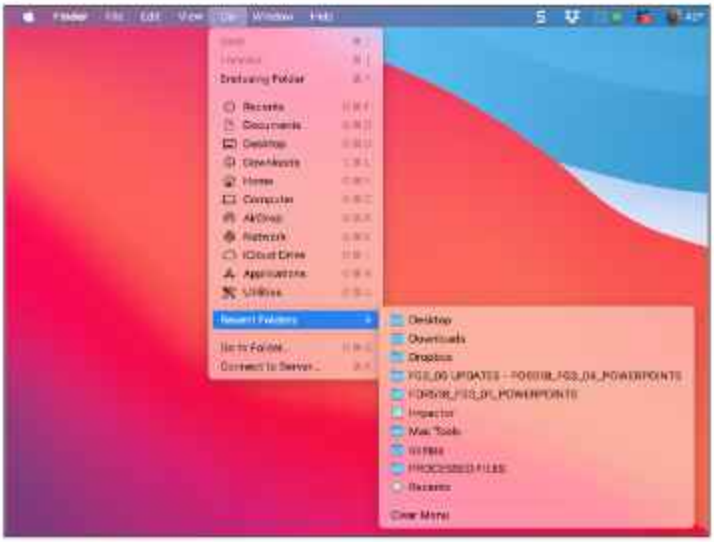
In the screenshot above, various items of interest can be viewed:

- Strings making up the file path (`/Users/oompa/Dropbox/GettingSpookyWithAPOLLO.pptx`)
- Volume Name (`Macintosh HD`)
- Volume GUID (`E22EF966-3C4A-4BE3-9CBC-8165D0D5AC7F`)

Reference:

CFURL:
https://developer.apple.com/documentation/corefoundation/cfurl-rd7

Recent Folders: FXRecentFolders ~/Library/Preferences/com.apple.finder.plist



FXRecentFolders	Array	(10 items)
Item 0	Dictionary	(2 items)
file-bookmark	Data	<626F6F6B6802000000000041030
name	String	Dropbox
Item 1	Dictionary	(2 items)
file-bookmark	Data	<626F6F6BC4020000000000041030
name	String	Mac Tools
Item 2	Dictionary	(2 items)
file-bookmark	Data	<626F6F6B9C020000000000041030
name	String	Downloads
Item 3	Dictionary	(2 items)
file-bookmark	Data	<626F6F6B9005000000000041030
name	String	Impactor
Item 4	Dictionary	(2 items)
file-bookmark	Data	<626F6F6B9802000000000041030
name	String	Desktop
Item 5	Dictionary	(2 items)
file-bookmark	Data	<626F6F6B7002000000000041030
name	String	oompa
Item 6	Dictionary	(2 items)
file-bookmark	Data	<626F6F6B3003000000000041030
name	String	Recents
Item 7	Dictionary	(2 items)

The Recent Folders are located in the `com.apple.finder.plist` in the `Preferences` directory under the FXRecentFolders key.

Each recent Item contains a folder name and a bookmark BLOB. Item 0 is the most recent, while Item 9 is the least recent.

The `file-bookmark` is a file reference using the same bookmark format as before.

Note that the order in the GUI may not necessarily reflect the plist. The plist is more accurate.

© 2022 Sarah Edwards

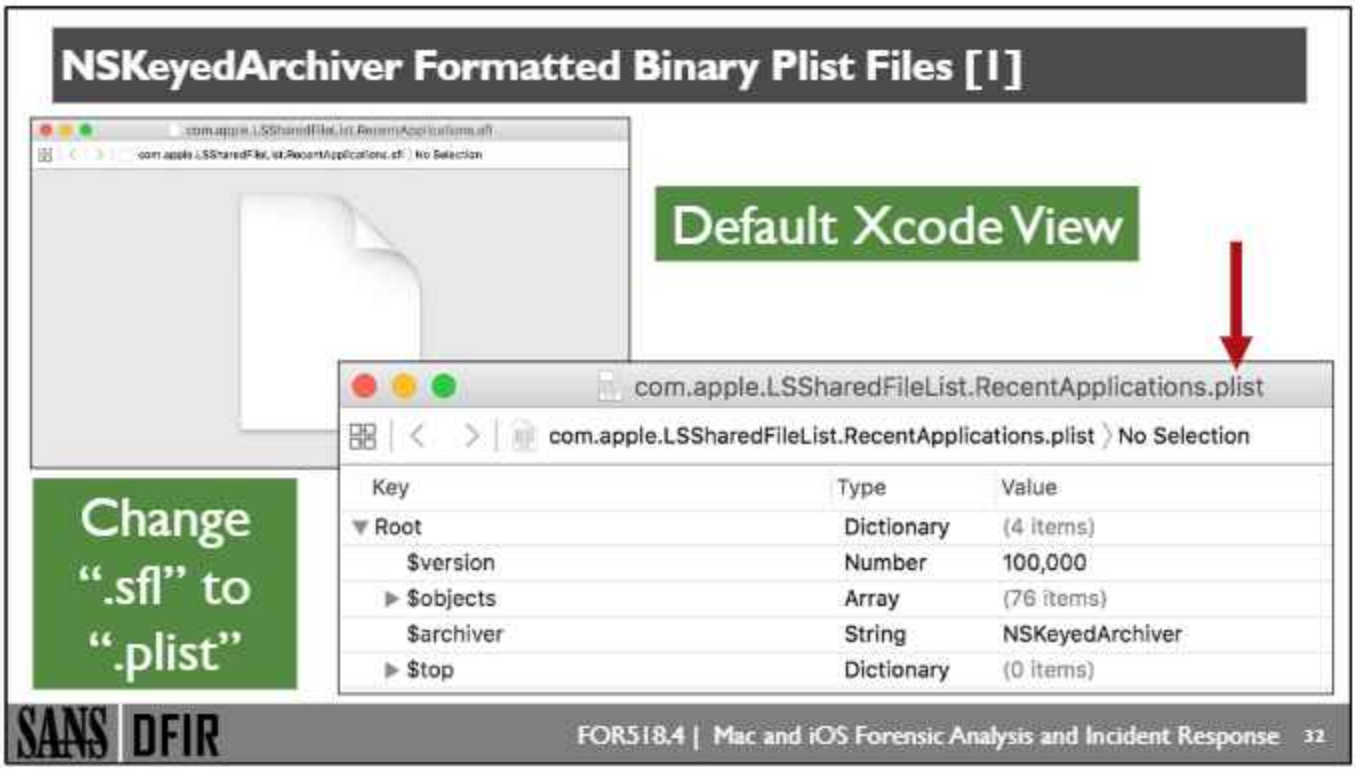
Recent Items per Application, Volumes, Apps, Servers, etc. ~/Library/Application Support/com.apple.sharedfilelist/*.sfl2 [10.13+]

```
oompa@MBP-M1 com.apple.sharedfilelist % ls
com.apple.LSSharedFileList.ApplicationRecentDocuments   com.apple.LSSharedFileList.RecentDocuments.sfl2
com.apple.LSSharedFileList.FavoriteItems.sfl2           com.apple.LSSharedFileList.RecentHosts.sfl2
com.apple.LSSharedFileList.FavoriteVolumes.sfl2         com.apple.LSSharedFileList.RecentServers.sfl2
com.apple.LSSharedFileList.ProjectsItems.sfl2           com.apple.LSSharedFileList.iCloudItems.sfl2
com.apple.LSSharedFileList.RecentApplications.sfl2
oompa@MBP-M1 com.apple.sharedfilelist % ls -l com.apple.LSSharedFileList.ApplicationRecentDocuments
total 168
-rw-r--r--  1 oompa  staff   1477 Jan 30 16:38 com.apple.assistivecontrol.editor.sfl2
-rw-r--r--  1 oompa  staff   2507 Mar  6 19:12 com.apple.dt.xcode.sfl2
-rw-r--r--  1 oompa  staff    420 Mar  2 10:59 com.apple.photos.sfl2
-rw-r--r--  1 oompa  staff  17147 Mar  1 21:34 com.apple.preview.sfl2
-rw-r--r--  1 oompa  staff   2771 Feb  2 20:53 com.apple.screensharing.sfl2
-rw-r--r--  1 oompa  staff   1354 Feb 15 13:19 com.apple.systemprofiler.sfl2
-rw-r--r--  1 oompa  staff  10843 Feb 23 21:04 com.apple.textedit.sfl2
-rw-r--r--  1 oompa  staff    420 Feb 23 21:33 com.microsoft.excel.sfl2
-rw-r--r--  1 oompa  staff    420 Mar  6 10:43 com.microsoft.powerpoint.sfl2
-rw-r--r--  1 oompa  staff    420 Feb 23 20:55 com.microsoft.word.sfl2
-rw-r--r--  1 oompa  staff   5075 Feb  4 06:35 com.ridiculousfish.hexfiend.sfl2
-rw-r--r--  1 oompa  staff  10087 Mar  6 19:12 com.techsmith.snagit2021.sfl2
```


Each application or set of recent items will have its own "SFL" file. Each of these is named by the "reverse DNS" format application bundle ID.

This list will contain both native Apple applications and third-party applications.

© 2022 Sarah Edwards



While there are many plists that use the NSKeyedArchiver format, the Shared File List "SFL" files make a good example of how to parse them.

If we take one of these files and open its directory in Xcode, you will notice it does not show you the plist structure—instead, it shows a blank page icon. This is due to the "`*.sfl`" file extension.

If we change the file extension to "`*.plist`", we can view the contents of the plist; however, due to how this plist is structured, some of the entries are not shown. Depending on the plist viewer, these values may or may not be shown.

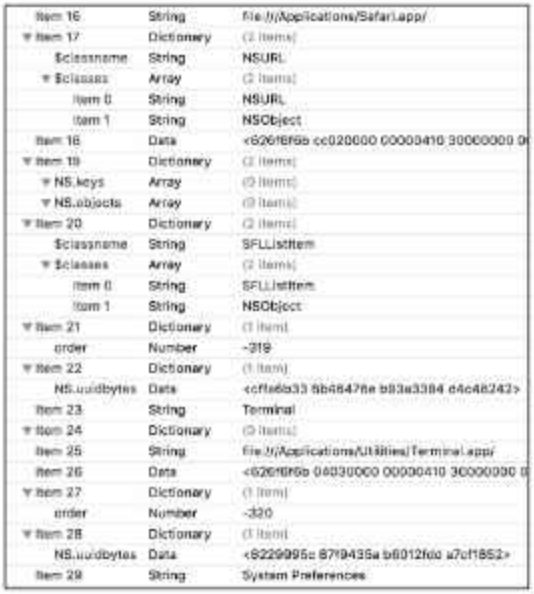
When viewing many plists at a time, you can easily tell you have one of these NSKeyedArchiver formatted plists by the first few keys:

- `$version`
- `$objects`
- `$archiver` with the value "`NSKeyedArchiver`"
- `$top`

© 2022 Sarah Edwards

NSKeyedArchiver Formatted Binary Plist Files [2]

No Context for Keys and Values



Item 16	String	file:///Applications/Safari.app/
▼ Item 17	Dictionary	(2 items)
$classname	String	NSURL
▼ $classes	Array	(2 items)
Item 0	String	NSURL
Item 1	String	NSObject
Item 18	Data	<626f6f6b cc020000 00000410 30000000 0
▼ Item 19	Dictionary	(2 items)
▼ NS.keys	Array	(0 items)
▼ NS.objects	Array	(0 items)
▼ Item 20	Dictionary	(2 items)
$classname	String	SFLListItem
▼ $classes	Array	(2 items)
Item 0	String	SFLListItem
Item 1	String	NSObject
▼ Item 21	Dictionary	(1 item)
order	Number	-319
▼ Item 22	Dictionary	(1 item)
NS.uuidbytes	Data	<cf1e6b33 5b4647fe b83a3384 d4c48242>
Item 23	String	Terminal
▼ Item 24	Dictionary	(0 items)
Item 25	String	file:///Applications/Utilities/Terminal.app/
Item 26	Data	<626f6f6b 04030000 00000410 30000000 0
▼ Item 27	Dictionary	(1 item)
order	Number	-320
▼ Item 28	Dictionary	(1 item)
NS.uuidbytes	Data	<8229995c 87f9435a b6012fdd a7cf1852>
Item 29	String	System Preferences

If we take a look at one these files, we can see items that appear to be of interest. The problem with this format of plist files is that there is no outright context put into the keys and values.

© 2022 Sarah Edwards

Item 16	String	file:///Applications/Safari.app/
▼ Item 17	Dictionary	(2 items)
$classname	String	NSURL
▼ $classes	Array	(2 items)
Item 0	String	NSURL
Item 1	String	NSObject
Item 18	Data	<626f6f6b cc020000 00000410 30000000 0(
▼ Item 19	Dictionary	(2 items)
▼ NS.keys	Array	(0 items)
▼ NS.objects	Array	(0 items)
▼ Item 20	Dictionary	(2 items)
$classname	String	SFLListItem
▼ $classes	Array	(2 items)
Item 0	String	SFLListItem
Item 1	String	NSObject
▼ Item 21	Dictionary	(1 item)
order	Number	-319
▼ Item 22	Dictionary	(1 item)
NS.uuidbytes	Data	<cffa6b33 6b46476e b93a3384 d4c48242>
Item 23	String	Terminal
▼ Item 24	Dictionary	(0 items)
Item 25	String	file:///Applications/Utilities/Terminal.app/
Item 26	Data	<626f6f6b 04030000 00000410 30000000 0
▼ Item 27	Dictionary	(1 item)
order	Number	-320
▼ Item 28	Dictionary	(1 item)
NS.uuidbytes	Data	<6229995c 87f9435a b6012fdd a7cf1852>
Item 29	String	System Preferences

© 2022 Sarah Edwards

NSKeyedArchiver Formatted Binary Plist Files [3]

```
{
  "$version" => 100000
  "$objects" => [
    0 => "$null"
    1 => {
      "NS.keys" => [
        0 => <CFKeyedArchiverUID 0x7ff370401bb0 [0x7fff760dd390]>{value = 2}
        1 => <CFKeyedArchiverUID 0x7ff370401bd0 [0x7fff760dd390]>{value = 3}
        2 => <CFKeyedArchiverUID 0x7ff370401bf0 [0x7fff760dd390]>{value = 4}
      ]
      "NS.objects" => [
        0 => <CFKeyedArchiverUID 0x7ff370401c60 [0x7fff760dd390]>{value = 5}
        1 => <CFKeyedArchiverUID 0x7ff370401c80 [0x7fff760dd390]>{value = 6}
        2 => <CFKeyedArchiverUID 0x7ff370401ca0 [0x7fff760dd390]>{value = 10}
      ]
      "$class" => <CFKeyedArchiverUID 0x7ff370401d10 [0x7fff760dd390]>{value = 9}
    }
    2 => "version"
    3 => "properties"
    4 => "items"
    5 => 1
    6 => {
      "NS.keys" => [
        0 => <CFKeyedArchiverUID 0x7ff370401cb0 [0x7fff760dd390]>{value = 7}
      ]
      "NS.objects" => [
        0 => <CFKeyedArchiverUID 0x7ff370401d10 [0x7fff760dd390]>{value = 8}
      ]
      "$class" => <CFKeyedArchiverUID 0x7ff370401d10 [0x7fff760dd390]>{value = 9}
    }
    7 => "com.apple.LSSharedFileList.MaxAmount"
    8 => 10
    9 => {
      "$classname" => "NSDictionary"
      "$classes" => [
        0 => "NSDictionary"
        1 => "NSObject"
      ]
    }
```

Output from plutil -p

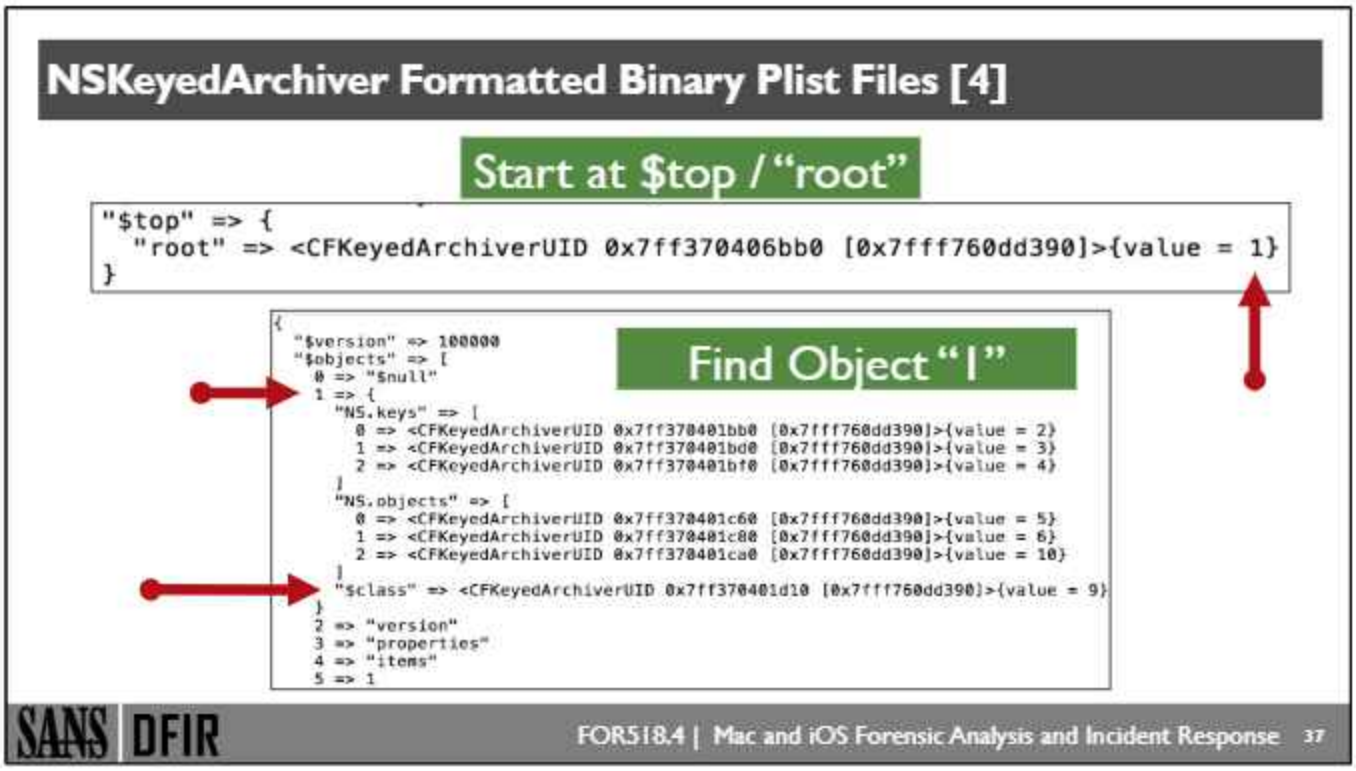
If your plist viewer shows all the values needed, feel free to use it! If it doesn't, we can use the output from the "`plutil -p <file>`" command.

This output shows the values in a somewhat human readable format in JSON style output.

© 2022 Sarah Edwards

```
{
  "$version" => 100000
  "$objects" => [
    0 => "$null"
    1 => {
      "NS.keys" => [
        0 => <CFKeyedArchiverUID 0x7ff370401bb0 [0x7fff760dd390]>{value = 2}
        1 => <CFKeyedArchiverUID 0x7ff370401bd0 [0x7fff760dd390]>{value = 3}
        2 => <CFKeyedArchiverUID 0x7ff370401bf0 [0x7fff760dd390]>{value = 4}
      ]
      "NS.objects" => [
        0 => <CFKeyedArchiverUID 0x7ff370401c60 [0x7fff760dd390]>{value = 5}
        1 => <CFKeyedArchiverUID 0x7ff370401c80 [0x7fff760dd390]>{value = 6}
        2 => <CFKeyedArchiverUID 0x7ff370401ca0 [0x7fff760dd390]>{value = 10}
      ]
      "$class" => <CFKeyedArchiverUID 0x7ff370401d10 [0x7fff760dd390]>{value = 9}
    }
    2 => "version"
    3 => "properties"
    4 => "items"
    5 => 1
    6 => {
      "NS.keys" => [
        0 => <CFKeyedArchiverUID 0x7ff370401db0 [0x7fff760dd390]>{value = 7}
      ]
      "NS.objects" => [
        0 => <CFKeyedArchiverUID 0x7ff370401e10 [0x7fff760dd390]>{value = 8}
      ]
      "$class" => <CFKeyedArchiverUID 0x7ff370401d10 [0x7fff760dd390]>{value = 9}
    }
    7 => "com.apple.LSSharedFileList.MaxAmount"
    8 => 10
    9 => {
      "$classname" => "NSDictionary"
      "$classes" => [
        0 => "NSDictionary"
        1 => "NSObject"
      ]
    }
```


© 2022 Sarah Edwards



To start parsing these plist files, we need to start at the `$top` key. This should hold the "`root`" of the plist tree. In the example above, "`root`" points to the value "`1`".

In the `$objects` section, we will find the "`1`" value, as noted by the arrow above. This key holds (sub)keys and objects. This particular format has a key = value format as shown below.

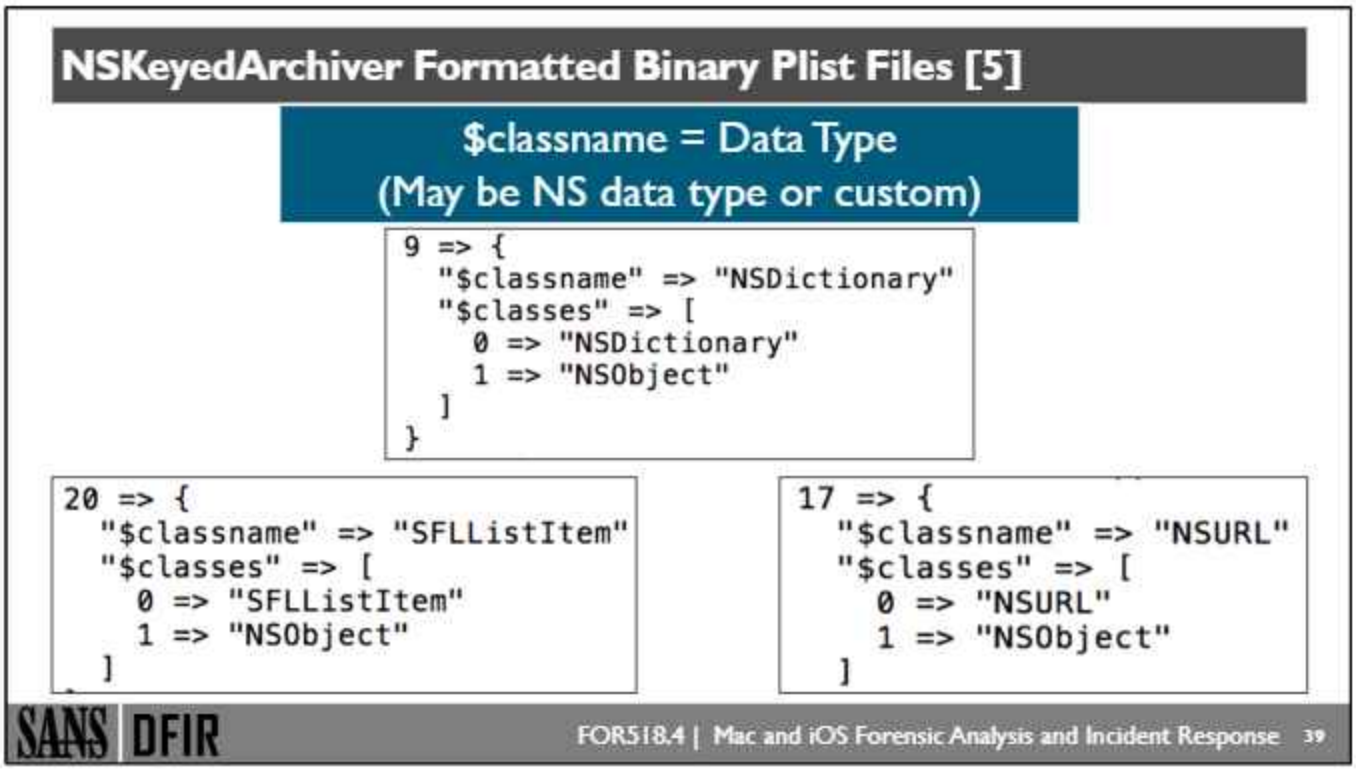
```
Key = Object
2 = 5
3 = 6
4 = 10
```

You will notice that certain "`objects`" have a `$class` key. This describes a bit about the values in this object. Some examples are in the next slide.

© 2022 Sarah Edwards

```
{
  "$version" => 100000
  "$objects" => [
    0 => "$null"
    1 => {
      "NS.keys" => [
        0 => <CFKeyedArchiverUID 0x7ff370401bb0 [0x7fff760dd390]>{value = 2}
        1 => <CFKeyedArchiverUID 0x7ff370401bd0 [0x7fff760dd390]>{value = 3}
        2 => <CFKeyedArchiverUID 0x7ff370401bf0 [0x7fff760dd390]>{value = 4}
      ]
      "NS.objects" => [
        0 => <CFKeyedArchiverUID 0x7ff370401c60 [0x7fff760dd390]>{value = 5}
        1 => <CFKeyedArchiverUID 0x7ff370401c80 [0x7fff760dd390]>{value = 6}
        2 => <CFKeyedArchiverUID 0x7ff370401ca0 [0x7fff760dd390]>{value = 10}
      ]
      "$class" => <CFKeyedArchiverUID 0x7ff370401d10 [0x7fff760dd390]>{value = 9}
    }
    2 => "version"
    3 => "properties"
    4 => "items"
    5 => 1
```

© 2022 Sarah Edwards



In the previous example, the `$class` referred to object "9", or an `NSDictionary` data type.

Other examples may include other `NSData` types, like `NSURL`, `NSDate`, `NSUUID`, or `NSArray`.

Other "custom" types may also be available. In the example plist, we have a `$classname` "`SFLListItem`" describing the recent item.

NSKeyedArchiver Formatted Binary Plist Files [6]

```
2 => "version"
3 => "properties"
4 => "items"
5 => 1
6 => {
  "NS.keys" => [
    0 => <CFKeyedArchiverUID 0x7ff370401db0 [0x7fff760dd398]>{value = 7}
  ]
  "NS.objects" => [
    0 => <CFKeyedArchiverUID 0x7ff370401e10 [0x7fff760dd398]>{value = 8}
  ]
  "$class" => <CFKeyedArchiverUID 0x7ff370401d10 [0x7fff760dd398]>{value = 9}
}
7 => "com.apple.LSSharedFileList.MaxAmount"
8 => 10
9 => {
  "$classname" => "NSDictionary"
  "$classes" => [
    0 => "NSDictionary"
    1 => "NSObject"
  ]
}
10 => {
  "NS.objects" => [
    0 => <CFKeyedArchiverUID 0x7ff370401ff0 [0x7fff760dd398]>{value = 11}
    1 => <CFKeyedArchiverUID 0x7ff3704020l0 [0x7fff760dd398]>{value = 21}
    2 => <CFKeyedArchiverUID 0x7ff370402030 [0x7fff760dd398]>{value = 27}
    3 => <CFKeyedArchiverUID 0x7ff370402050 [0x7fff760dd398]>{value = 33}
    4 => <CFKeyedArchiverUID 0x7ff370402070 [0x7fff760dd398]>{value = 39}
    5 => <CFKeyedArchiverUID 0x7ff370402090 [0x7fff760dd398]>{value = 45}
    6 => <CFKeyedArchiverUID 0x7ff3704020b0 [0x7fff760dd398]>{value = 51}
    7 => <CFKeyedArchiverUID 0x7ff3704020d0 [0x7fff760dd398]>{value = 57}
    8 => <CFKeyedArchiverUID 0x7ff3704020f0 [0x7fff760dd398]>{value = 63}
    9 => <CFKeyedArchiverUID 0x7ff370402110 [0x7fff760dd398]>{value = 69}
  ]
  "$class" => <CFKeyedArchiverUID 0x7ff3704021b0 [0x7fff760dd398]>{value = 75}
}
```


Continuing with our example, we can see that keys "2", "3", and "4" are associated with "`version`", "`properties`", and "`items`", respectively. The "`items`" key sounds most promising. In this case, let's follow this one down the line.

Previously, we saw that value "4" is associated with value "`10`". Value "`10`" shown above contains an array of objects: `11, 21, 27, 33, 39, 45, 51, 57, 63, 69`; 10 items! This sounds like our Recent Applications.

Let's look at recent item "`11`" …

© 2022 Sarah Edwards

```
2 => "version"
3 => "properties"
4 => "items"
5 => 1
6 => {
  "NS.keys" => [
    0 => <CFKeyedArchiverUID 0x7ff370401db0 [0x7fff760dd390]>{value = 7}
  ]
  "NS.objects" => [
    0 => <CFKeyedArchiverUID 0x7ff370401e10 [0x7fff760dd390]>{value = 8}
  ]
  "$class" => <CFKeyedArchiverUID 0x7ff370401d10 [0x7fff760dd390]>{value = 9}
}
7 => "com.apple.LSSharedFileList.MaxAmount"
8 => 10
9 => {
  "$classname" => "NSDictionary"
  "$classes" => [
    0 => "NSDictionary"
    1 => "NSObject"
  ]
}
10 => {
  "NS.objects" => [
    0 => <CFKeyedArchiverUID 0x7ff370401ff0 [0x7fff760dd390]>{value = 11}
    1 => <CFKeyedArchiverUID 0x7ff370402010 [0x7fff760dd390]>{value = 21}
    2 => <CFKeyedArchiverUID 0x7ff370402030 [0x7fff760dd390]>{value = 27}
    3 => <CFKeyedArchiverUID 0x7ff370402050 [0x7fff760dd390]>{value = 33}
    4 => <CFKeyedArchiverUID 0x7ff370402070 [0x7fff760dd390]>{value = 39}
    5 => <CFKeyedArchiverUID 0x7ff370402090 [0x7fff760dd390]>{value = 45}
    6 => <CFKeyedArchiverUID 0x7ff3704020b0 [0x7fff760dd390]>{value = 51}
    7 => <CFKeyedArchiverUID 0x7ff3704020d0 [0x7fff760dd390]>{value = 57}
    8 => <CFKeyedArchiverUID 0x7ff3704020f0 [0x7fff760dd390]>{value = 63}
    9 => <CFKeyedArchiverUID 0x7ff370402110 [0x7fff760dd390]>{value = 69}
  ]
  "$class" => <CFKeyedArchiverUID 0x7ff3704021b0 [0x7fff760dd390]>{value = 75}
}
```

Item "11" contains a variety of keys:

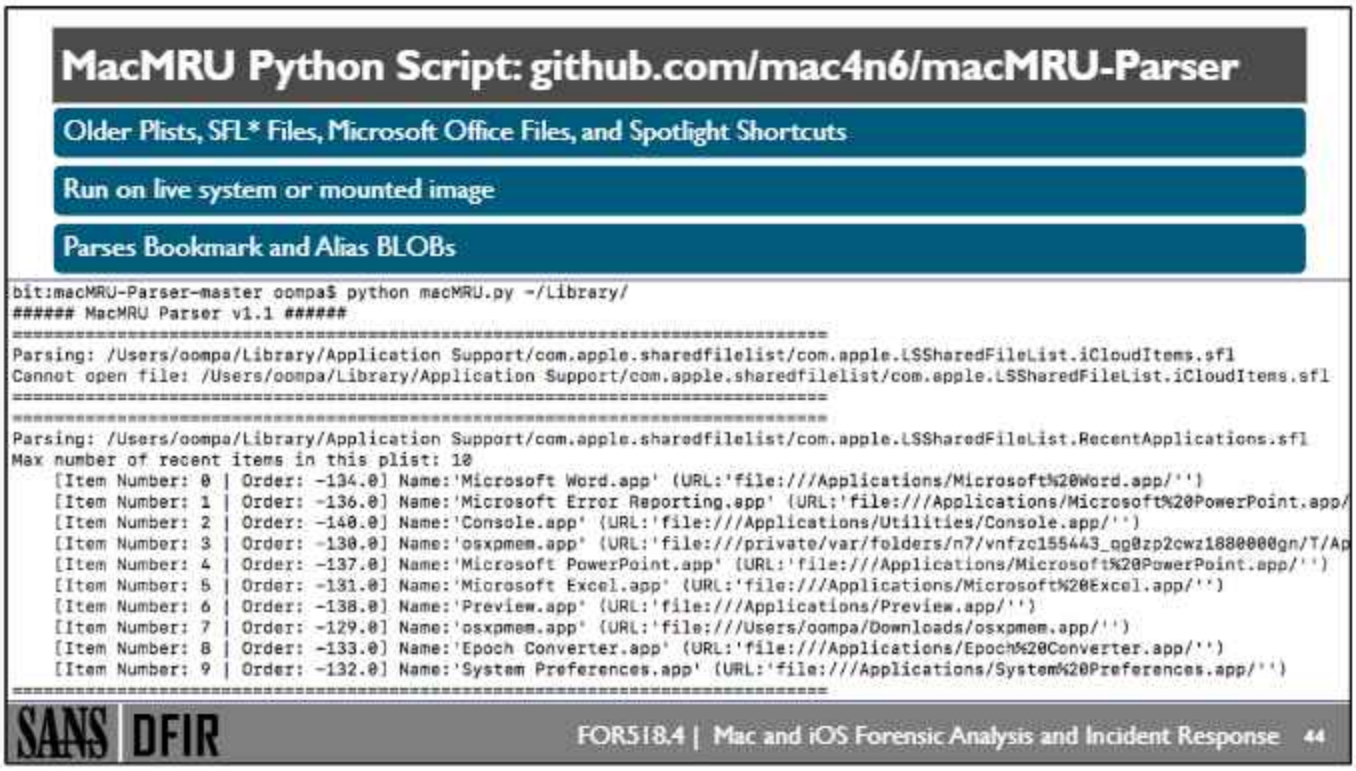
- `$class` = `value` = `20` (not shown: "`SFLListItem`")
- `order` = `-146`
  - Research is ongoing in this area, as some plists have both positive and negative numbers. It does not appear consistent which items are older or newer.
- `bookmark` = `value 18:` This is the binary BLOB shown in the example. We can extract this to look at the bookmark data for this recent application.
- `uniqueIdentifier` = value `12`: This is an `NS.uuidbytes`, or GUID, for this recent application.
- `properties` = value `19`: (now shown, but empty)
- `URL` = value `15`: The path to the application (`file:///Applications/System%20Preferences.app/`)
- `name` = value `14:` Application Name "`System Preferences`"

Most of this information is the same data we found in previous MRU plist files—just in a slightly different, more convoluted format.

© 2022 Sarah Edwards

```
    11 => {
      "$class" => <CFKeyedArchiverUID 0x7ff370402280 [0x7fff760dd390]>{value = 20}
      "order" => -146
      "bookmark" => <CFKeyedArchiverUID 0x7ff3704022a0 [0x7fff760dd390]>{value = 18}
      "uniqueIdentifier" => <CFKeyedArchiverUID 0x7ff3704022c0 [0x7fff760dd390]>{value = 12}
      "properties" => <CFKeyedArchiverUID 0x7ff3704022e0 [0x7fff760dd390]>{value = 19}
      "URL" => <CFKeyedArchiverUID 0x7ff370402300 [0x7fff760dd390]>{value = 15}
      "name" => <CFKeyedArchiverUID 0x7ff370402320 [0x7fff760dd390]>{value = 14}
    }
    12 => {
      "NS.uuidbytes" => <63c56a16 3b014cdf 9de42fab 8a616942>
      "$class" => <CFKeyedArchiverUID 0x7ff3704024d0 [0x7fff760dd390]>{value = 13}
    }
    13 => {
      "$classname" => "NSUUID"
      "$classes" => [
        0 => "NSUUID"
        1 => "NSObject"
      ]
    }
    14 => "System Preferences"
    15 => {
      "NS.base" => <CFKeyedArchiverUID 0x7ff370402650 [0x7fff760dd390]>{value = 0}
      "NS.relative" => <CFKeyedArchiverUID 0x7ff370402690 [0x7fff760dd390]>{value = 16}
      "$class" => <CFKeyedArchiverUID 0x7ff370402670 [0x7fff760dd390]>{value = 17}
    }
    16 => "file:///Applications/System%20Preferences.app/"
    17 => {
      "$classname" => "NSURL"
      "$classes" => [
        0 => "NSURL"
        1 => "NSObject"
      ]
    }
    18 => <626f6f6b e4020000 00000410 30000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
04020000 0c000000 01010000 4170706c 69636174 696f6e73 16000000 01010000 53797374 656d2050 72656665 72656e63 65732e61
70700000 08000000 01060000 04000000 18000000 08000000 04030000 47000000 00000000 08000000 04030000 2ca70000 00000000
08000000 01060000 48000000 58000000 08000000 00040000 41bb8965 13000000 18000000 01020000 02000000 00000000 0f000000
00000000 00000000 00000000 08000000 01090000 66696c65 3a2f2f2f 0a000000 01010000 456c2043 61707469 616e0000 08000000
04030000 00000032 74000000 08000000 00040000 41bbf944 dc000000 24000000 01010000 44363942 44313038 2d463934 322d3330
46372d42 4535422d 34413644 43343134 35463336 18000000 01020000 81000000 01000000 ef130000 01000000 00000000 00000000
01000000 01010000 2f000000 00000000 01050000 af000000 01020000 65376461 65303539 39643635 31353263 37366266 33643266
35373966 38643166 32633137 66343231 3b303030 30303030 303b3030 30303030 30303b30 30303030 30303030 30303030 3031613b
636f6d2e 6170706c 652e6170 702d7361 6e64626f 782e7265 61643b30 30303030 3030313b 30313030 30303034 3b303030 30303030
30303030 30613732 633b2f61 70706c69 63617469 6f6e732f 73797374 656d2070 72656665 72656e63 65732e61 70700000 a8000000
feffffff 01000000 00000000 0d000000 04100000 38000000 00000000 05100000 68000000 00000000 10100000 88000000 00000000
40100000 78000000 00000000 02200000 38010000 00000000 05200000 a8000000 00000000 10200000 b8000000 00000000 11200000
ec000000 00000000 12200000 cc000000 00000000 13200000 dc000000 00000000 20200000 18010000 00000000 30200000 44010000
00000000 81f00000 4c010000 00000000>
```

© 2022 Sarah Edwards

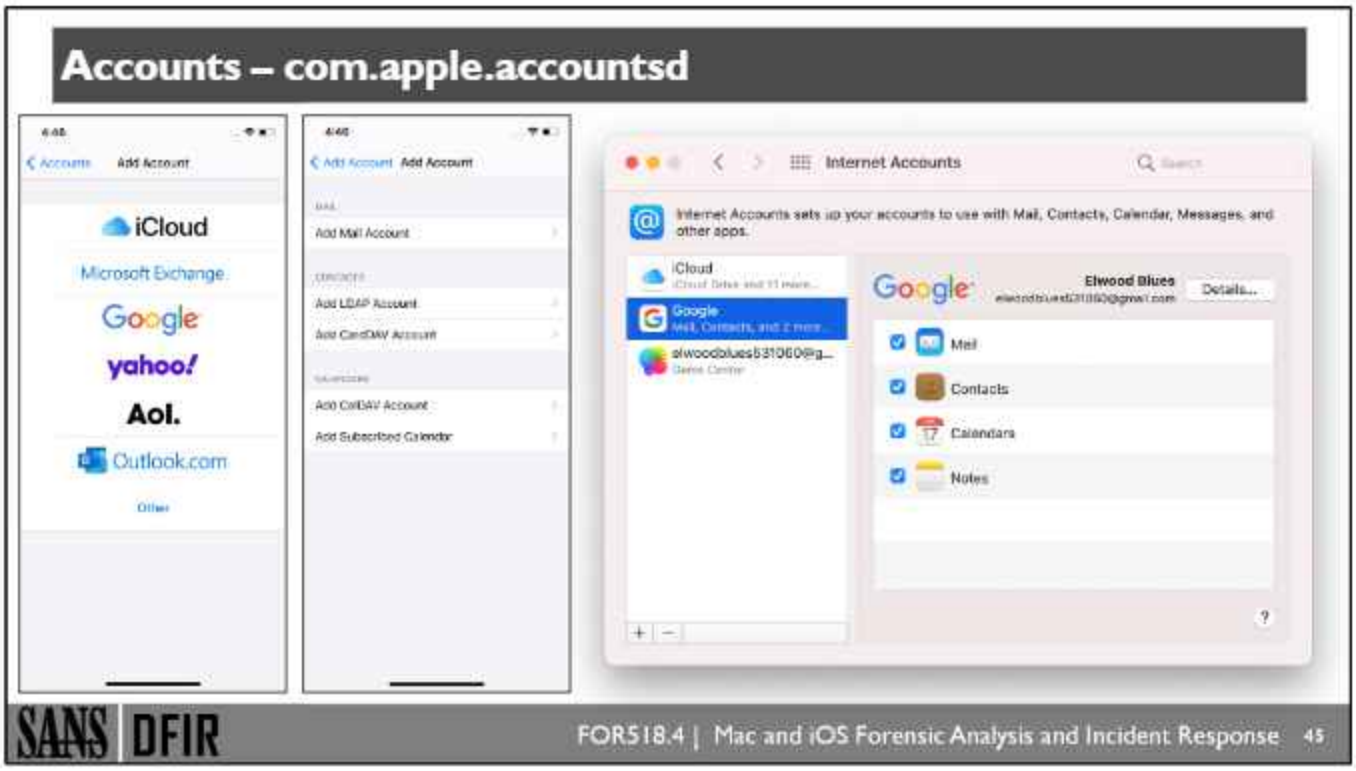


A script was created by the course author to deal with these pesky new Mac MRU plist files. It should make an analysis of these files easier on the analyst.

This Python script can be downloaded from https://github.com/mac4n6/macMRU-Parser

References:
https://www.mac4n6.com/blog/2016/7/10/new-script-macmru-most-recently-used-plist-parser
https://www.mac4n6.com/blog/2016/8/15/update-to-macmru-parser-now-with-microsoft-office-support
https://www.mac4n6.com/blog/2017/7/19/script-update-mac-mru-parser-spotlight-shortcuts-blob-parsing

© 2022 Sarah Edwards



Mac:

- 10.11: `~/Library/Accounts/Accounts3.sqlite`
- 10.12+: `~/Library/Accounts/Accounts4.sqlite`

iOS: `[/private/var]/mobile/Library/Accounts/Accounts3.sqlite`

Knowing what email, calendar, iCloud accounts a user has configured with their device can be good investigative information to know. Mac and iOS devices both have a single area of configured account information that may include email, and/or calendar accounts. The configuration elements for these accounts are stored in a SQLite database but one plist file can be used to perform a quick triage of available accounts.

The `com.apple.accounts.exists.plist` file is located in the `/preferences/SystemConfiguration/` directory for backups, file system extractions, and physical images.This property list shows which types of accounts are globally configured on the device. These can include email (IMAP/SMTP/Exchange), Google, Calendar, Apple, etc. Each account has two associated keys: a "`Count`" key and an "`Exists`" key.

The "`Exists`" key shows if that particular type of account is in use. This can have a few different options:

1. At least one account is set up for this type
2. No accounts of this type

The "`Count`" key shows how many accounts of this type there are.

© 2022 Sarah Edwards

Internet Accounts - Accounts3.sqlite | Accounts4.sqlite [1]

```
SELECT
ZACCOUNT.Z_PK,ZACCOUNTTYPE.ZACCOUNTTYPEDESCRIPTION,ZACCOUNT.ZIDENTIFIER,ZACCOUNT.ZUSERNAME,
ZACCOUNT.ZACCOUNTDESCRIPTION,ZACCOUNTPROPERTY.ZKEY,ZACCOUNTPROPERTY.ZVALUE
FROM ZACCOUNTPROPERTY
LEFT JOIN ZACCOUNT ON ZACCOUNT.Z_PK == ZACCOUNTPROPERTY.ZOWNER
LEFT JOIN ZACCOUNTTYPE ON ZACCOUNT.ZACCOUNTTYPE == ZACCOUNTTYPE.Z_PK
order by ZACCOUNT.ZIDENTIFIER
```

	Z_PK	ZACCOUNTTYPEDESCRIPTION	ZIDENTIFIER	ZUSERNAME	ZACCOUNTDESCRIPTION	ZKEY	ZVALUE
1	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	handles	BLOB
2	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	account-info	BLOB
3	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	apple-id	BLOB
4	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	self-handle	BLOB
5	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	invitation-context	BLOB
6	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	auth-token-receipt-date	BLOB
7	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	add-timestamp	BLOB
8	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	email-address	BLOB
9	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	profile-id	BLOB
10	4	Messages	05E7C21C-8F6E-4258-981C-9E396C14C178	elwoodblues531050@gmail.com	NULL	bundleIdof	BLOB
11	17	IMAP	38330C55-06CA-4577-9362-79B737D5593D	NULL	NULL	EmailAliases	BLOB
12	17	IMAP	38330C55-06CA-4577-9362-79B737D5593D	NULL	NULL	SendingAccountIdentifier	BLOB
13	17	IMAP	38330C55-06CA-4577-9362-79B737D5593D	NULL	NULL	SSLEnabled	BLOB
14	17	IMAP	38330C55-06CA-4577-9362-79B737D5593D	NULL	NULL	AuthenticationScheme	BLOB
15	17	IMAP	38330C55-06CA-4577-9362-79B737D5593D	NULL	NULL	Hostname	BLOB
16	17	IMAP	38330C55-06CA-4577-9362-79B737D5593D	NULL	NULL	AllowsInsecureAuthentication	BLOB
17	17	IMAP	38330C55-06CA-4577-9362-79B737D5593D	NULL	NULL	PortNumber	BLOB
18	17	IMAP	38330C55-06CA-4577-9362-79B737D5593D	NULL	NULL	SSLIsDirect	BLOB
19	6	iCloud	3FEFF8BF-3C88-4591-A833-7C3E338F7227	elwoodblues531050@gmail.com	iCloud	altDSID	BLOB
20	6	iCloud	3FEFF8BF-3C88-4591-A833-7C3E338F7227	elwoodblues531050@gmail.com	iCloud	cloudDocsMigrationComplete	BLOB
21	6	iCloud	3FEFF8BF-3C88-4591-A833-7C3E338F7227	elwoodblues531050@gmail.com	iCloud	accountClass	BLOB

46

Mac:

- 10.11: `~/Library/Accounts/Accounts3.sqlite`
- 10.12+: `~/Library/Accounts/Accounts4.sqlite`

iOS: `[/private/var]/mobile/Library/Accounts/Accounts3.sqlite`

The `Accounts3.sqlite or Accounts4.sqlite` database keeps track of which accounts (Mail, Messages, Communication, Calendar, Social Media, etc.,) are integrated into the Mac or iOS system.

The `ZACCOUNTTYPE` table contains the types of accounts that can be configured globally on the device. The `Z_PK` column contains the identification number for each account. This table also includes the account description, credential types, and the identifier for the account.

The `ZACCOUNT` table contains specific user account information. We can match up the `Z_PK` key from the `ZACCOUNTTYPE` table with the `ZACCOUNTTYPE` column of this table to determine what account this information is associated with.

The `ZACCOUNT` table in the example above shows the following columns of interest:

- `Z_PK` = Primary Key
- `ZACCOUNTTYPEDESCRIPTION` from `ZACCOUNTTYPE` Table = Type of Account
- `ZUSERNAME` = Account Username
- `ZACCOUNTDESCRIPTION` from `ZACCOUNT` Table = Account Description, more specific
- `ZPARENTACCOUNT` = Parent Account Type
- `ZDATE` = Account Setup Timestamp, Mac Epoch timestamp format (labeled in the query as "Timestamp")
- `ZIDENTIFIER` = Account GUID
- ZKEY = Configuration Key
- ZVALUE = Configuration Value

© 2022 Sarah Edwards

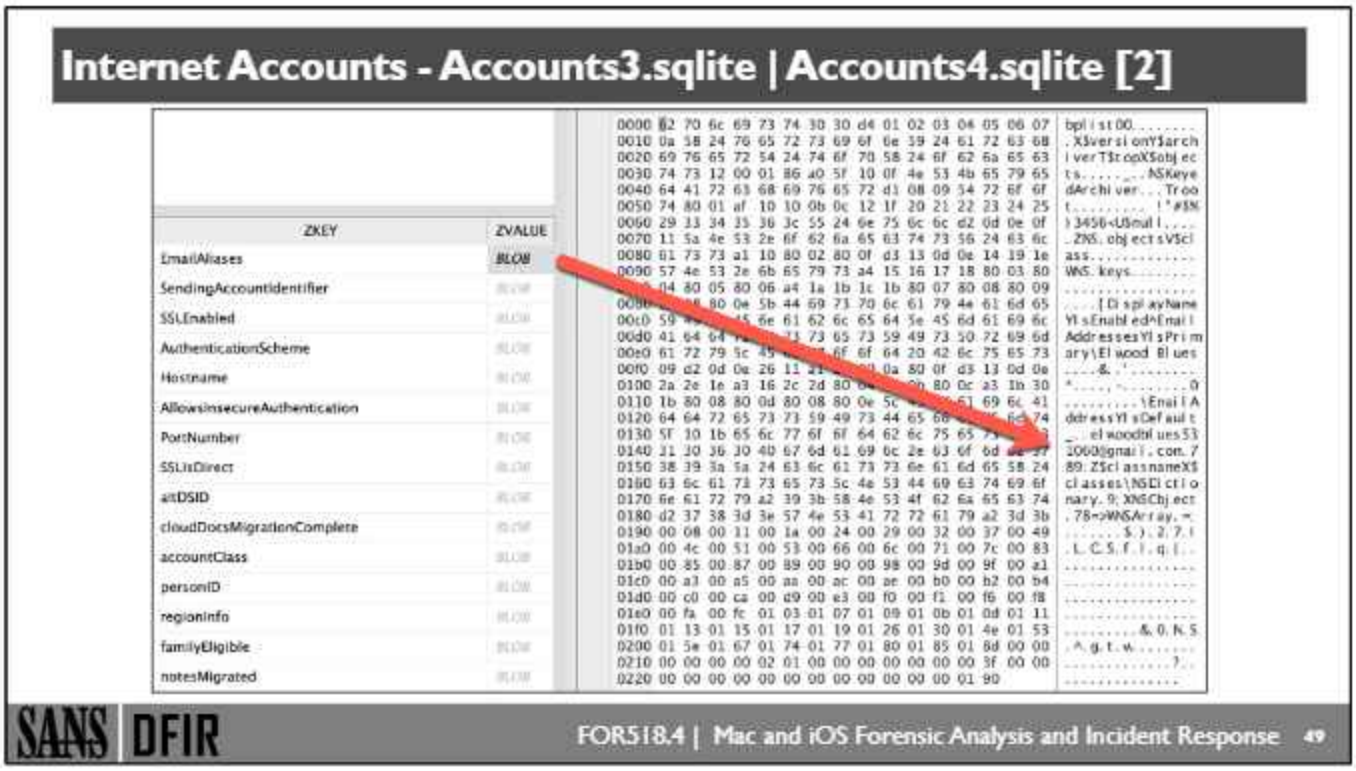
```
SELECT
ZACCOUNT.Z_PK,ZACCOUNTTYPE.ZACCOUNTTYPEDESCRIPTION,ZACCOUNT.ZIDENTIFIER,ZACCOUNT.ZUSERNAME,
ZACCOUNT.ZACCOUNTDESCRIPTION,ZACCOUNTPROPERTY.ZKEY,ZACCOUNTPROPERTY.ZVALUE
FROM ZACCOUNTPROPERTY
LEFT JOIN ZACCOUNT ON ZACCOUNT.Z_PK == ZACCOUNTPROPERTY.ZOWNER
LEFT JOIN ZACCOUNTTYPE ON ZACCOUNT.ZACCOUNTTYPE == ZACCOUNTTYPE.Z_PK
order by ZACCOUNT.ZIDENTIFIER
```

	Z_PK	ZACCOUNTTYPEDESCRIPTION	ZIDENTIFIER	ZUSERNAME	ZACCOUNTDESCRIPTION	ZKEY	ZVALUE
1	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	handles	BLOB
2	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	account-info	BLOB
3	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	apple-id	BLOB
4	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	self-handle	BLOB
5	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	invitation-context	BLOB
6	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	auth-token-receipt-date	BLOB
7	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	add-timestampe	BLOB
8	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	email-address	BLOB
9	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	profile-id	BLOB
10	4	Messages	05E7C21C-8F6E-425B-9B1C-9E396C14C178	elwoodblues531060@gmail.com	NULL	bundleRef	BLOB
11	17	IMAP	38330C55-06CA-45F7-93A1-73B737D5931D	NULL	NULL	EmailAliases	BLOB
12	17	IMAP	38330C55-06CA-45F7-93A1-73B737D5931D	NULL	NULL	SendingAccountIdentifier	BLOB
13	17	IMAP	38330C55-06CA-45F7-93A1-73B737D5931D	NULL	NULL	SSLEnabled	BLOB
14	17	IMAP	38330C55-06CA-45F7-93A1-73B737D5931D	NULL	NULL	AuthenticationScheme	BLOB
15	17	IMAP	38330C55-06CA-45F7-93A1-73B737D5931D	NULL	NULL	Hostname	BLOB
16	17	IMAP	38330C55-06CA-45F7-93A1-73B737D5931D	NULL	NULL	AllowsInsecureAuthentication	BLOB
17	17	IMAP	38330C55-06CA-45F7-93A1-73B737D5931D	NULL	NULL	PortNumber	BLOB
18	17	IMAP	38330C55-06CA-45F7-93A1-73B737D5931D	NULL	NULL	SSLIsDirect	BLOB
19	6	ICloud	3FEFF8EF-3CE8-4591-A839-7C3E398F7227	elwoodblues531060@gmail.com	iCloud	altDSID	BLOB
20	6	iCloud	3FEFF8EF-3CE8-4591-A839-7C3E398F7227	elwoodblues531060@gmail.com	iCloud	cloudDocsMigrationComplete	BLOB
21	6	ICloud	3FEFF8EF-3CE8-4591-A839-7C3E398F7227	elwoodblues531060@gmail.com	iCloud	accountClass	BLOB

© 2022 Sarah Edwards

The highlighted `ZIDENTIFIER GUID` is the email account that was previously shown in the Mail section. The keys and values for each configuration element provide different types of information such as usernames, account ids, servers, ports, etc. The values are embedded `binary plist files.` Different accounts will have different types of configuration data that is named by the `ZKEY` column.

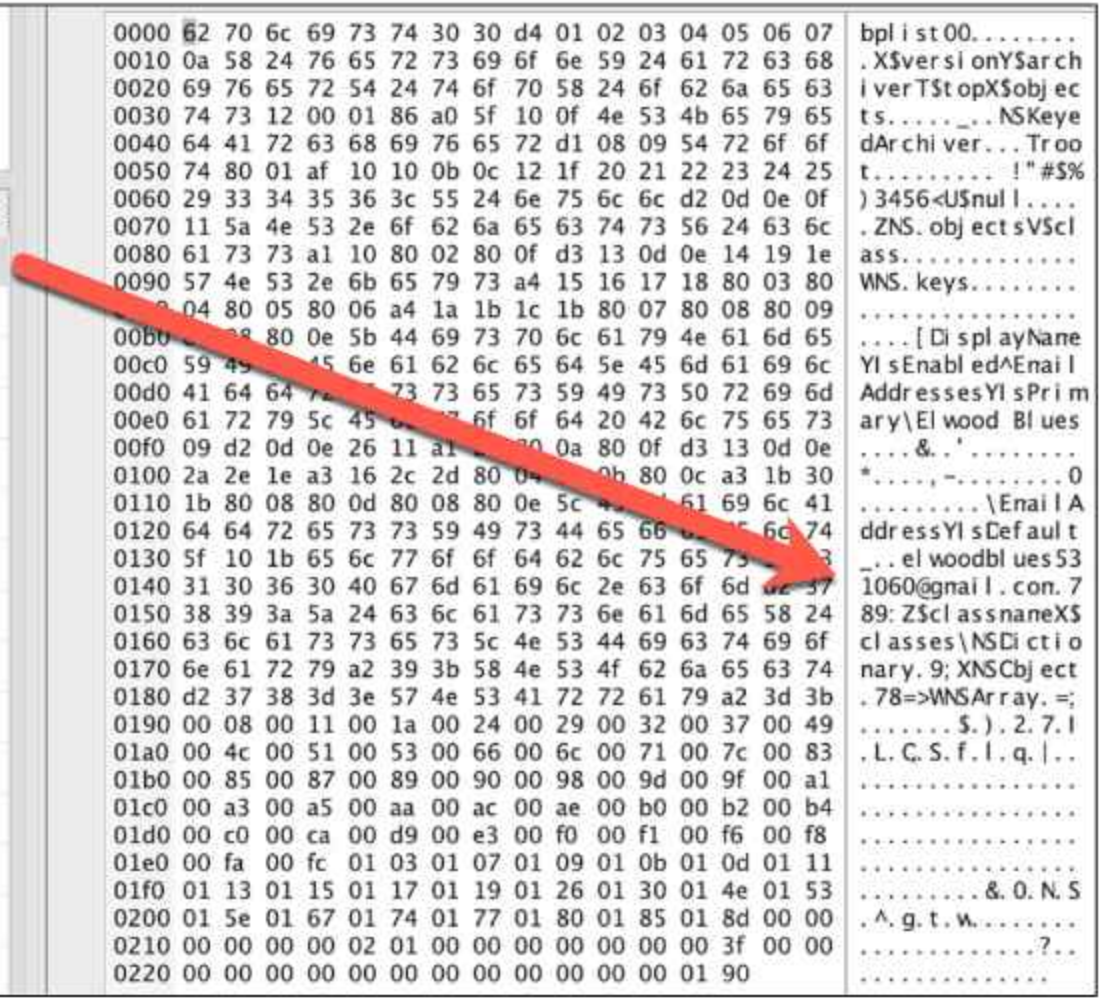
© 2022 Sarah Edwards



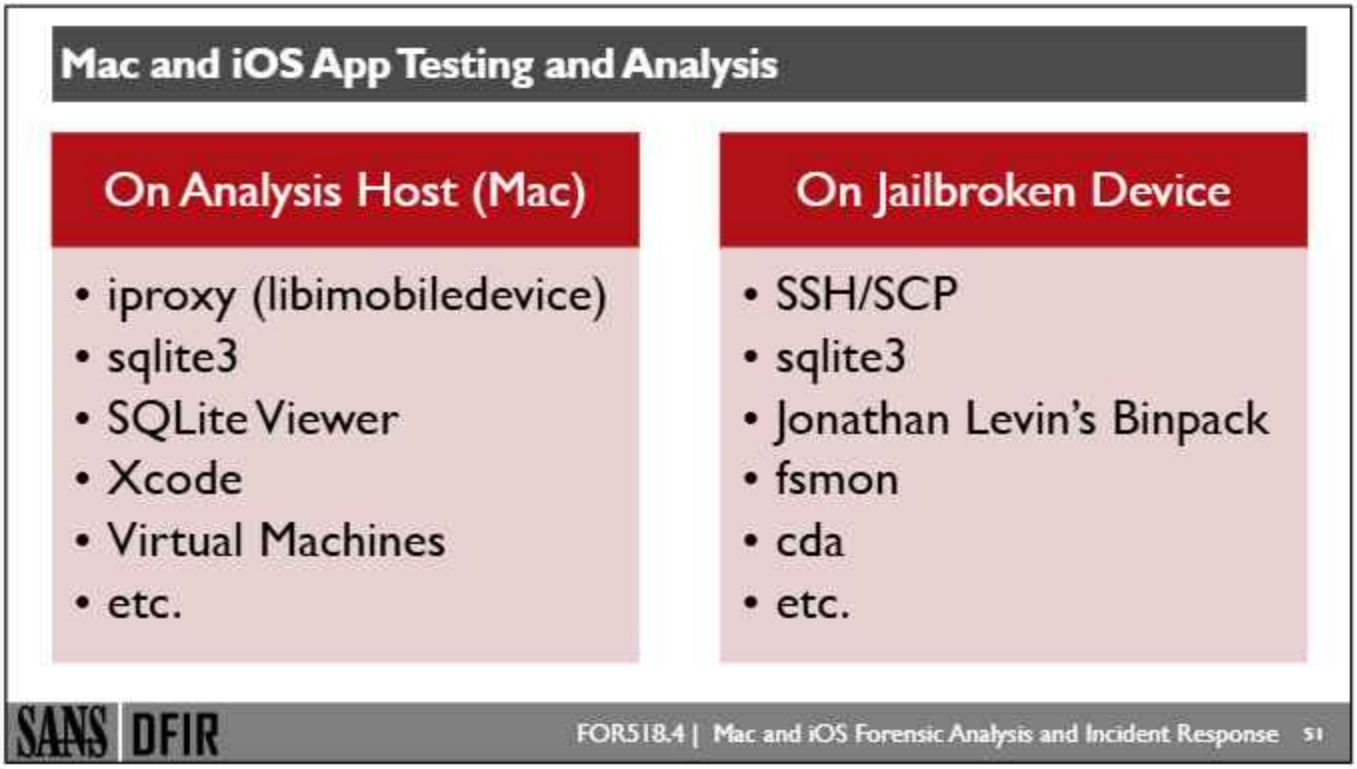
The embedded binary plist ZVALUE contains items like the email server, username, and port information. The example above shows a binary plist file that contains the "EmailAliases" for the IMAP account `elwoodblues531060@gmail.com`.

© 2022 Sarah Edwards

ZKEY	ZVALUE
EmailAliases	*BLOB*
SendingAccountIdentifier	*BLOB*
SSLEnabled	*BLOB*
AuthenticationScheme	*BLOB*
Hostname	*BLOB*
AllowsInsecureAuthentication	*BLOB*
PortNumber	*BLOB*
SSLIsDirect	*BLOB*
altDSID	*BLOB*
cloudDocsMigrationComplete	*BLOB*
accountClass	*BLOB*
personID	*BLOB*
regionInfo	*BLOB*
familyEligible	*BLOB*
notesMigrated	*BLOB*



© 2022 Sarah Edwards



Third-party (or native!) app analysis can be done in a variety of ways, using both command-line tools as well as GUI-based tools.

On the host analysis system (for this example, we will use a Mac) we can install the `libimobiledevice` (https://libimobiledevice.org/) suite of utilities. The easiest way to install these tools is to install Homebrew first (https://brew.sh/), then use the command "`brew install libimobiledevice`". This will install lots of tools, including `iproxy` and `ifuse`. The tool `iproxy` can be used to connect via USB to an analysis device (jailbroken w/SSH installed). Then, SSH can be used to connect to the device for analysis without the need to connect over Wi-Fi.

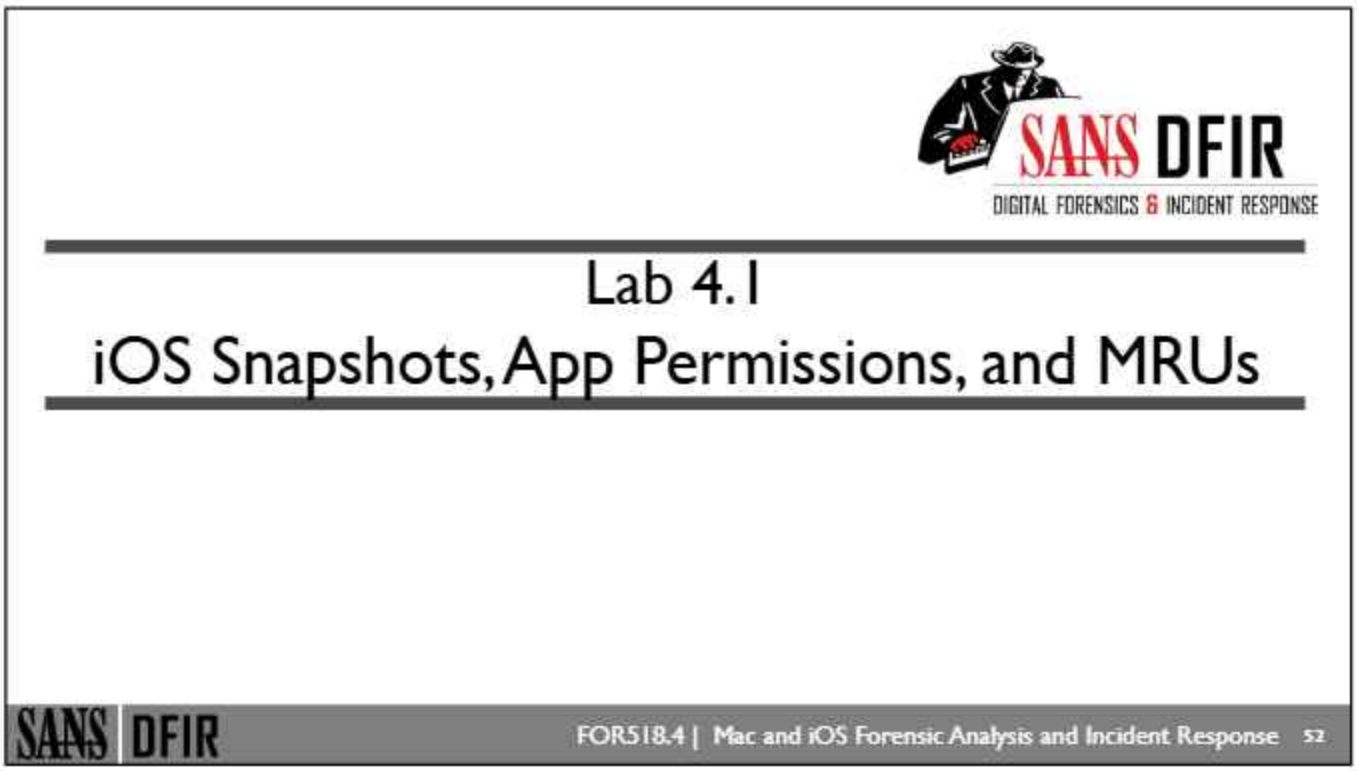
```
iproxy 4242 22
ssh root@127.0.0.1 -p 4242
```

Other utilities on the host will include viewing applications like SQLite (GUI or CLI), and Xcode for plist files.

On the iDevice itself, you will want to use a jailbroken device to get full root access to the system. You never know where files are going to be stored! In the test device, you will want to at least install SSH server (some jailbreaks come with it) to get easy access to the device. Once you install SSH, make sure you change the default password from "`alpine`" as soon as possible, to protect from others getting on your test device.

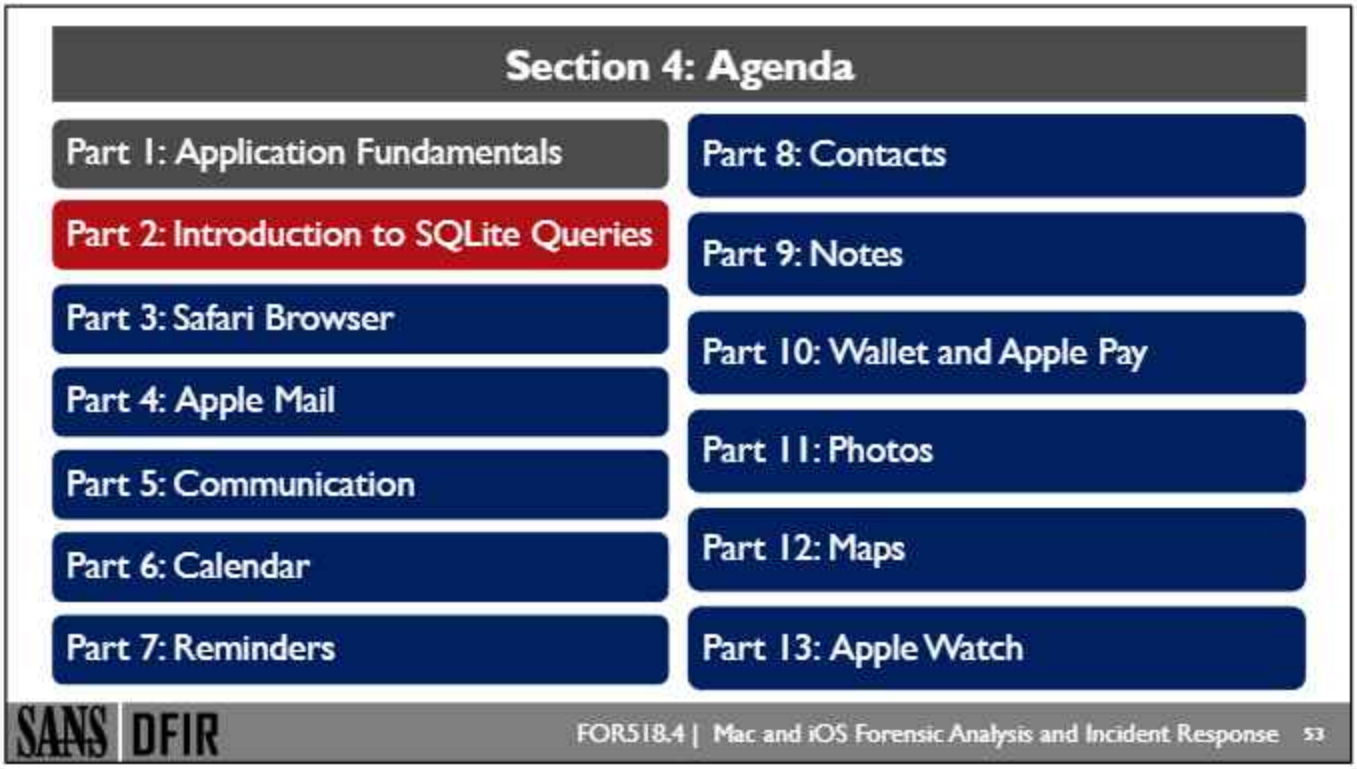
Other utilities include basic UNIX commands contained in Jonathan Levin's binpack (newosxbook.com), fsmon for file system monitoring (https://github.com/nowsecure/fsmon), and CDA (https://github.com/ay-kay/cda) to find where app data is stored.

Reference:
https://www.mac4n6.com/blog/2018/11/25/do-it-live-dynamic-ios-forensic-testing



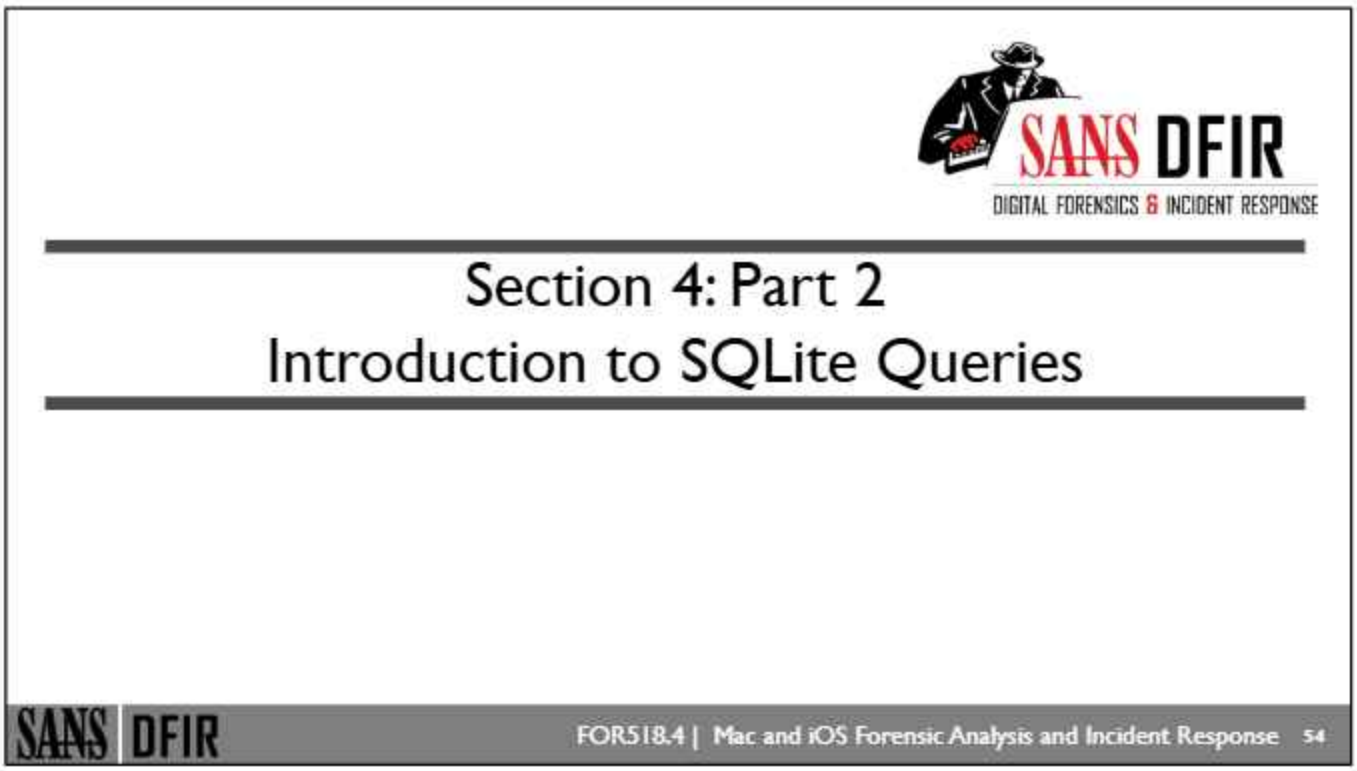
This page intentionally left blank.

© 2022 Sarah Edwards



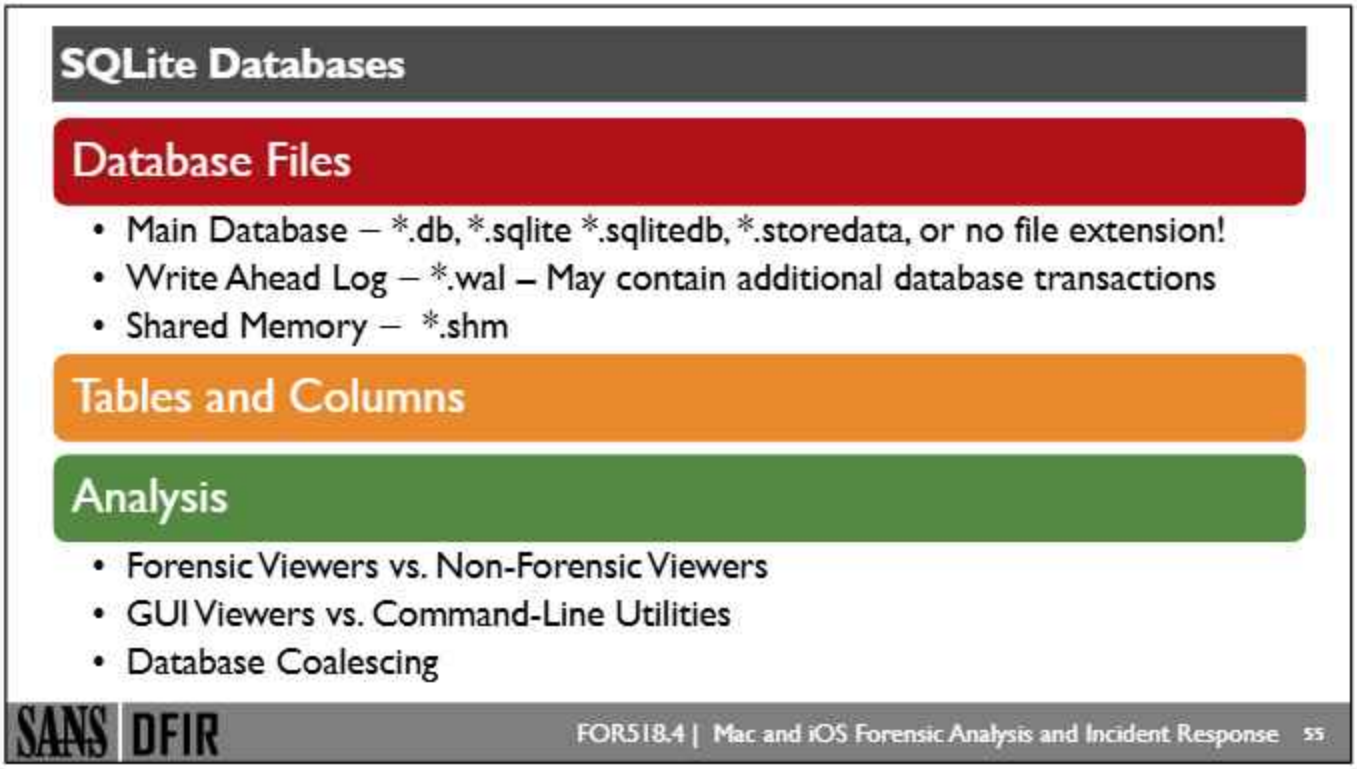
This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards



SQLite databases are made up of multiple files. The main database may have a variety of file extensions or none at all. Looking for that "`SQLite format 3`" header can help you determine if a file is a SQLite database. The Write Ahead Log or WAL file is important as it may contain additional database transactions not yet coalesced into the main database file. The Shared Memory file (`*.shm`) likely does not contain any database transaction but is used to help facilitate transactions into the WAL file.

As with other databases, SQLite databases can be comprised from multiple tables, each table containing different data in columns. Some databases may have one table, others may have hundreds. Most database schemas are quite different from each other. The data in the columns will be a variety of different datatypes.

There are many types of SQLite viewers out there to use for analysis. Some are GUI-based, others can be run from the command line. Most are not forensic-based viewers and may coalesce the transactions in the WAL file into the main database, like DB Browser for SQLite. This is fine as long as the analyst is aware that it's happening or not. Forensic-based utilities may allow the analyst to choose to coalesce or not. Forensic-based browsers may also attempt to recover database entries.

References:
https://www.sqlite.org/docs.html
https://www.sqlitetutorial.net/

© 2022 Sarah Edwards

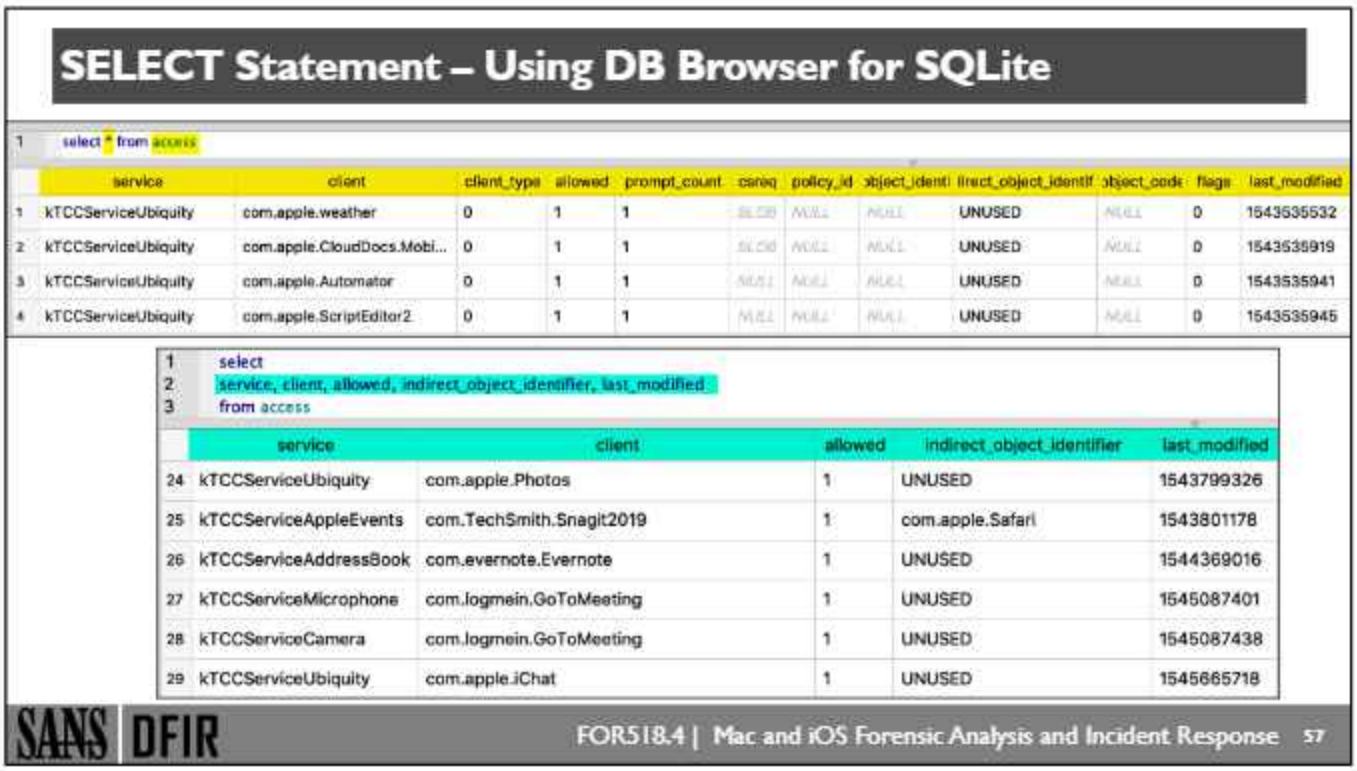
Example Database – TCC.db

Table: access

	service	client	client_type	allowed	prompt_count	csreq	policy_id	indirect_object_identifier_type	indirect_object_identifier	indirect_object_code_identity	flags	last_modified
	Filter	Filter	Filter	Fi...	Filter	...	Fi...	Filter	Filter	Filter	Filter	Filter
1	kTCCServiceUbiquity	com.apple.weather	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555532
2	kTCCServiceUbiquity	com.apple.CloudDocs.MobileDocumentsFileProvider	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555616
3	kTCCServiceUbiquity	com.apple.Automator	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555541
4	kTCCServiceUbiquity	com.apple.ScriptEditor2	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555548
5	kTCCServiceUbiquity	com.apple.iWork.Numbers	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555440
6	kTCCServiceUbiquity	com.apple.Preview	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555548
7	kTCCServiceUbiquity	com.apple.iWork.Keynote	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555548
8	kTCCServiceUbiquity	com.apple.Backlot	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555548
9	kTCCServiceUbiquity	com.apple.iWork.Pages	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555548
10	kTCCServiceUbiquity	com.apple.mail	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555553
11	kTCCServiceUbiquity	com.apple.QuickTimePlayerX	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555535
12	kTCCServiceUbiquity	com.apple.TextEdit	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545555538
13	kTCCServiceLivespaces	com.apple.systempreferences	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545557781
14	kTCCServiceLivespaces	com.apple.Safari	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545558173
15	kTCCServiceUbiquity	com.getdropbox.dropbox	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545558575
16	kTCCServiceAppleEvents	com.egnyte.desktopsyncinstaller.Egnyte-Desktop-Sync-Installer	0	1	1	BLOB	NULL	0	com.apple.systemevents	BLOB	NULL	1545559960
17	kTCCServiceAppleEvents	com.egnyte.desktopsyncinstaller.Egnyte-Desktop-Sync-Installer	0	1	1	BLOB	NULL	0	com.apple.finder	BLOB	NULL	1545559997
18	kTCCServiceLivespaces	com.apple.clocks	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545545772
19	kTCCServiceCamera	com.objective-see.dhs	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	NULL	1545548814
20	kTCCServiceAppleEvents	/usr/bin/osascript	1	1	1	BLOB	NULL	0	com.apple.systemevents	BLOB	NULL	1545545508
21	kTCCServiceUbiquity	com.apple.reminders	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545997814
22	kTCCServiceLivespaces	com.apple.Notes	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545981871
23	kTCCServiceUbiquity	com.apple.Safari	0	1	1	BLOB	NULL	NULL	UNUSED	NULL	0	1545998245

The database above is an example that will be used for most of the following query examples. This `TCC.db` database is a good starter database to use because it is relatively simple. We will be looking at one table, '`access`' to review the permissions for various applications on a macOS system.

© 2022 Sarah Edwards



The `SELECT` SQLite statement is the basis of most SQL queries that will extract data from a database. `SELECT` is used to "select" different items from tables and columns.

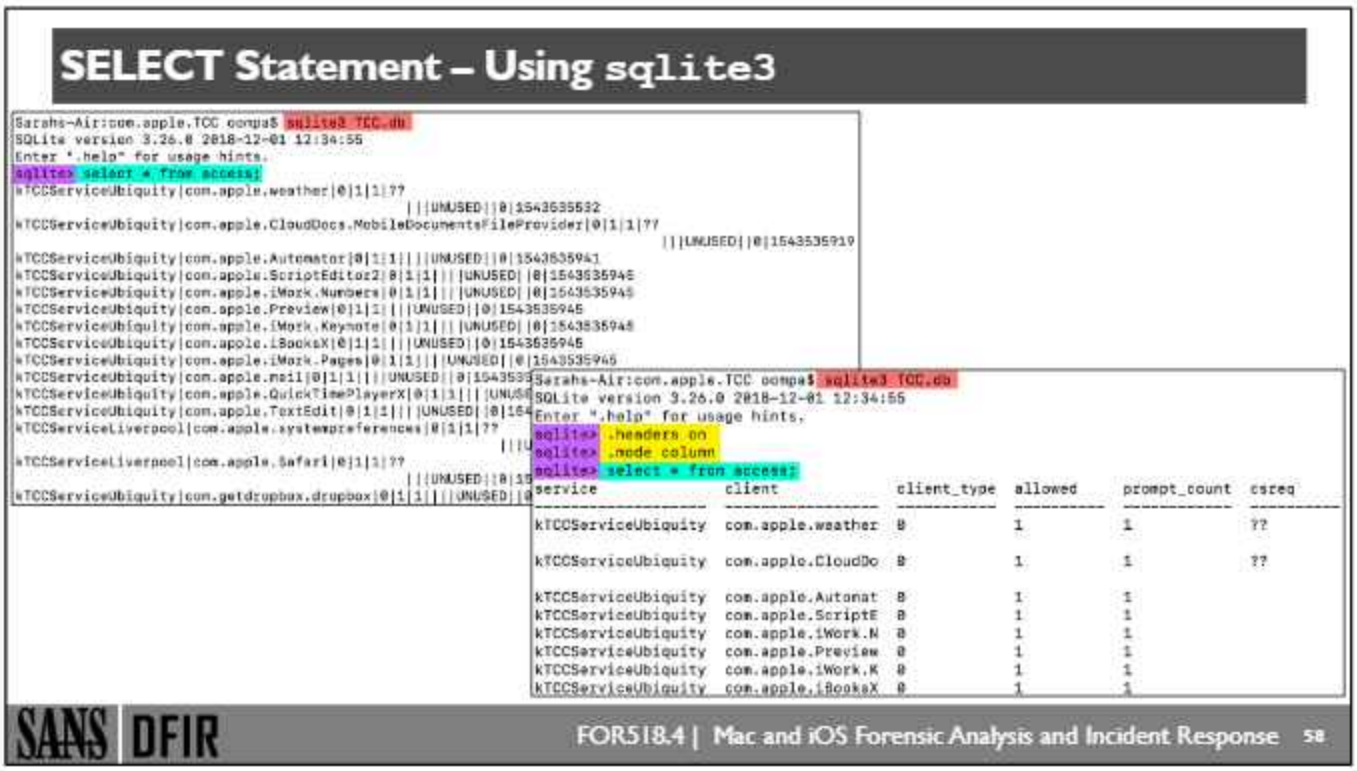
In the top example, a wildcard (`*`) is used to "select" everything from the "`access`" table. This is the broadest example, but also useful to filter down columns to what is of investigative importance while in a query building window of DB Browser for SQLite.

The second example shows data being extracted from only certain columns; `service`, `client`, `allowed`, `indirect_object_identifier`, and `last_modified`.

It is worth mentioning that once queries start becoming more complicated with table JOINs, you will want to preface the column name with a specific table to differentiate columns that may have the same name across tables (i.e., `access.service`, `access.client`, `access.allowed` and so on).

Reference:
https://www.sqlite.org/docs.html

© 2022 Sarah Edwards



If you want to use `sqlite3` on the command line to do your analysis, that is another option. Performing queries on the command line can be done in a couple of ways.

The examples show a query being performed in the SQLite shell. This is an interactive shell specifically for SQLite databases. The default output in the top example is not exactly the easiest to visually parse. Using a few SQLite shell commands in the second example we can make the output more appealing for analysis.

- `.headers on` – This puts the column names at the top of the output.
- `.mode column` – Puts the data in columns. The column width can be configured; this example shows a default width. The data in the columns may be cut off.

Reference:
https://www.sqlite.org/docs.html

© 2022 Sarah Edwards

SELECT Statement – Using `sqlite3`, CLI One-Liner

```
Sarahs-Air:com.apple.TCC oompa$ sqlite3 TCC.db 'select * from access;'
kTCCServiceUbiquity|com.apple.weather|0|1|1|??
                                            |||UNUSED||0|1543535532
kTCCServiceUbiquity|com.apple.CloudDocs.MobileDocumentsFileProvider|0|1|1|??
                                                                        |||UNUSED||0|1543535919
kTCCServiceUbiquity|com.apple.Automator|0|1|1||||UNUSED||0|1543535941
kTCCServiceUbiquity|com.apple.ScriptEditor2|0|1|1||||UNUSED||0|1543535945
kTCCServiceUbiquity|com.apple.iWork.Numbers|0|1|1||||UNUSED||0|1543535945
```

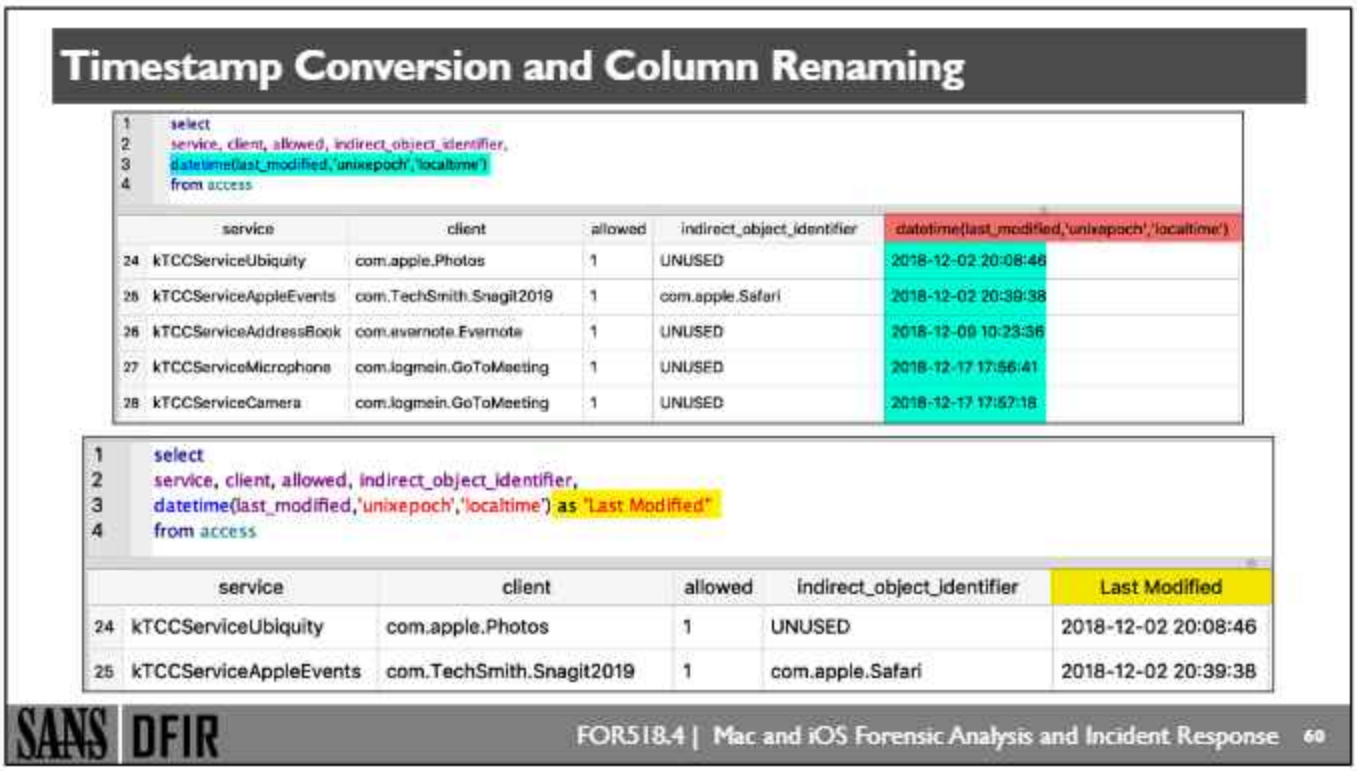
```
Sarahs-Air:com.apple.TCC oompa$ sqlite3 -header -column TCC.db 'select * from access;'
service              client              client_type  allowed     prompt_count  csreq
-------------------  ------------------  -----------  ----------  ------------  -------
kTCCServiceUbiquity  com.apple.weather   0            1           1             ??

kTCCServiceUbiquity  com.apple.CloudDo   0            1           1             ??

kTCCServiceUbiquity  com.apple.Automat   0            1           1
kTCCServiceUbiquity  com.apple.ScriptE   0            1           1
kTCCServiceUbiquity  com.apple.iWork.N   0            1           1
kTCCServiceUbiquity  com.apple.Preview   0            1           1
```


The `sqlite3` utilities can also be used from the command line in a one-liner script (or in a more complex bash script!). The query is provided at the end of the one-liner in quotes.

Like the previous example, the output is not pretty. We can use `-header` and `-column` to fix this.

Reference:
https://www.sqlite.org/docs.html

© 2022 Sarah Edwards



Adding to our initial `SELECT` query, we can manipulate some of the data to make it easier to read. One data type that you will often want to convert is epoch timestamps. Our example stores a Last Modified timestamp in Unix epoch, or number of seconds from 1/1/1970 00:00:00 UTC.

SQLite can convert this by using a `DATETIME` function. The example above is converting from `'unixepoch'` to `'localtime'`. The `'localtime'` modifier will show you the timestamp in local system time. If you wanted to keep it to UTC, remove that modifier, as it will be in UTC by default.

Once converted, the column name is long and unwieldly; this can be changed by using `AS`. The example at the bottom is changing the column name to "`Last Modified`". This can be useful to put more context to otherwise strange column names. Some database designers are better at naming columns/tables than others.

References:
https://www.sqlite.org/docs.html
https://www.sqlite.org/lang_datefunc.html

© 2022 Sarah Edwards

DISTINCT Keyword and Commenting

```
1  select
2  distinct service
3  --, client, allowed, indirect_object_identifier, datetime(last_modified,'unixepoch','localtime') as "Last Modified"
4  from access
```

	service
1	kTCCServiceAddressBook
2	kTCCServiceAppleEvents
3	kTCCServiceCamera
4	kTCCServiceLiverpool
5	kTCCServiceMicrophone
6	kTCCServiceUbiquity

The `DISTINCT` keyword is useful to find unique values in a column. Using our TCC example, if I wanted to see all the different types of permissions being used in this table, I would use `DISTINCT` on the `service` column. There are six distinct permissions for this example.

`DISTINCT` is here used on a single column. When doing analysis, sometimes you just want to quickly see what the values are but not delete everything in your query. This example also shows me commenting out the rest of the query by using "`--`".

Reference:
https://www.sqlite.org/docs.html

© 2022 Sarah Edwards

CASE Expression

```
1  select
2  service, client,
3  case allowed
4      when 0 then 'Not Allowed'
5      when 1 then 'Allowed'
6  end Allowed,
7  indirect_object_identifier,
8  datetime(last_modified,'unixepoch','localtime') as "Last Modified"
9  from access
```

	service	client	Allowed	indirect_object_identifier	Last Modified
30	kTCCServiceLiverpool	com.apple.Maps	Allowed	UNUSED	2018-12-24 13:05:45
31	kTCCServiceAppleEvents	com.blackbagtech.BlackLight	Allowed	com.apple.finder	2019-01-12 19:02:38
32	kTCCServiceAddressBook	com.microsoft.onenote.mac	Allowed	UNUSED	2019-01-18 12:15:28
33	kTCCServiceAddressBook	com.microsoft.Word	Allowed	UNUSED	2019-01-18 12:16:09
34	kTCCServiceAddressBook	com.microsoft.Powerpoint	Allowed	UNUSED	2019-01-18 12:17:15
35	kTCCServiceAppleEvents	com.logitech.presenter	Allowed	com.apple.systempreferences	2019-01-28 10:16:50
36	kTCCServiceAppleEvents	com.logitech.presenter	Allowed	com.apple.Safari	2019-01-28 10:21:03
37	kTCCServiceUbiquity	com.apple.Grab	Allowed	UNUSED	2019-01-31 16:38:24
38	kTCCServiceAddressBook	com.microsoft.Excel	Not Allowed	UNUSED	2019-02-06 20:07:05
39	kTCCServiceAppleEvents	com.logitech.presenter	Allowed	com.google.Chrome	2019-02-20 09:09:02

SANS DFIR FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 62

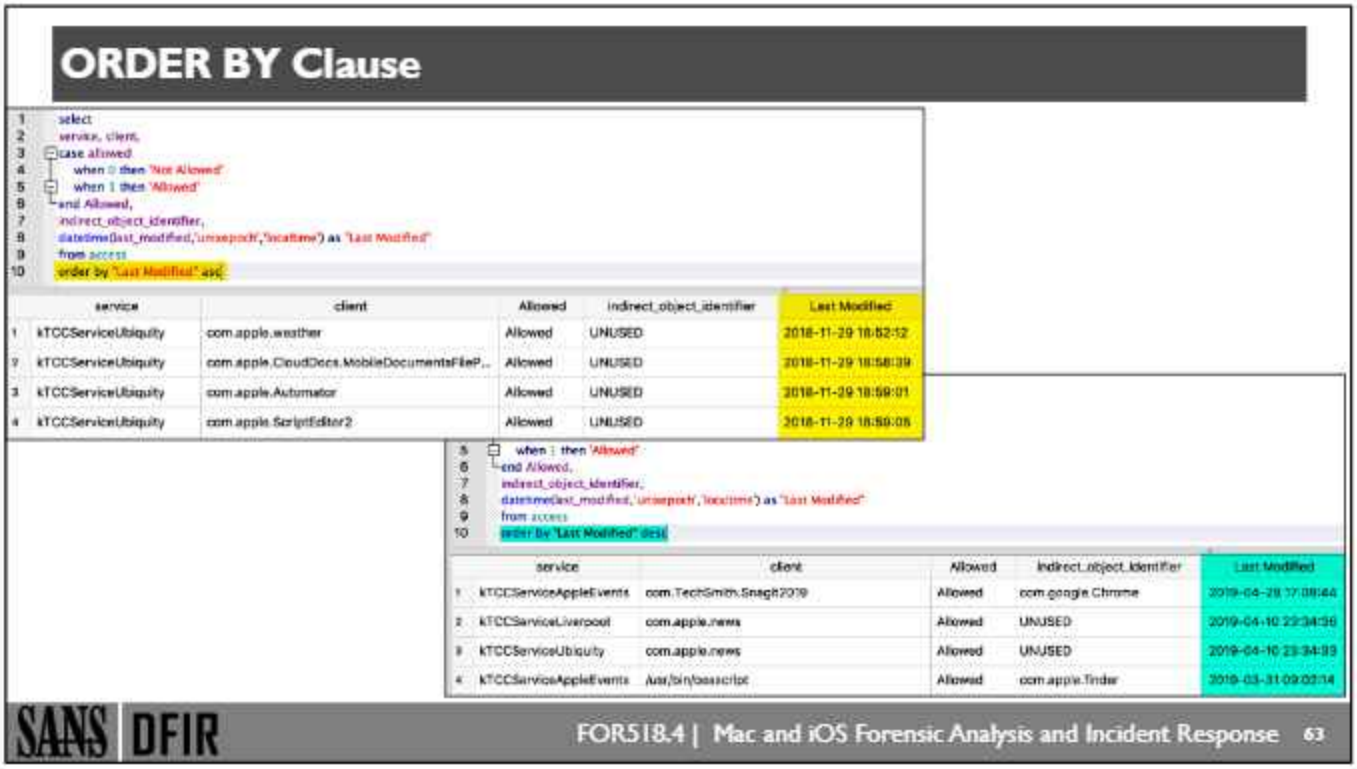
When you have certain values for different types of entries it can be useful to rename them or perform a certain action on just that value. This is where the `CASE` expression is handy.

The example above is changing a value of "`0`" in the allowed column to "`Not Allowed`" and a value of "`1`" to "`Allowed`". This can help provide context to the extracted data.

Another `CASE` example might be to change the service to something more human friendly. (i.e.: `kTCCServiceUbiquity` to "`iCloud`")

Reference:
https://www.sqlite.org/docs.html

© 2022 Sarah Edwards

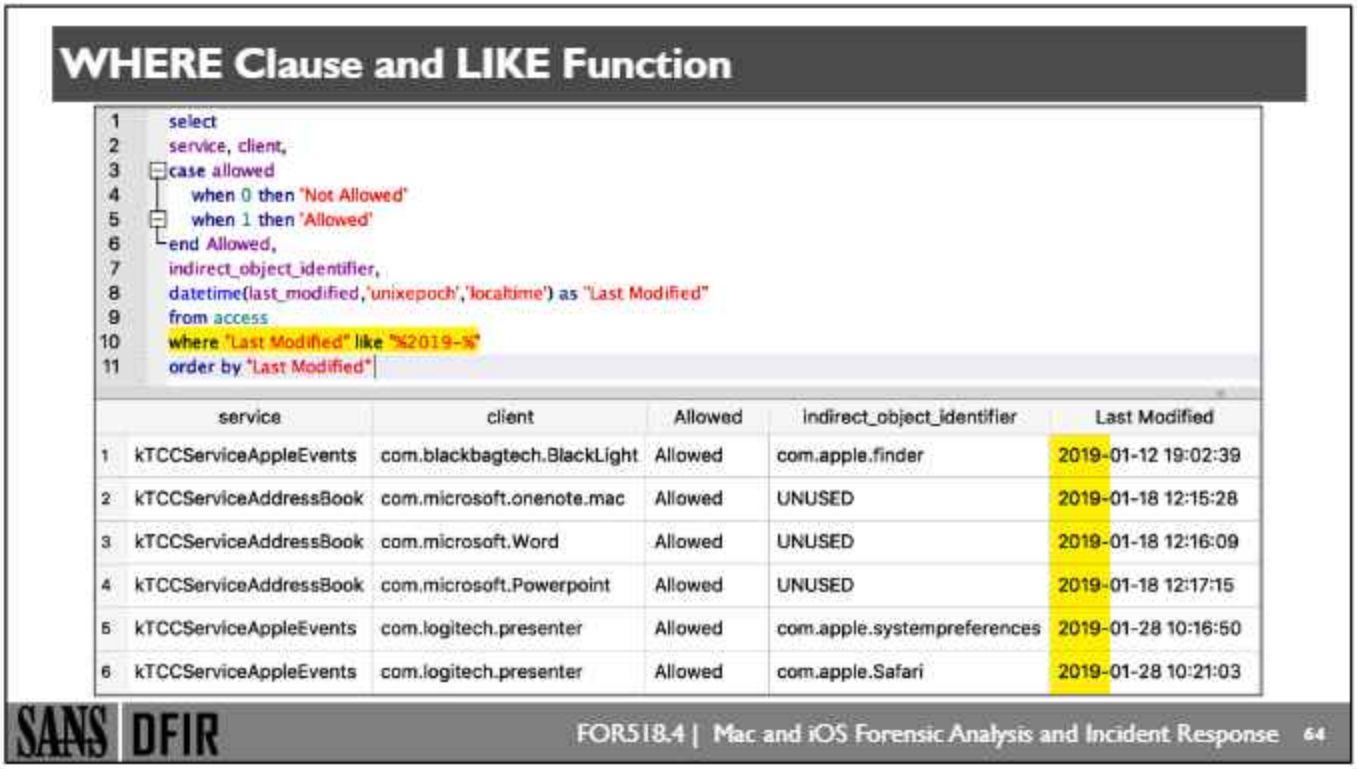


An analysis technique could be to order certain columns, this is commonly done on timestamp columns to determine what records were performed before others.

The `ORDER BY` clause can be used to order temporally in the example above on the "`Last Modified`" column. The two examples show the same statement but different orderings.

- `ASC` – Ascending Order, this is the default if this keyword is not provided.
- `DESC` – Descending Order

Reference:
https://www.sqlite.org/docs.html



Some databases may have thousands of records and filtering is required to find specific records of interest.

Using the `WHERE` clause, we can filter on different pieces of the output. The example shows a filter for "`Last Modified`" times that start with the string "`2019-`". The '`%`' wildcards are used as part of the string matching, zero or more characters before or after the specified string.

Reference:
https://www.sqlite.org/docs.html

© 2022 Sarah Edwards

Table JOINs (Using Safari CloudTabs.db)

```
1  select
2  cloud_tab_devices.device_name,
3  cloud_tabs.title, cloud_tabs.url,
4  datetime(cloud_tab_devices.last_modified+978307200,'unixepoch','localtime') as last_modified
5  from cloud_tabs
6  left join cloud_tab_devices where cloud_tabs.device_uuid = cloud_tab_devices.device_uuid
```

	device_name	title	url	last_modified
1	Sarah's MacBook Air	Twitter / Notifications	https://twitter.com/i/notifications	2019-01-01 13:45:50
2	Sarah's MacBook Air	About Secure Boot - Apple Support	https://support.apple.com/en-us/HT208330	2019-01-01 13:45:50
3	Sarah's MacBook Air	Twitter	https://twitter.com/	2019-01-01 13:45:50
4	Sarah's MacBook Air	Objective-See's Blog	https://objective-see.com/blog/blog_0x3C.html	2019-01-01 13:45:50
5	Sarah's MacBook Air	Jailbreak - The iPhone Wiki	https://www.theiphonewiki.com/wiki/Jailbreak	2019-01-01 13:45:50
6	miPhoneX	Koala - Wikipedia	https://en.m.wikipedia.org/wiki/Koala	2019-01-01 13:47:51
7	miPhoneX	Mac & iOS Forensic Analysis & Incident Resp...	https://www.sans.org/course/mac-and-ios-fo...	2019-01-01 13:47:51
8	miPhoneX	london - Google Search	https://www.google.com/search?q=london&ie...	2019-01-01 13:47:51
9	miPhoneX	Apple Park Visitor Center - Apple	https://www.apple.com/retail/appleparkvisitor...	2019-01-01 13:47:51

SANS DFIR FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 65

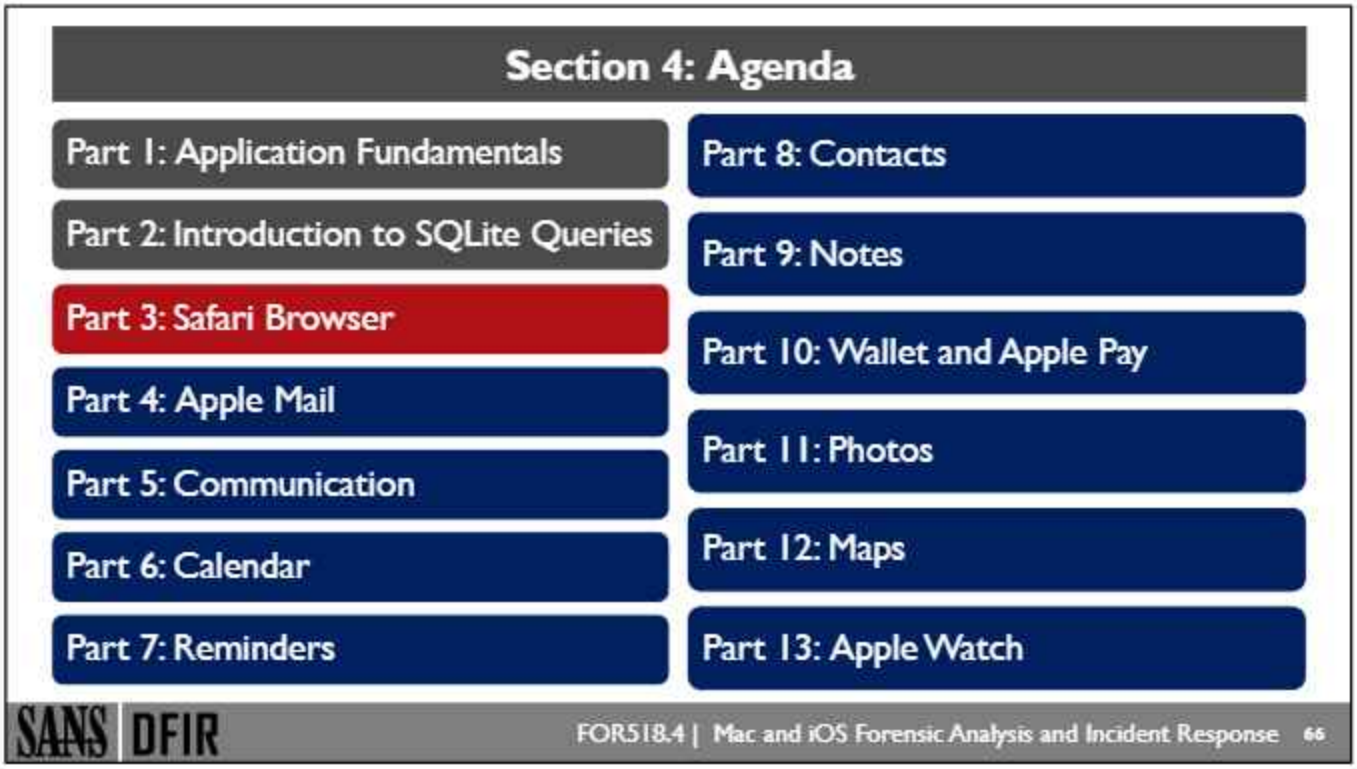
Table `JOINs` are used to combine data between two tables. These joins are performed on a common piece of data in each table.

The example above shows the `CloudTabs.db` Safari database. The table `cloud_tabs` contains the `title` and `url` or synced Safari tabs, while the table `cloud_tab_devices` contains the `device_name`, and `last_modified` timestamp.

The table join is done with a `device_uuid` (not shown) column in each table.

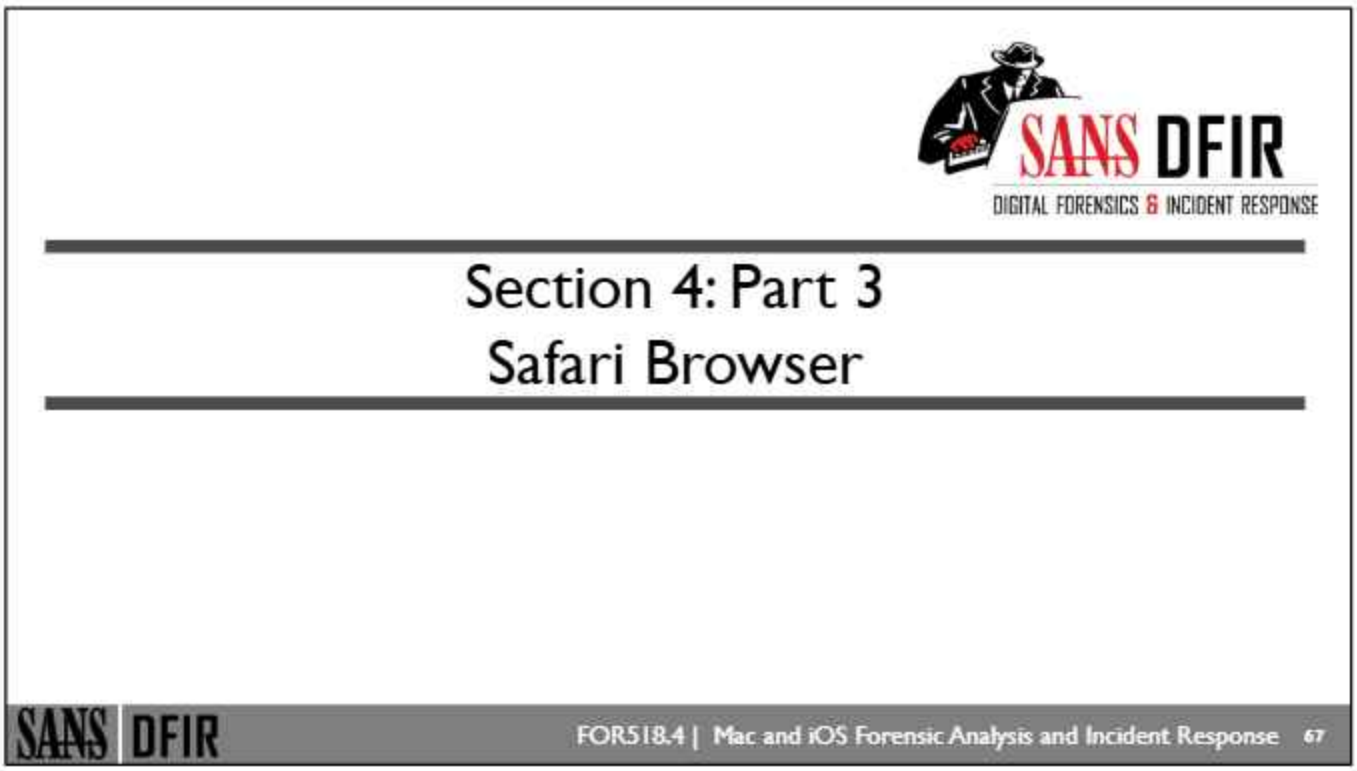
Reference:
https://www.sqlite.org/docs.html

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.

Safari Browser - com.apple.Safari and com.apple.mobilesafari

macOS

- ~/Library/Safari/
- ~/Library/Containers/com.apple.safari/

iOS

- /private/var/mobile/Library/Safari/
- /private/var/mobile/Containers/Data/Application/<GUID>/

Data Types

- History
- Cache(s)
- Session Info
- Tab Snapshots & Thumbnails
- Downloads
- Cookies
- More!

macOS:

- `~/Library /Safari/`
- `~/Library/Containers/com.apple.Safari/`

iOS FFS:

- `/private/var/mobile/Containers/Data/Application/<GUID>`
- `/private/var/mobile/Library/Safari/`

iOS Backup:

- `/mobile/Library/Safari/`

© 2022 Sarah Edwards

Safari History – History.db [1]

macOS & iOS

Synced iCloud History

History Retention:

- macOS: 1 year (default) … or 1 month, 2 weeks, 1 week, 1 day, manually
- iOS: ~1 month

macOS:

- `~/Library/Safari/History.db`

iOS FFS:

- `/private/var/mobile/Library/Safari/History.db`

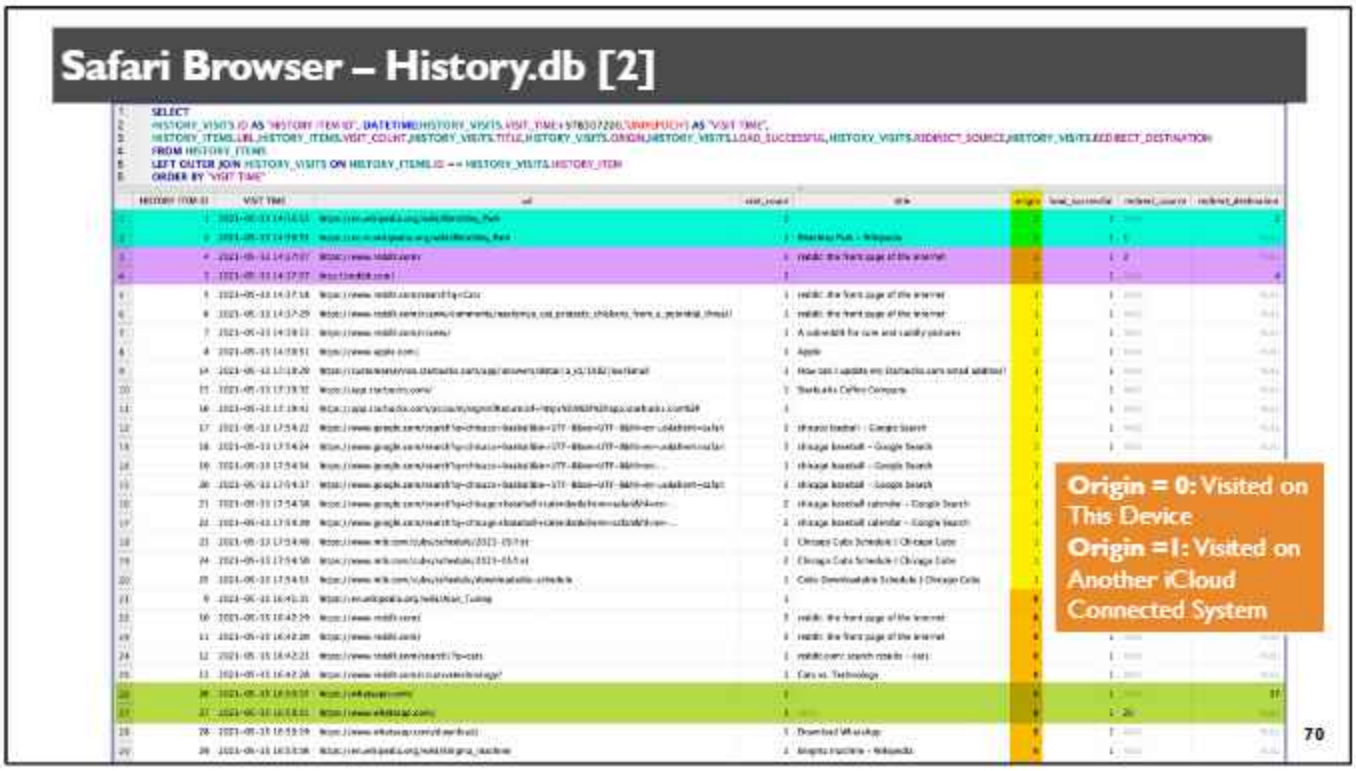
iOS Backup (Encrypted):

- `/mobile/Library/Safari/History.db`

The `History.db` contains the users internet browsing history. If the user has chosen to sync their Safari history using iCloud, the history will be synced to all devices using that iCloud login. If the user clears their history on one device.

In iOS, the history is kept for about a month, while on macOS, it is kept for potentially a year unless otherwise configured.

© 2022 Sarah Edwards



macOS:

- `~/Library/Safari/History.db`

iOS FFS:

- `/private/var/mobile/Library/Safari/History.db`

iOS Backup (Encrypted):

- `/mobile/Library/Safari/History.db`

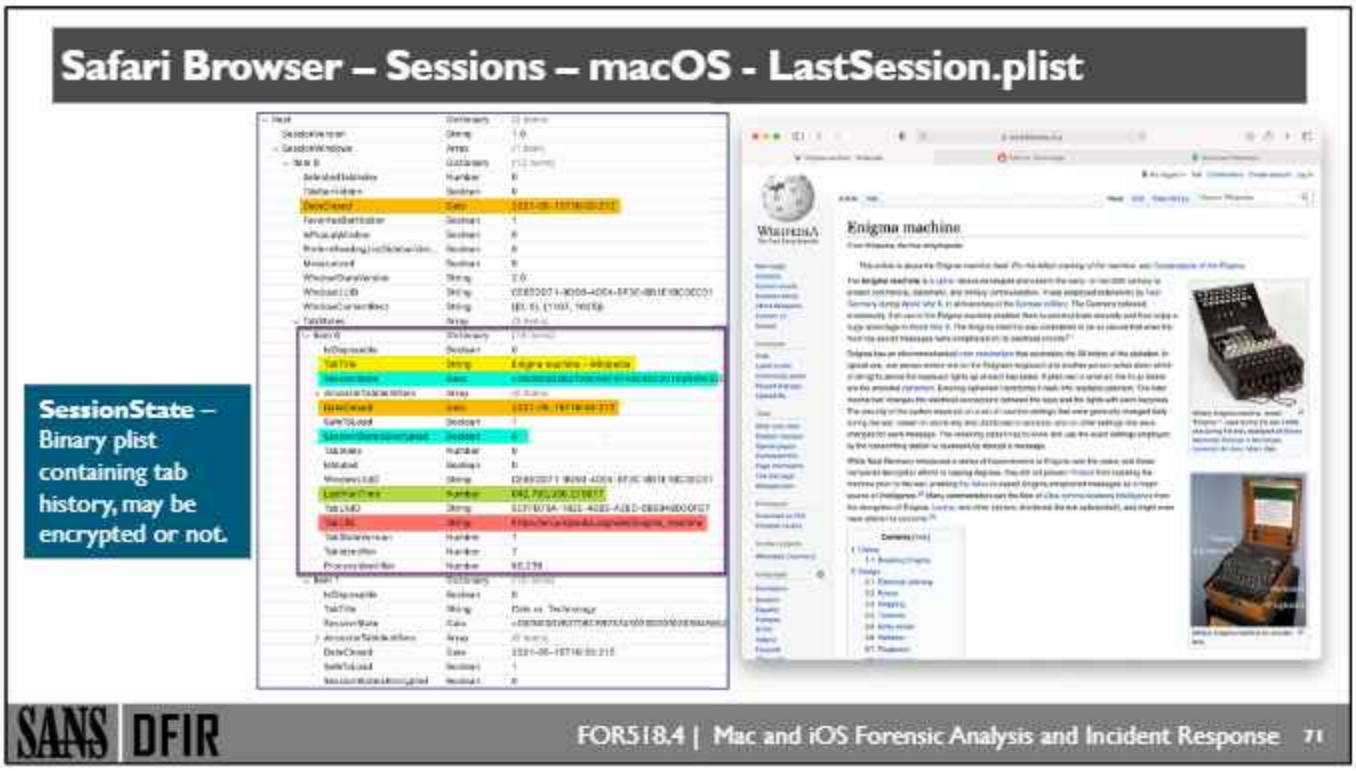
Using the SQLite query above, the following data was extracted from the Safari `History.db` file: The history ID, visit timestamp (Mac Epoch), URL, visit count, webpage title, the origin, if it loaded successfully, and redirection source/destination.

Two tables in this database hold the most relevant data:

- `history_items`: Contains the URLS, domains, and visit count.
- `history_visits`: Contains the Mac Epoch timestamp of when the visits occurred, and the title of the webpage.

If iCloud Safari syncing is enabled, the "`origin`" column will display a "`1`" if that URL was viewed on another device, while a "`0`" means it was viewed from this device.

© 2022 Sarah Edwards

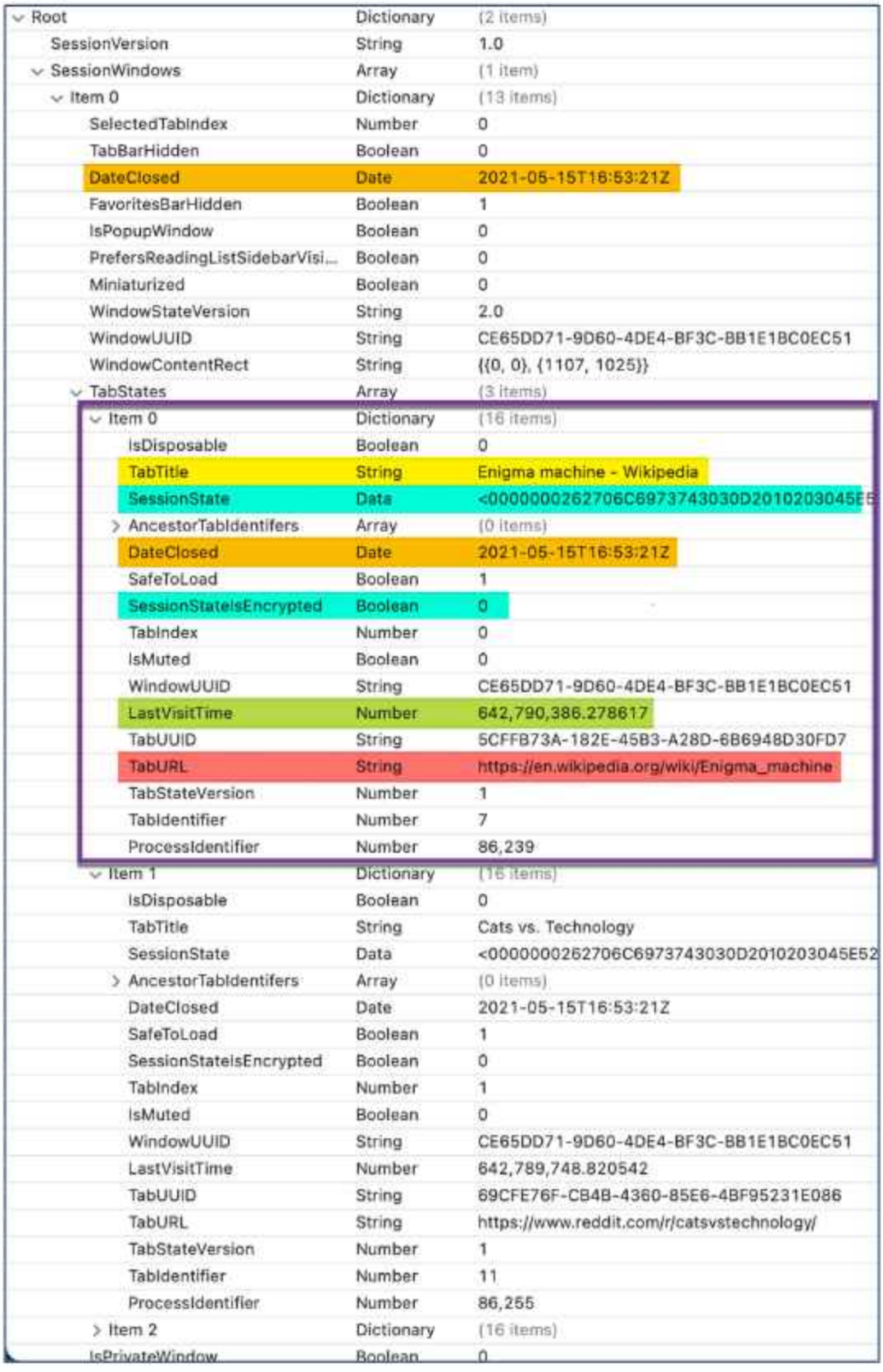


macOS: `~/Library/Safari/LastSession.plist`

The `LastSession.plist` contains items from the last browsing session. If multiple browsing tabs were open, there will be multiple items showing the visited URLs.

Each tab will have a tab identifier, which will not necessarily be in numeric order if tabs were removed. The `TabTitle` and `TabURL` hold the webpage title and URL, respectively. The `SessionState` key may hold more information if the `SessionStateIsEncrypted` key is set to 0. If unencrypted, it will contain an embedded binary plist containing the contents of the tab history.

© 2022 Sarah Edwards



Root	Dictionary	(2 items)
SessionVersion	String	1.0
SessionWindows	Array	(1 item)
Item 0	Dictionary	(13 items)
SelectedTabIndex	Number	0
TabBarHidden	Boolean	0
DateClosed	Date	2021-05-15T16:53:21Z
FavoritesBarHidden	Boolean	1
IsPopupWindow	Boolean	0
PrefersReadingListSidebarVisi...	Boolean	0
Miniaturized	Boolean	0
WindowStateVersion	String	2.0
WindowUUID	String	CE65DD71-9D60-4DE4-BF3C-BB1E1BC0EC51
WindowContentRect	String	{{0, 0}, {1107, 1025}}
TabStates	Array	(3 items)
Item 0	Dictionary	(16 items)
IsDisposable	Boolean	0
TabTitle	String	Enigma machine - Wikipedia
SessionState	Data	<0000000262706C6973743030D2010203045E5
AncestorTabIdentifers	Array	(0 items)
DateClosed	Date	2021-05-15T16:53:21Z
SafeToLoad	Boolean	1
SessionStateIsEncrypted	Boolean	0
TabIndex	Number	0
IsMuted	Boolean	0
WindowUUID	String	CE65DD71-9D60-4DE4-BF3C-BB1E1BC0EC51
LastVisitTime	Number	642,790,386.278617
TabUUID	String	5CFFB73A-182E-45B3-A28D-6B6948D30FD7
TabURL	String	https://en.wikipedia.org/wiki/Enigma_machine
TabStateVersion	Number	1
TabIdentifier	Number	7
ProcessIdentifier	Number	86,239
Item 1	Dictionary	(16 items)
IsDisposable	Boolean	0
TabTitle	String	Cats vs. Technology
SessionState	Data	<0000000262706C6973743030D2010203045E52
AncestorTabIdentifers	Array	(0 items)
DateClosed	Date	2021-05-15T16:53:21Z
SafeToLoad	Boolean	1
SessionStateIsEncrypted	Boolean	0
TabIndex	Number	1
IsMuted	Boolean	0
WindowUUID	String	CE65DD71-9D60-4DE4-BF3C-BB1E1BC0EC51
LastVisitTime	Number	642,789,748.820542
TabUUID	String	69CFE76F-CB4B-4360-85E6-4BF95231E086
TabURL	String	https://www.reddit.com/r/catsvstechnology/
TabStateVersion	Number	1
TabIdentifier	Number	11
ProcessIdentifier	Number	86,255
Item 2	Dictionary	(16 items)
IsPrivateWindow	Boolean	0

© 2022 Sarah Edwards

Safari Browser – Sessions – iOS – BrowserState.db

- Session data is unencrypted
- Regular and Private Modes
- External Links
- Tab UUID Screenshot in Thumbnails directory

```
SELECT
DATETIME(LAST_VIEWED_TIME+978307200,'UNIXEPOCH','LOCALTIME') AS "LAST_VIEWED_TIME",
TAB_SESSIONS.TAB_UUID,TABS.TITLE,TABS.URL,TABS.ORDER_INDEX,TABS.PRIVATE_BROWSING,TABS.OPENED_FROM_LINK,TAB_SESSIONS.SESSION_DATA
FROM TABS
LEFT JOIN TAB_SESSIONS ON TABS.ID == TAB_SESSIONS.ID
ORDER BY LAST_VIEWED_TIME
```

	LAST_VIEWED_TIME	tab_uuid	title	url	order_index	private_browsing	opened_from_link	session_data
1	2021-05-13 10:50:58	FCD0D583-4360-405D-8427-6B785A57A8B8	Bletchley Park - Wikipedia	https://en.m.wikipedia.org/wiki/Bletchley_Park	0	0	0	
2	2021-05-13 10:59:40	E7275B77-F564-4C80-8EF8-CBF3005EFD98	A subreddit for cute and cuddly pictures	https://www.reddit.com/r/aww/	1	0	0	
3	2021-05-13 13:13:32	653D8962-89C9-479C-9D5F-59B0AE01C3E	Starbucks Coffee Company	https://app.starbucks.com/	4	0	1	
4	2021-05-13 13:51:40	06CC0C85-0923-46AD-8B84-8F6088918000	How can I update my Starbucks.com email address?	https://customerservice.starbucks.com/app/answers/...	3	0	0	
5	2021-05-13 13:51:43	1C49AD1A-8780-4674-8F8E-55C205302148	Apple	https://www.apple.com/	2	0	0	
6	2021-05-13 12:43:23	C64AAC40-1D61-4610-5AA5-81E79A6DA000	puppies - Google Search	https://www.google.com/search?...	0	1	0	
7	2021-05-13 12:43:59	983E8DC8-CAAE-41F7-9288-752E2D80F548	Cubs Downloadable Schedule | Chicago Cubs	https://www.mlb.com/cubs/schedule/downloadable-...	1	0	0	
8	2021-05-13 12:43:59	5B2BF29F-4900-4558-A6EE-39CE9B03B17D	clipsy - Google Search	https://www.google.com/search?...	1	1	0	

iOS FFS: /private/var/mobile/Library/Safari/BrowserState.db
iOS Backup (Encrypted): `/mobile/Library/Safari/BrowserState.db`

The `BrowserState.db` file is a SQLite database that contains relatively the same information `LastSession.plist` on macOS – browser session information.

The screenshot above shows the last viewed time for the tab contents (in Mac Epoch), tab GUID, webpage title, URL and browser visit information. The `order_index` shows that regular and private modes are being used, with those private windows marked in `private_browsing`.

If the link was opened from an external source (like an email), you can see it marked in `opened_from_link`. The session data is unencrypted in the `session_data` BLOB.

© 2022 Sarah Edwards

```
1  SELECT
2  DATETIME(LAST_VIEWED_TIME+978307200,'UNIXEPOCH','LOCALTIME') AS "LAST_VIEWED_TIME",
3  TAB_SESSIONS.TAB_UUID,TABS.TITLE,TABS.URL,TABS.ORDER_INDEX,TABS.PRIVATE_BROWSING,TABS.OPENED_FROM_LINK,TAB_SESSIONS.SESSION_DATA
4  FROM TABS
5  LEFT JOIN TAB_SESSIONS ON TABS.ID == TAB_SESSIONS.ID
6  ORDER BY LAST_VIEWED_TIME
```

	LAST_VIEWED_TIME	tab_uuid	title	url	order_index	private_browsing	opened_from_link	session_data
1	2021-05-13 10:36:56	FCD0D583-4360-405D-8427-68785A57ABBB	Bletchley Park - Wikipedia	https://en.m.wikipedia.org/wiki/Bletchley_Park	0	0	0	
2	2021-05-13 10:39:40	E7275877-F564-4CB0-8EF8-CBF3005EFD98	A subreddit for cute and cuddly pictures	https://www.reddit.com/r/aww/	1	0	0	
3	2021-05-13 13:19:32	653D8962-89C9-4F9C-8D9F-59E89EE65C3E	Starbucks Coffee Company	https://app.starbucks.com/	4	0	1	
4	2021-05-13 13:50:40	D6CC0CB5-0923-46AD-8E8A-8F6D8B91B9D0	How can I update my Starbucks.com email address?	https://customerservice.starbucks.com/app/answers/...	3	0	0	
5	2021-05-13 13:50:43	1C49AD18-87B0-4674-8F6E-55C2053E2148	Apple	https://www.apple.com/	2	0	0	
6	2021-05-15 12:43:23	C64AAC40-3D81-4610-9AA5-8EF78A6DAD60	puppies - Google Search	https://www.google.com/search?...	0	1	0	
7	2021-05-15 12:43:59	983E6DCB-CAAE-41F7-9286-752E2DB0F548	Cubs Downloadable Schedule | Chicago Cubs	https://www.mlb.com/cubs/schedule/downloadable-...	3	0	0	
8	2021-05-15 12:43:59	5828F29F-4909-455B-A6EE-39CE9893857D	clippy - Google Search	https://www.google.com/search?...	1	1	0	

© 2022 Sarah Edwards

Safari Browser – Sessions – iOS – Safari Thumbnails

- KTX Format
- Correlated with UUID in BrowserState.db
  - Includes Regular and Private Mode
- When tab is closed, removed from file system

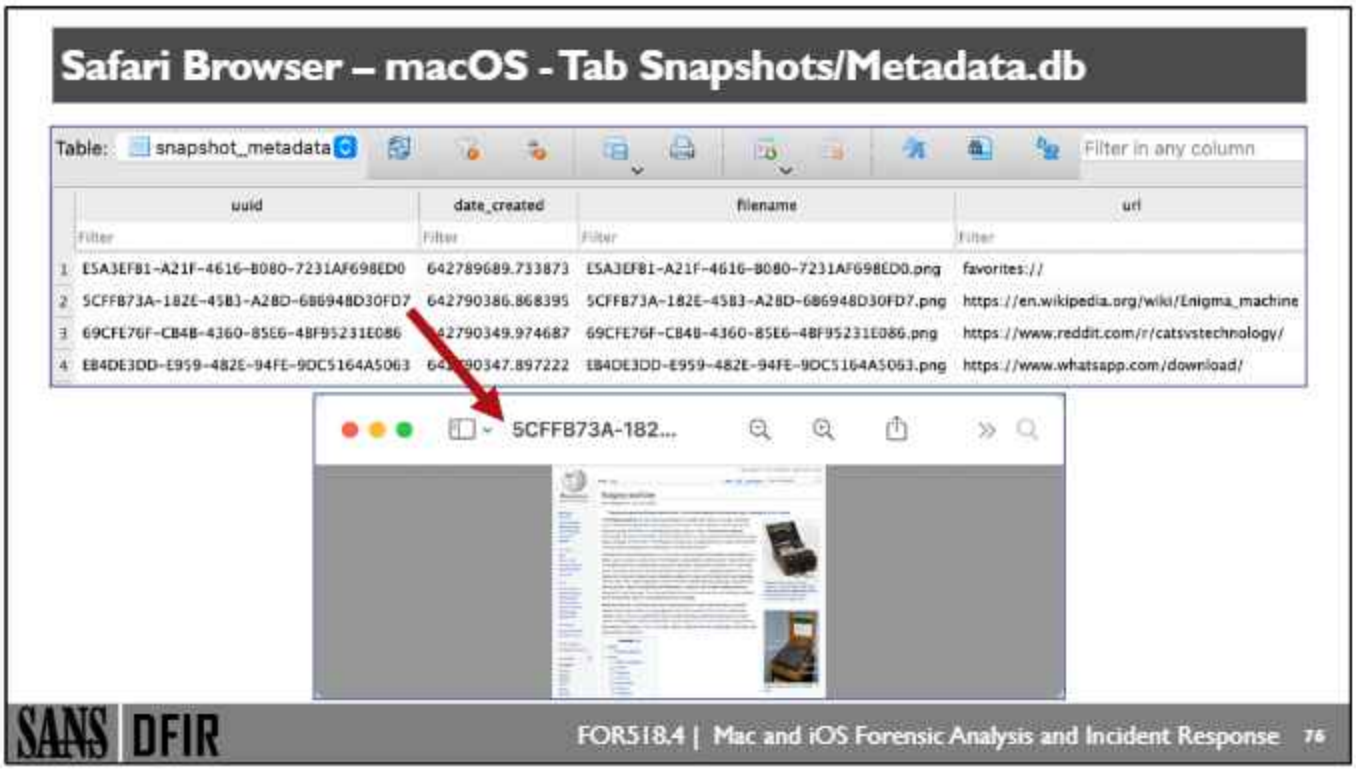
iOS FFS:

- `/private/var/mobile/Containers/Data/Application/<GUID>/Library/Safari/Thumbnails/`

Each GUID in the Thumbnails directory contains a thumbnail of the tab "screenshot", including those in Private Mode. If the user visits another website, the tab "screenshot" will get updated with a new "screenshot". This happens when the Safari application is backgrounded, the device is locked, or the user selects a different tab.

In the example above, this device has eight tabs open, some of which are in "Private Mode". A search for 'clippy' was done in private browsing mode. Once this tab is cleared by the user, the screenshot will no longer exist in this location.

© 2022 Sarah Edwards



macOS: ~/Library/Containers/com.apple.Safari/Data/Library/Caches/com.apple.Safari/TabSnapshots/Metadata.db

Tab screenshots may be found in the Safari cache directory. These screenshots are not exactly high resolution; however, we can determine what website they are a screenshot of by looking in the related `Metadata.db` SQLite database file.

In the example above, each tab screenshot is assigned a UUID, which is also used in the saved filename. Going to the `TabSnapshots` directory, we can open and view these screenshots.

© 2022 Sarah Edwards

Safari Browser – CloudTabs.db

- Synced iCloud Tabs
- Find other iCloud connected devices

```
1  select
2  cloud_tab_devices.device_uuid, cloud_tab_devices.device_name,
3  cloud_tabs.title, cloud_tabs.url,datetime(cloud_tab_devices.last_modified+978307200,'unixepoch','localtime') as last_modified
4  from cloud_tabs
5  left join cloud_tab_devices where cloud_tabs.device_uuid = cloud_tab_devices.device_uuid
```

	device_uuid	device_name	title	url	last_modified
1	119351E9-C886-4763-8818-900F7ECD6977	MBP-M1	Cats vs. Technology	https://www.reddit.com/r/catsvstechnology/	2021-05-15 00:00:00
2	333F0A90-EA0C-4C87-93D8-072DF01D6172	Elwood's iPhone	A subreddit for cute and cuddly pictures	https://www.reddit.com/r/aww/	2021-05-15 12:44:19
3	333F0A90-EA0C-4C87-93D8-072DF01D6172	Elwood's iPhone	Bletchley Park - Wikipedia	https://en.m.wikipedia.org/wiki/Bletchley_Park	2021-05-15 12:44:19
4	333F0A90-EA0C-4C87-93D8-072DF01D6172	Elwood's iPhone	Cubs Downloadable Schedule | Chicago Cubs	https://www.mlb.com/cubs/schedule/downloadable-...	2021-05-15 12:44:19
5	333F0A90-EA0C-4C87-93D8-072DF01D6172	Elwood's iPhone	Apple	https://www.apple.com/	2021-05-15 12:44:19
6	119351E9-C886-4763-8818-900F7ECD6977	MBP-M1	Download WhatsApp	https://www.whatsapp.com/download/	2021-05-15 00:00:00
7	119351E9-C886-4763-8818-900F7ECD6977	MBP-M1	Enigma machine - Wikipedia	https://en.wikipedia.org/wiki/Enigma_machine	2021-05-15 00:00:00

macOS: `~/Library/Safari/CloudTabs.db`
iOS FFS: `/private/var/mobile/Library/Safari/CloudTabs.db`

Devices that are syncing a user's Safari web history can be found in the `CloudTabs.db` database.

The example above shows two devices by hostname: MBP-M1 and Elwood's iPhone. Each device has a few (non-private) tabs open in their respective Safari browsers. The last modification timestamp is a good place to see how stale these tabs are but does not necessarily show their last visit time.

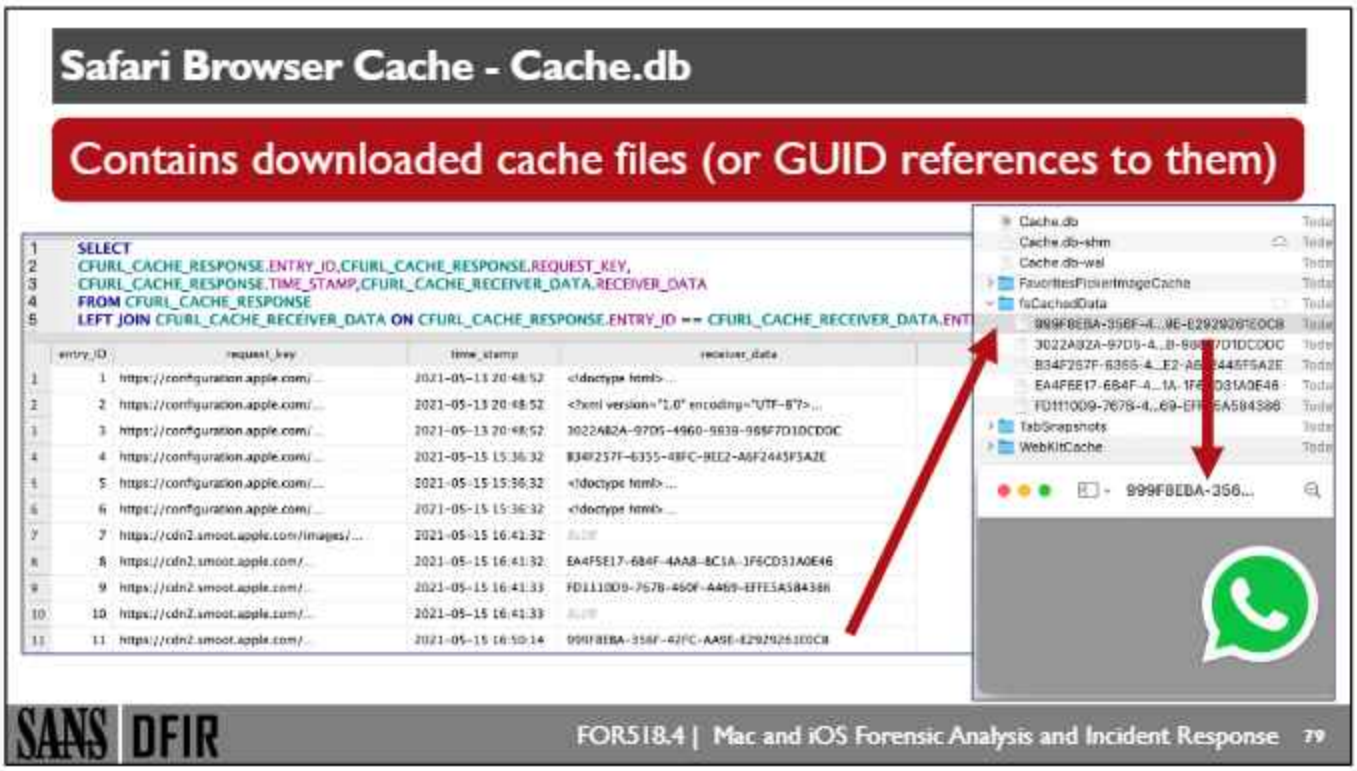
The device UUID can be found in the `/private/var/Library/Safari/SyncedTabsMetadata.plist` on iOS and in the `~/Library/Containers/com.apple.Safari/Data/Library/Preferences/ByHost/com.apple.Safari.<HW UUID>.plist` on macOS.

© 2022 Sarah Edwards

```
1    select
2    cloud_tab_devices.device_uuid, cloud_tab_devices.device_name,
3    cloud_tabs.title, cloud_tabs.url,datetime(cloud_tab_devices.last_modified+978307200,'unixepoch','localtime') as last_modified
4    from cloud_tabs
5    left join cloud_tab_devices where cloud_tabs.device_uuid = cloud_tab_devices.device_uuid
```

	device_uuid	device_name	title	url	last_modified
1	119351E9-C886-4763-B81B-900F7ECD6977	MBP-M1	Cats vs. Technology	https://www.reddit.com/r/catsvstechnology/	2021-05-15 00:00:00
2	333F0A90-EA0C-4C87-93DB-D72DF01D6172	Elwood's iPhone	A subreddit for cute and cuddly pictures	https://www.reddit.com/r/aww/	2021-05-15 12:44:19
3	333F0A90-EA0C-4C87-93DB-D72DF01D6172	Elwood's iPhone	Bletchley Park - Wikipedia	https://en.m.wikipedia.org/wiki/Bletchley_Park	2021-05-15 12:44:19
4	333F0A90-EA0C-4C87-93DB-D72DF01D6172	Elwood's iPhone	Cubs Downloadable Schedule | Chicago Cubs	https://www.mlb.com/cubs/schedule/downloadable-...	2021-05-15 12:44:19
5	333F0A90-EA0C-4C87-93DB-D72DF01D6172	Elwood's iPhone	Apple	https://www.apple.com/	2021-05-15 12:44:19
6	119351E9-C886-4763-B81B-900F7ECD6977	MBP-M1	Download WhatsApp	https://www.whatsapp.com/download/	2021-05-15 00:00:00
7	119351E9-C886-4763-B81B-900F7ECD6977	MBP-M1	Enigma machine - Wikipedia	https://en.wikipedia.org/wiki/Enigma_machine	2021-05-15 00:00:00

© 2022 Sarah Edwards



macOS:
`~/Library/Containers/com.apple.Safari/Data/Library/Caches/com.apple.Safari/`
iOS FFS: `/private/var/mobile/Containers/Data/Application/<GUID>/Library/Caches/`

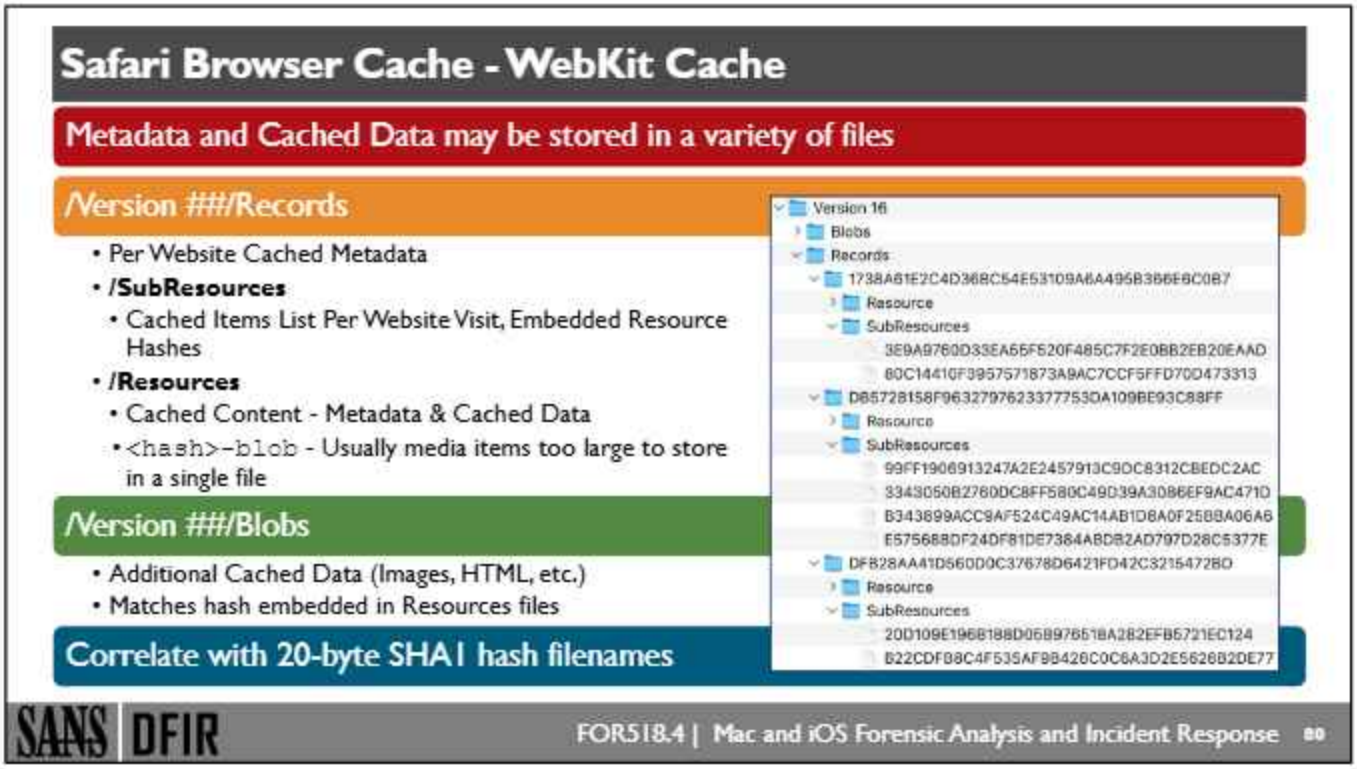
The SQLite database `Cache.db` contains the downloaded cache files. Each file has a corresponding location and download date. On newer macOS and iOS devices not much cached information will be located here. Instead, it can be found in WebKit Cache.

This format for `Cache.db` is used for many other applications, not just Safari. It's worth looking into other applications' `Cache.db` databases to find items of interest.

The SQLite query above extracts and correlates the cache from two tables: One metadata, the other with data. The `cfurl_cache_response` table contains cache file metadata, including the file cached and its corresponding timestamp. The `cfurl_cache_receiver_data` table contains the cached file. The cached file can be matched up with its metadata by using the `entry_ID` number.

Each cached entry has an associated entry_ID, URL (`request_key`), timestamp, and cached data (`receiver_data`). The `receiver_data` column may contain a GUID that can be associated with a file in the `fsCachedData` directory.

© 2022 Sarah Edwards



macOS:
`~/Library/Containers/com.apple.Safari/Data/Library/Caches/com.apple.Safari/WebKitCache/`
iOS FFS:
`/private/var/mobile/Containers/Data/Application/<GUID>/Library/Caches/WebKit/`

The bulk of the cached data is stored in the WebKit Cache in a variety of nested directories and files.

All WebKit cached items can be correlated together with 20-byte filename hashes.

The `Records` directory contains the cached information for each web visit. In each visit there are `SubResources` files that contain a listing of cached items. The actual cashed data is stored in the Resource directory.

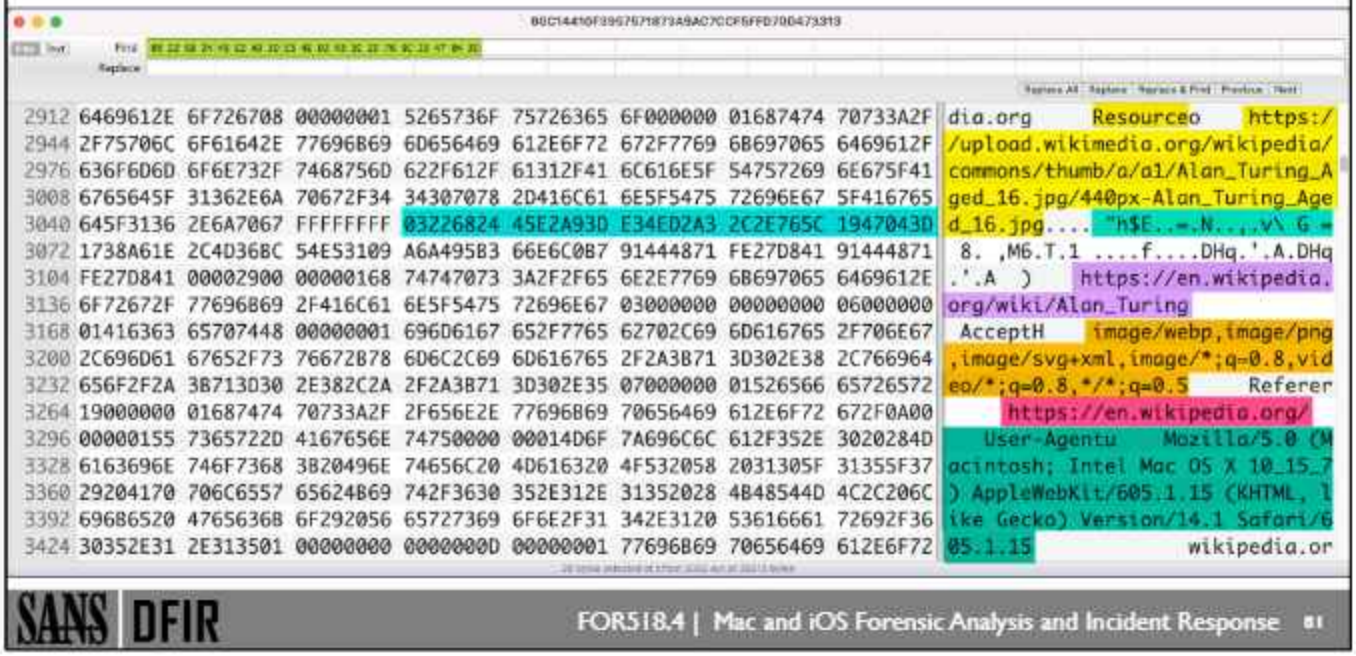
The `Blobs` directory contains additional cached items that match the hashes embedded in the Resource files.

Each of these files contain embedded SHA1 hashes for the filenames of the related content.

Reference:
https://webkit.org/blog/7477/new-web-features-in-safari-10-1/

© 2022 Sarah Edwards

SubResource – Website Visit, List of Cached Items



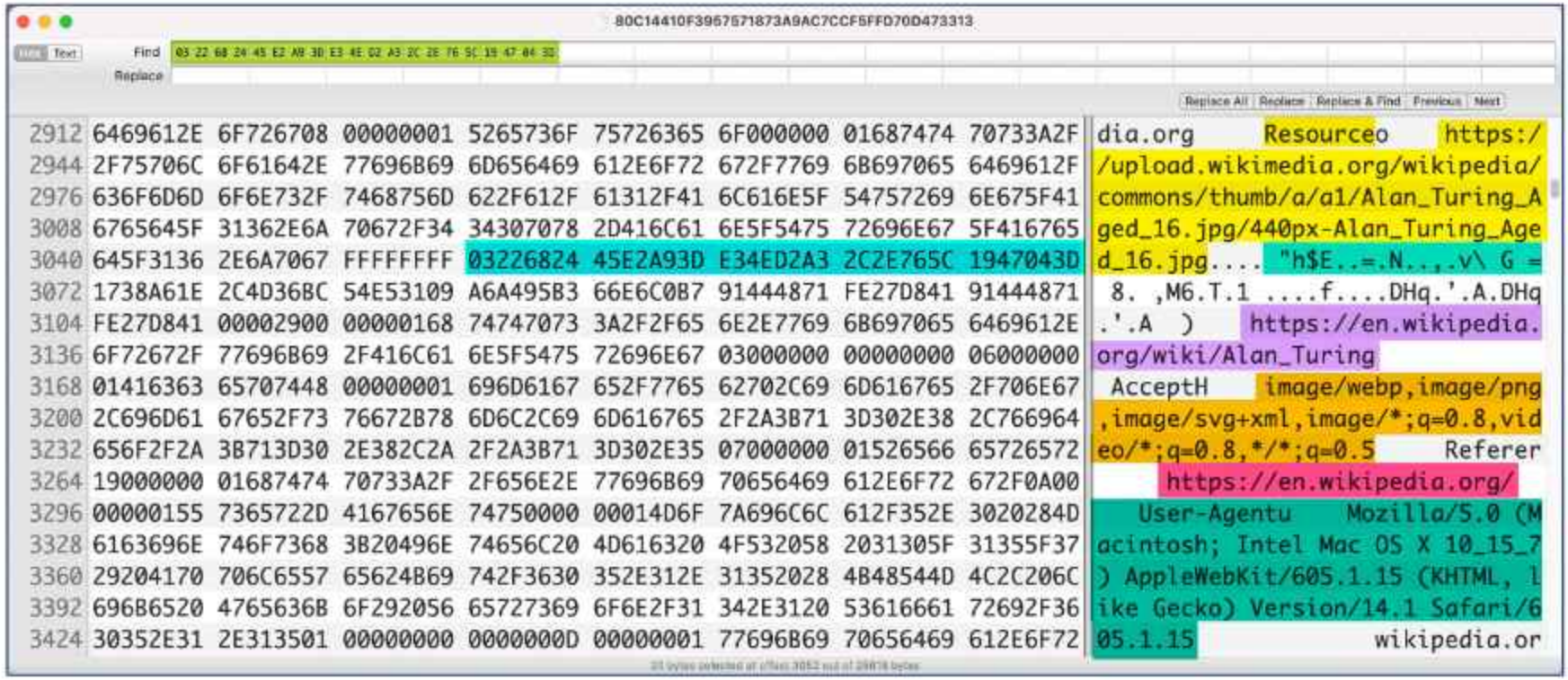
This `SubResource` file contains the listing of cached items for a visit to https://en.wikipedia.org/wiki/Alan_Turing.

The file cache metadata shown above is a sample of what can be found in this file. Each item contains embedded SHA1 hashes that can be correlated across the WebKit data. Show above is the cached metadata for a photo in the Wikpedia article.

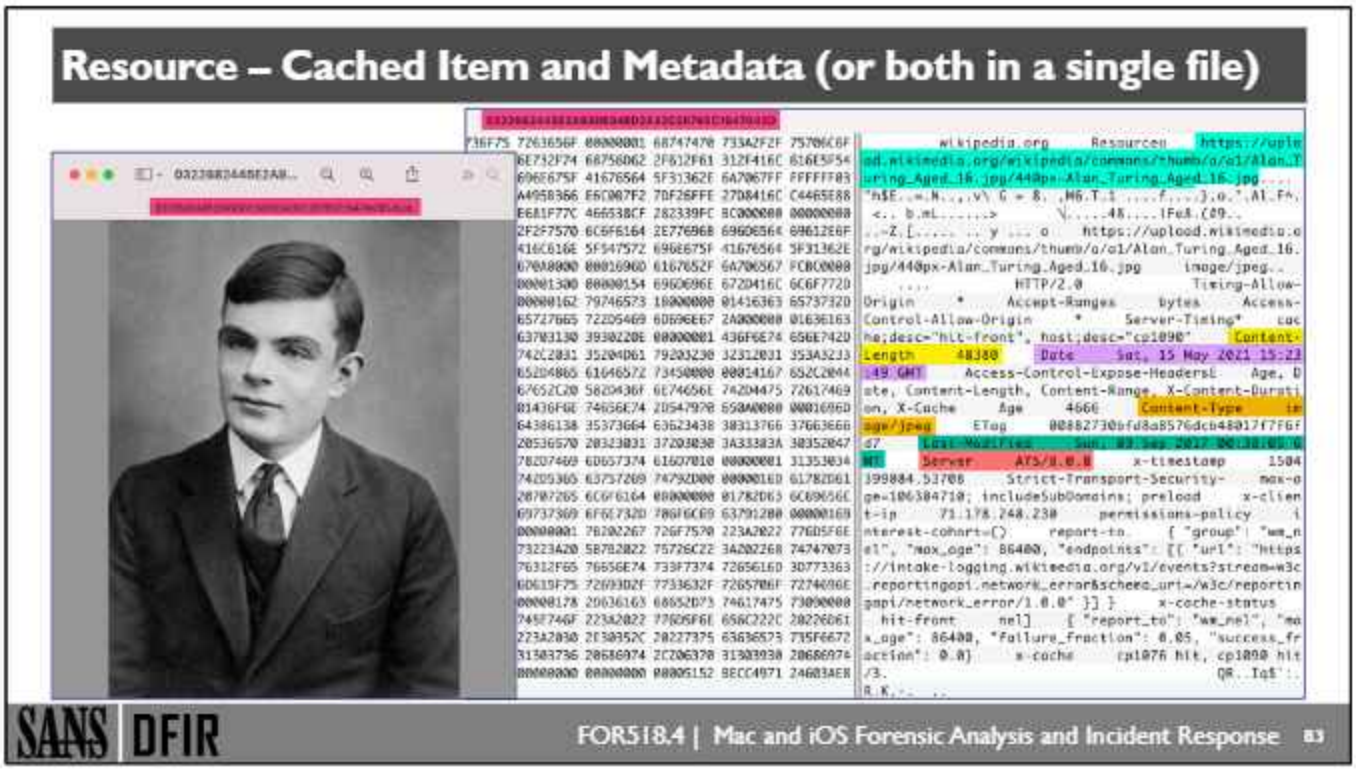
This metadata includes the original source website, the location of the photo on the webserver, and the recorded User Agent info. (It should be noted that the User Agent states 10.15.7, while the Safari browser on macOS 11.3 was actually used.).

Highlighted is an embedded hash that can be used to find the photo in the `Resources` directory, it starts with `0x03226824...`

© 2022 Sarah Edwards



© 2022 Sarah Edwards

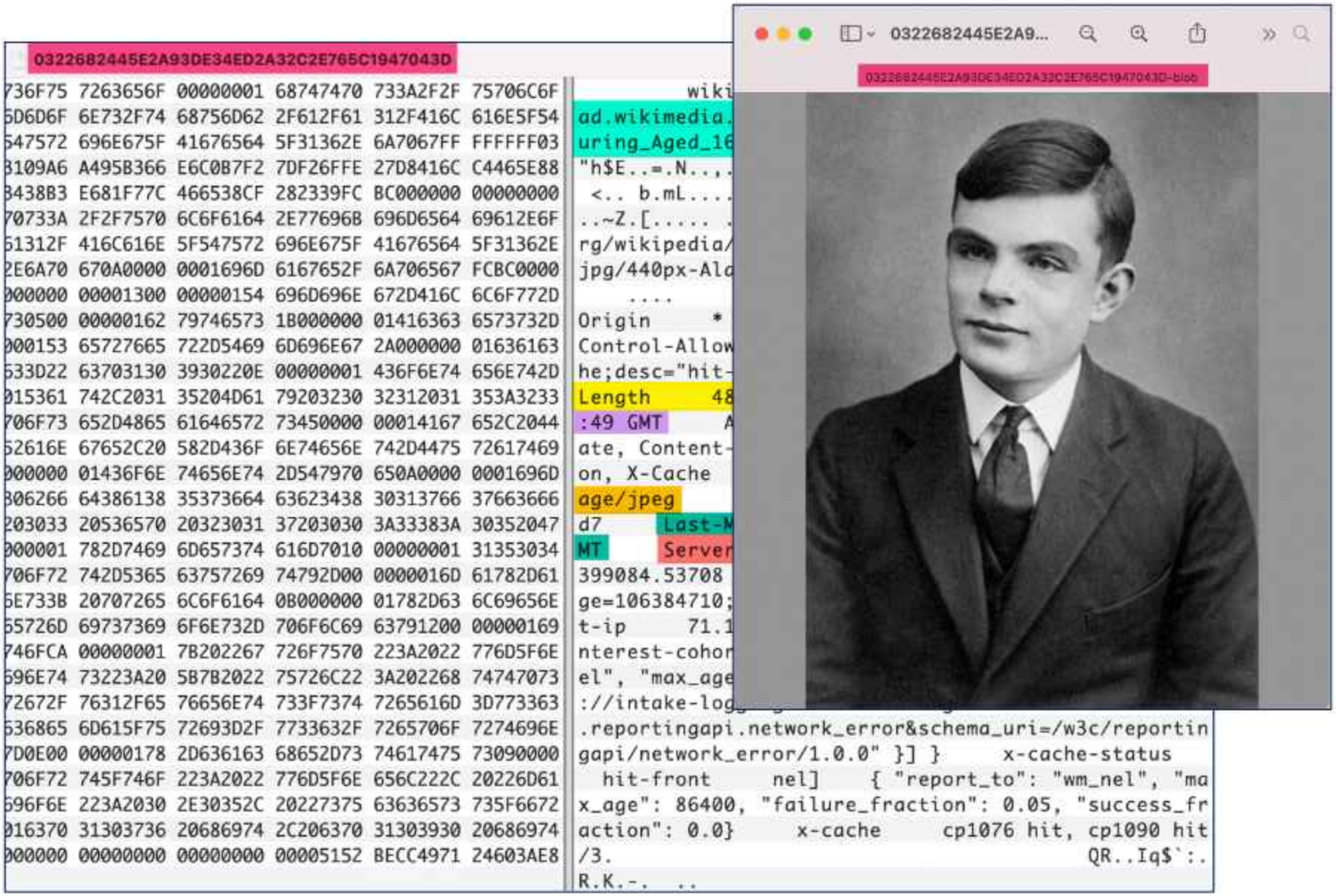


The cached photo of Alan Turing can be found in the `Resources` directory using the embedded SHA1 has found in the `SubResources` file.

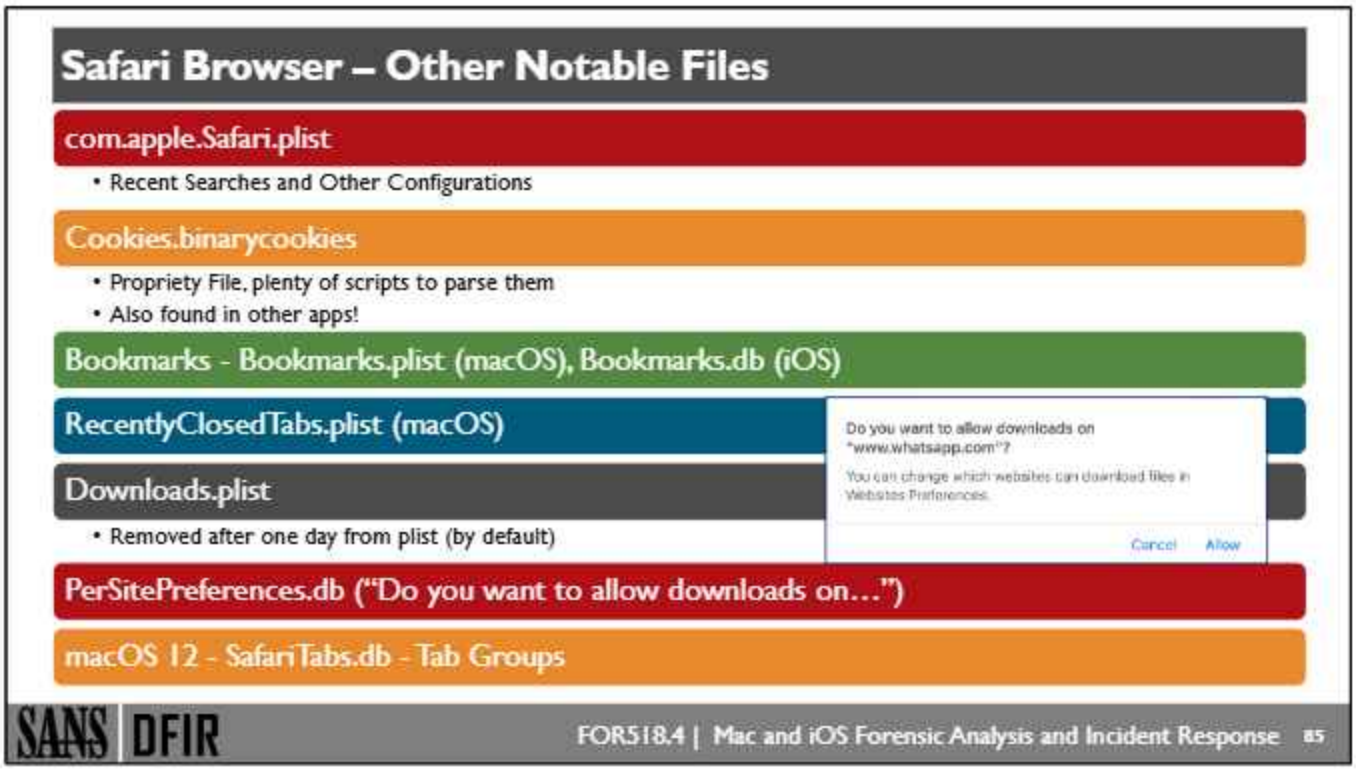
This is stored in two separate files, as it was too large to fit into a single file like others can be. This is often seen with media.

The metadata includes various HTTP information such as timestamps, file sizes, server information, and file type. This also included the embedded SHA1 hash filename of the related file found in the `Blobs` directory.

© 2022 Sarah Edwards



© 2022 Sarah Edwards



The `com.apple.Safari.plist` contains useful configuration details as well as a listing of recent searches performed in the browser.

Cookies for applications are stored in the `Cookies.binarycookies` (also `com.apple.Safari.SafeBrowsing.binarycookie`). This file starts with the file header "`cook`". The file format for these binary cookies is a proprietary format. However, there are many open-source scripts available to parse these files, such as Mari DeGrazia's absolutely fantastic Google Analytic Cookies Forensics presentation at https://github.com/mdegrazia/Presentations and her binary cookie parser: https://github.com/mdegrazia/Safari-Binary-Cookie-Parser.

It is worth noting that it's not just the Safari browser that may have cookies, other applications that have their own internal browsers (i.e., Reading a link posted via Twitter) may have their own WebKit cache and cookies.

Bookmarks are kept in two different file formats for each platform, a plist on macOS and a database on iOS.

The `RecentlyClosedTabs.plist`, keeps a "recent" history of closed tabs. This plist keeps track of the current visible tabs that the user had opened.

The `Downloads.plist` file contains many items that may interest a forensic analyst about the Safari download history. Instead of having the user manually remove items from the Safari Download list, they have instituted a "`Remove download list items`" configuration item that has "`After one day`" selected by default. Other options include: "`When Safari Quits`", "`Upon Successful Download`", or "`Manually`".

This plist is also available on iOS and appears to remove items after one day by default, with options to change it to "`Upon Successful Download`", and "`Manually`".

Root	Dictionary	(1 item)
DownloadHistory	Array	(1 item)
Item 0	Dictionary	(10 items)
DownloadEntryProgressBytesSoFar	Number	102,503,658
DownloadEntryProgressTotalToLoad	Number	102,503,658
DownloadEntryBookmarkBlob	Data	<626F6F6B0403000000000041030000000795B77340
DownloadEntryDateAddedKey	Date	2021-05-15T16:50:21Z
DownloadEntryDateFinishedKey	Date	2021-05-15T16:50:37Z
DownloadEntryIdentifier	String	3884FC4D-A5F2-4A56-94CC-767CFE5CD080
DownloadEntryURL	String	https://scontent-iad3-2.xx.fbcdn.net/v/t39.16592-6/10
DownloadEntryRemoveWhenDoneKey	Boolean	0
DownloadEntryPath	String	/Users/elwoodblues/Downloads/files-WhatsApp.dmg
DownloadEntryShouldUseRequestURLAsOriginURLIfNecessaryKey	Boolean	0

macOS `Downloads.plist`

- **DownloadEntryIdentifier:** Each download has a unique GUID.
- **DownloadEntryURL:** A URL is saved showing where the download originated.
- **DownloadEntryProgressTotalToLoad, ProgressBytesSoFar**: These keys store the total bytes and downloaded bytes for the download. The downloads file may not always have the same quantity of bytes if the download was canceled or otherwise stopped.
- **DownloadEntryPath:** Where the item was downloaded to; very likely will be in the default Downloads directory (`~/Downloads`)
- **DownloadEntryBookmark:** Bookmark BLOB
- **DownloadEntryDateAddedKey, DownloadEntryDateFinishedKey:** The timestamps of when the download started and finished.

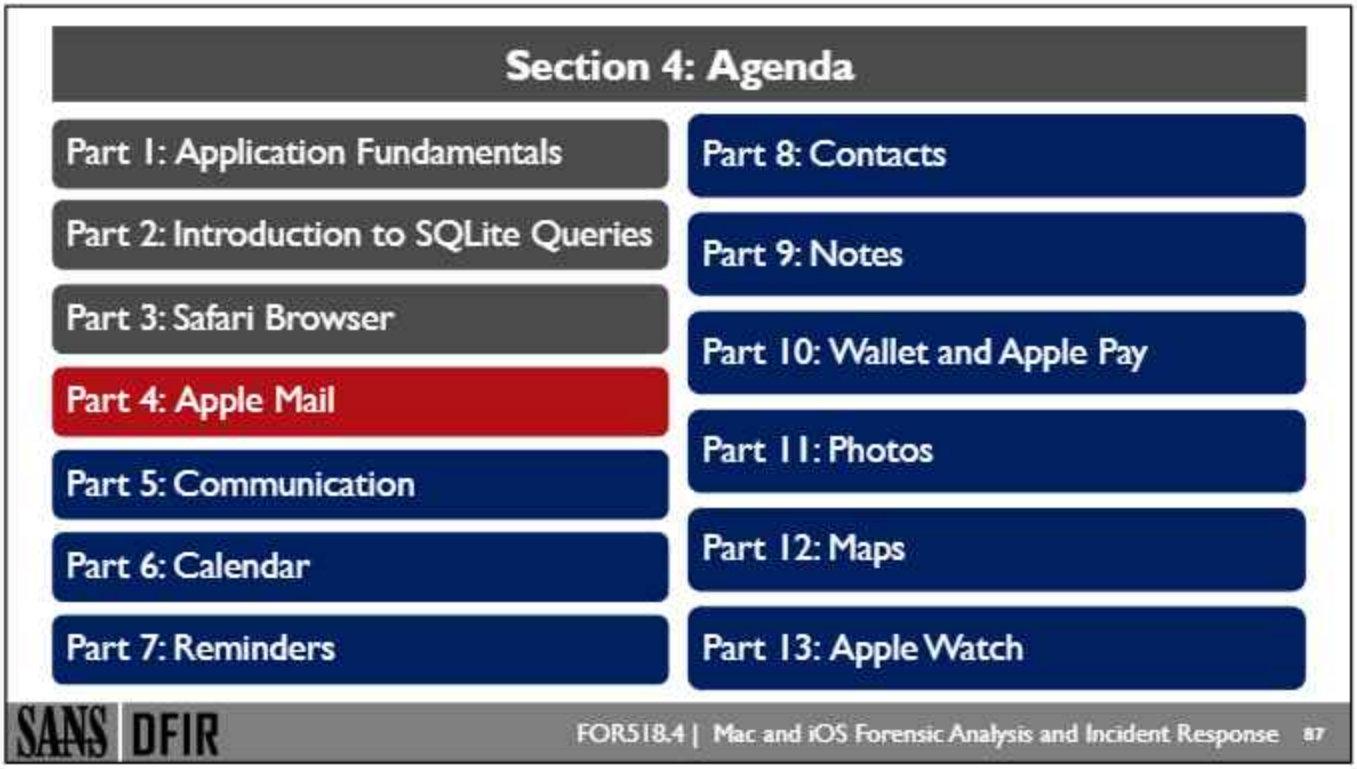
Root	Dictionary	(1 item)
DownloadHistory	Array	(1 item)
Item 0	Dictionary	(12 items)
MIMEType	String	text/csv
State	Number	4
UTI	String	public.comma-separated-values-text
DateFinished	Date	2021-05-13T17:55:06Z
Response	Data	<62706C6973743030D4010203040506070A582476657273696F6E592461726368697665725424
BytesExpected	Number	37,197
BytesLoaded	Number	37,197
DateAdded	Date	2021-05-13T17:55:02Z
Request	Data	<62706C6973743030D4010203040506070A582476657273696F6E592461726368697665725424
Identifier	String	D1FB1EBC-22EB-459B-935D-68D21171223A
BookmarkData	Data	<626F6F6B5C0600000000041030000000000000000000000000000000000000000000000000000000
SourceURL	String	http://www.ticketing-client.com/ticketing-client/csv/EventTicketPromotionPrice.tiksrv?team_id=112&disp

iOS `Downloads.plist`

- **Identifier:** Each download has a unique GUID.
- **SourceURL:** A URL is saved showing where the download originated.
- **BytesExpected, BytesLoaded**: These keys store the total bytes and downloaded bytes for the download. The downloads file may not always have the same quantity of bytes if the download was canceled or otherwise stopped.
- **Response/Request:** Embedded binary plists with HTTP Request/Response data
- **MIMEType/UTI:** File type
- **BookmarkData:** Bookmark BLOB
- **DateAdded, DateFinished:** The timestamps of when the download started and finished.

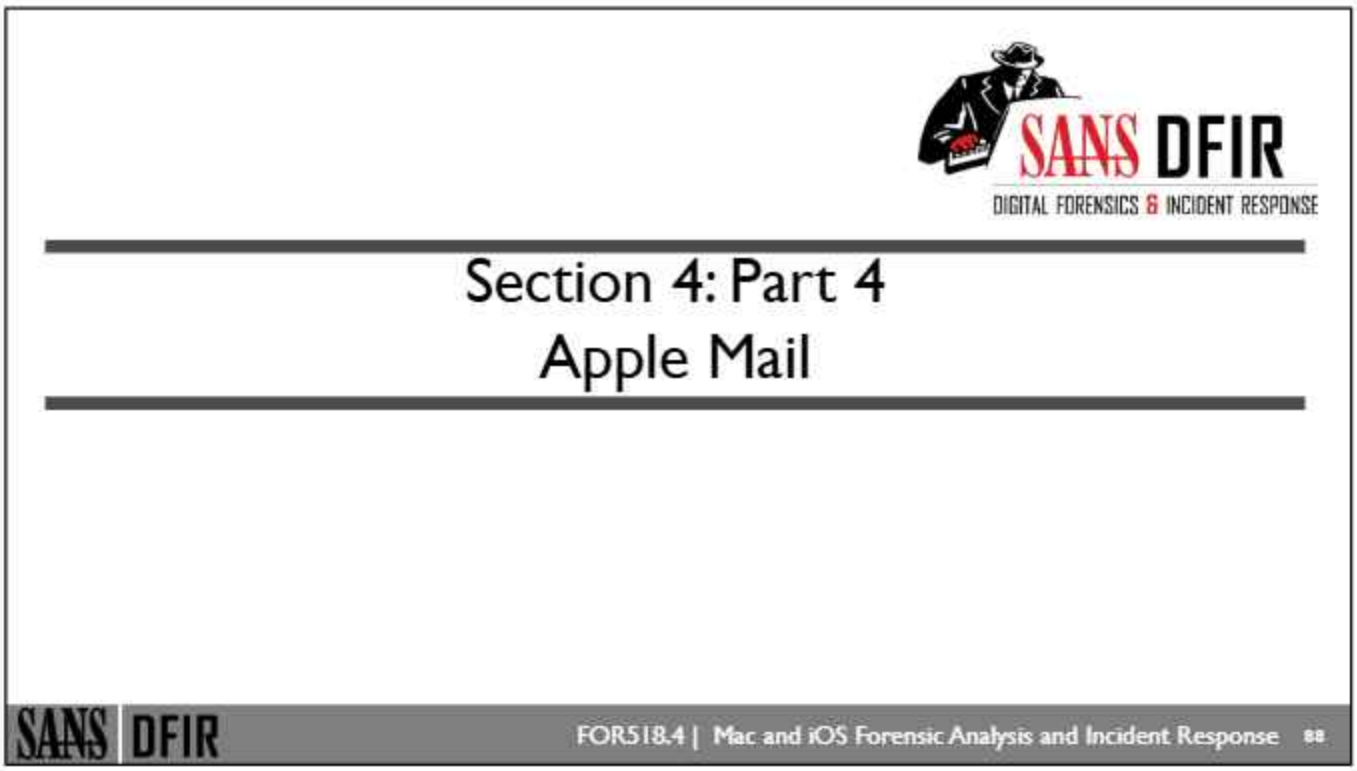
These files may be saved in various locations on iOS devices such as iCloud Drive, "On My iPhone", etc.

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards



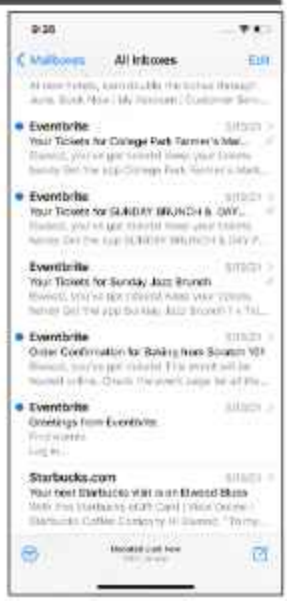
This page intentionally left blank.

© 2022 Sarah Edwards

Apple Mail - com.apple.mail and com.apple.mobilemail

macOS

- ~/Library/Mail/
- ~/Library/Containers/com.apple.mail/



iOS

- iOS Backup: Not much.
- /private/var/mobile/Library/Mail/
- /private/var/mobile/Containers/Data/Application/<GUID>/

Data Types

- Accounts
- Cached Messages and Attachments
- Envelope Index
- Mail Downloads

macOS:
- `~/Library/Mail/`
- `~/Library/Containers/com.apple.mail/`

iOS FFS:
- `/private/var/mobile/Containers/Data/Application/<GUID>`
- `/private/var/mobile/Library/Mail/`

iOS Backup:
- `/mobile/Library/Mail/`

© 2022 Sarah Edwards

macOS Mail - ~/Library/Mail/V#/
[10.13 V5] [10.14 V6] [10.15 V7] [11 V8] [12 V9]

Directory for each email account

- GUID directory name
- Can be correlated in the `Accounts[3|4].sqlite` database

MailData: Mail Application Data

```
root@MBP-M1 V8 # pwd
/Users/elwoodblues/Library/Mail/V8
root@MBP-M1 V8 # ls -l
total 0
drwxr-xr-x@  4 elwoodblues  staff  128 May 15 11:36 0E39064A-8DD9-43F6-B7A8-3BA1039344B2
drwxr-xr-x@  7 elwoodblues  staff  224 May 16 11:14 38330C55-06CA-45F7-93A1-73B737D5931D
drwxr-xr-x@  4 elwoodblues  staff  128 May 16 11:17 53903244-C0B8-49A3-8366-3D8BA537EFF8
drwxr-xr-x  15 elwoodblues  staff  480 May 16 00:00 MailData
```


Each user has a Mail directory in their Library. The "version" number of the Mail directory has changed over the last few macOS versions; however, the general underlying structure has more or less stayed the same.

- 10.13 = V5
- 10.14 = V6
- 10.15 = V7
- 11 = V8

In these directories, you will find GUID-named directories. These can be correlated with various email accounts by using the `Accounts3.sqlite` and `Accounts4.sqlite` database files.

© 2022 Sarah Edwards

macOS Mail – Account Directories: ~/Library/Mail/V#/[GUID]/

Each *.mbox file is a mailbox

An account may have multiple mailboxes

- Inbox
- Sent Messages
- [Gmail]
- Deleted Messages
- Notes
- Drafts
- User Created Mailboxes

```
root@MBP-M1 38330C55-06CA-45F7-93A1-73B737D5931D # pwd
/Users/elwoodblues/Library/Mail/V8/38330C55-06CA-45F7-93A1-73B737D5931D
root@MBP-M1 38330C55-06CA-45F7-93A1-73B737D5931D # ls -la
total 16
drwxr-xr-x@  7 elwoodblues  staff   224 May 16 11:24 .
drwxr-xr-x@  6 elwoodblues  staff   192 May 15 11:37 ..
-rw-r--r--@  1 elwoodblues  staff  5941 May 16 11:24 .mboxCache.plist
drwxr-xr-x@  3 elwoodblues  staff    96 May 15 11:37 Deleted Messages.mbox
drwxr-xr-x@  3 elwoodblues  staff    96 May 15 11:37 INBOX.mbox
drwxr-xr-x@  3 elwoodblues  staff    96 May 15 11:37 Sent Messages.mbox
drwxr-xr-x@ 10 elwoodblues  staff   320 May 15 11:37 [Gmail].mbox
```

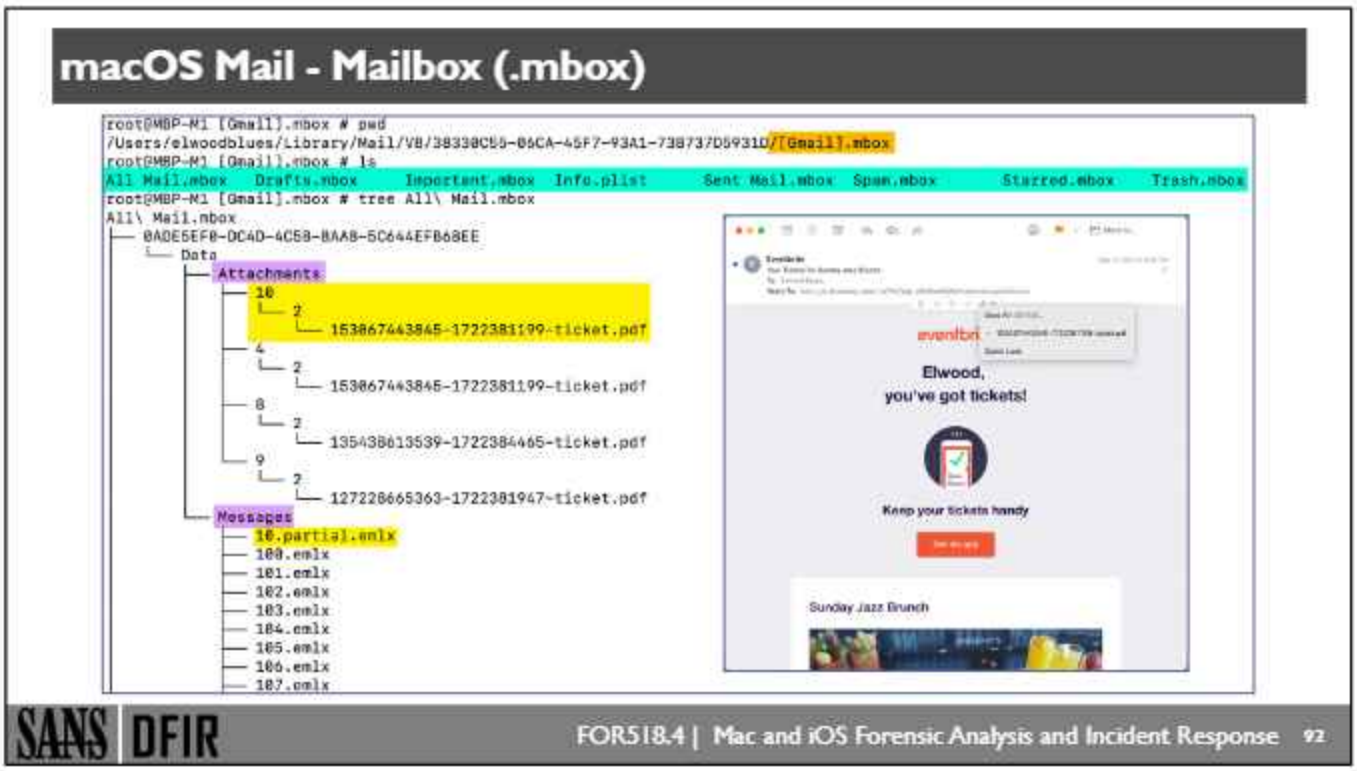

Each Mailbox Account GUID directory may contains one or more mailbox (`.mbox`) directories.

The screenshot shows mailbox (`.mbox`) directories. Depending on the email provider these can be named differently:

- `Archive.mbox`
- `Deleted Messages.mbox`
- `Drafts.mbox`
- `INBOX.mbox`
- `Junk.mbox`
- `Notes.mbox`
- `Sent Messages.mbox`
- `[Gmail].mbox`

The `.mboxCache.plist` file contains some of the organization of the mailboxes.

© 2022 Sarah Edwards



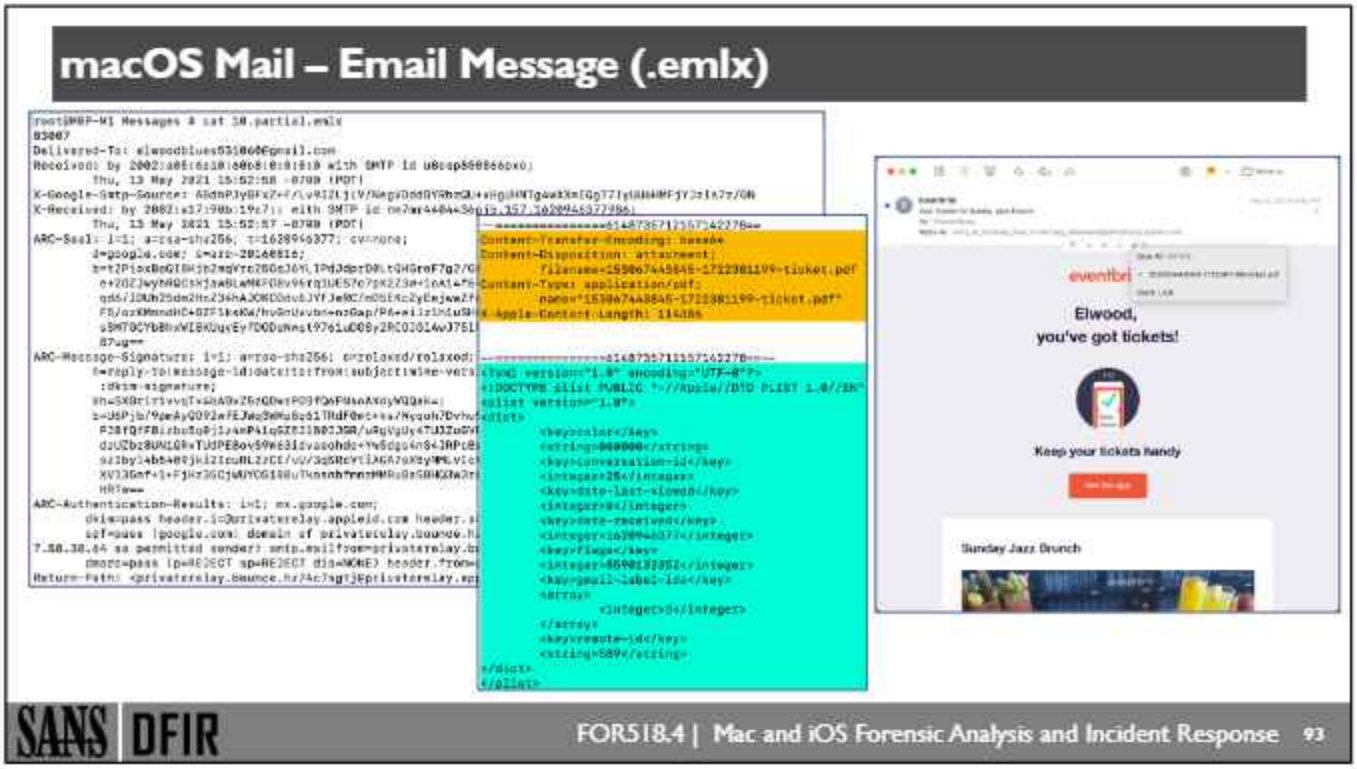
Each mailbox (mbox) may contain nested sub-mailboxes as shown above for this Gmail account. Finally, once through the nested structure, you should find `Messages` and `Attachments` directories for this mailbox.

The GUID directory contains the raw email messages (`.emlx`) and email metadata in the Messages directory.

- `Messages`: Contains the raw email messages (`.emlx`), with an appended property list containing message metadata.
- `Attachments`: Contains the message file attachments.

Highlighted above is message and its associated attachment number 10.

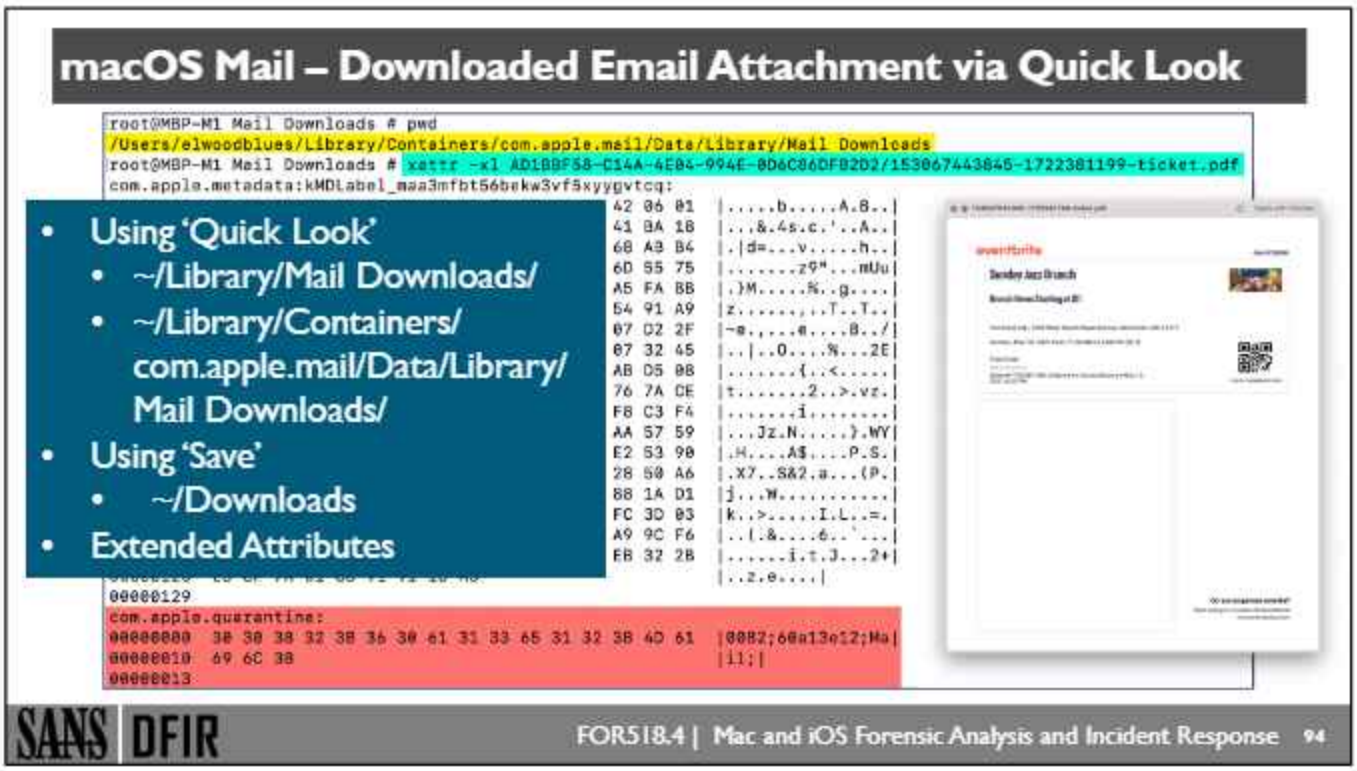
© 2022 Sarah Edwards



EMLX files are plaintext files with the normal email headers and email contents with a tiny plist at the end of the file. The 'partial' emlx files do not necessarily have all items downloaded, thus they are 'partial'.

These kind of email messages are often seen with email with media and rich text. The attachment information is also stored in the emlx file.

© 2022 Sarah Edwards

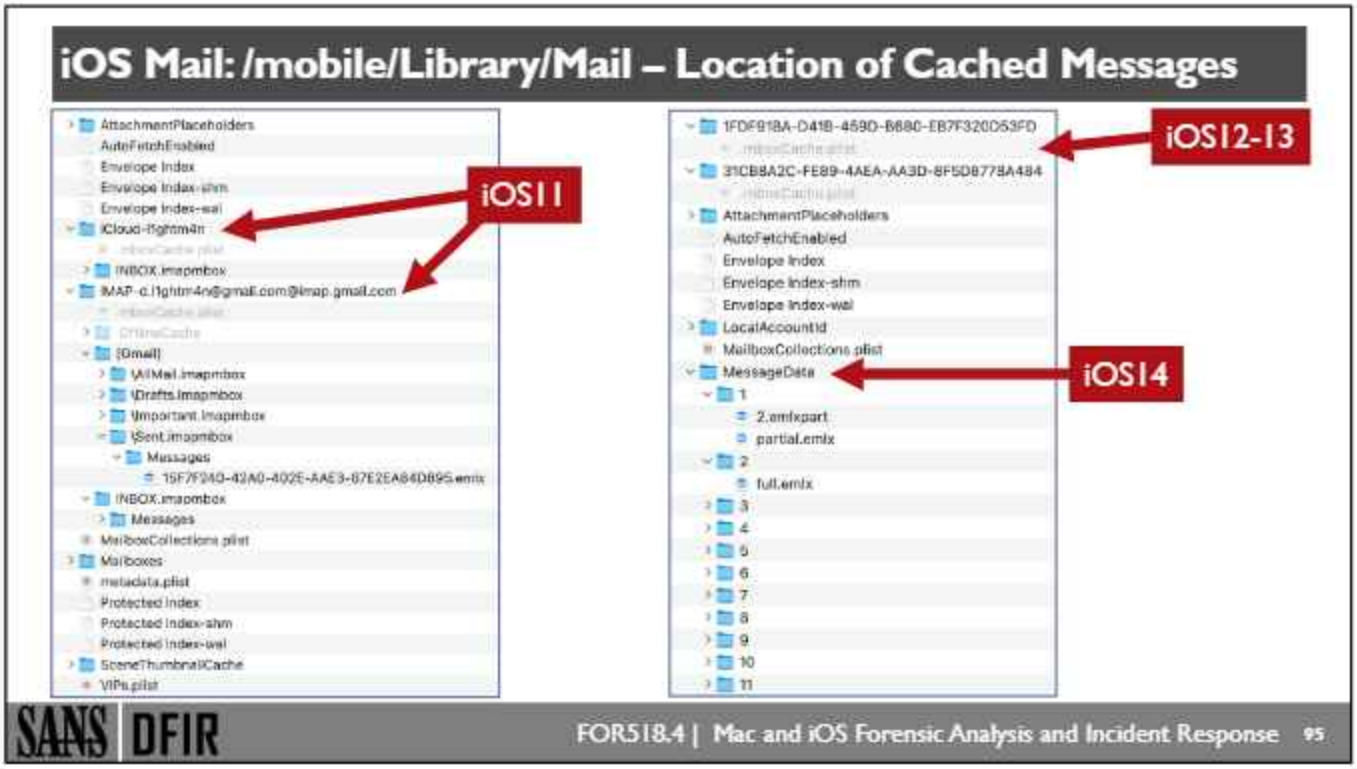


A user can view an attachment in a couple of ways:

- The "`Save`" button will save the attachment in the default Downloads directory, very likely `~/Downloads`.
- The "`Quick Look`" button opens the attachment for the viewer to see quickly and saves the attachment in the `~/Library/Mail Downloads/` directory. The sandbox directory may also be used `~/Library/Containers/com.apple.mail/Data/Library/Mail Downloads/`.

Each email attachment downloaded will have a set of metadata stored in its extended attributes, such as the quarantine extended attribute which can show how and when it was downloaded to the system.

© 2022 Sarah Edwards



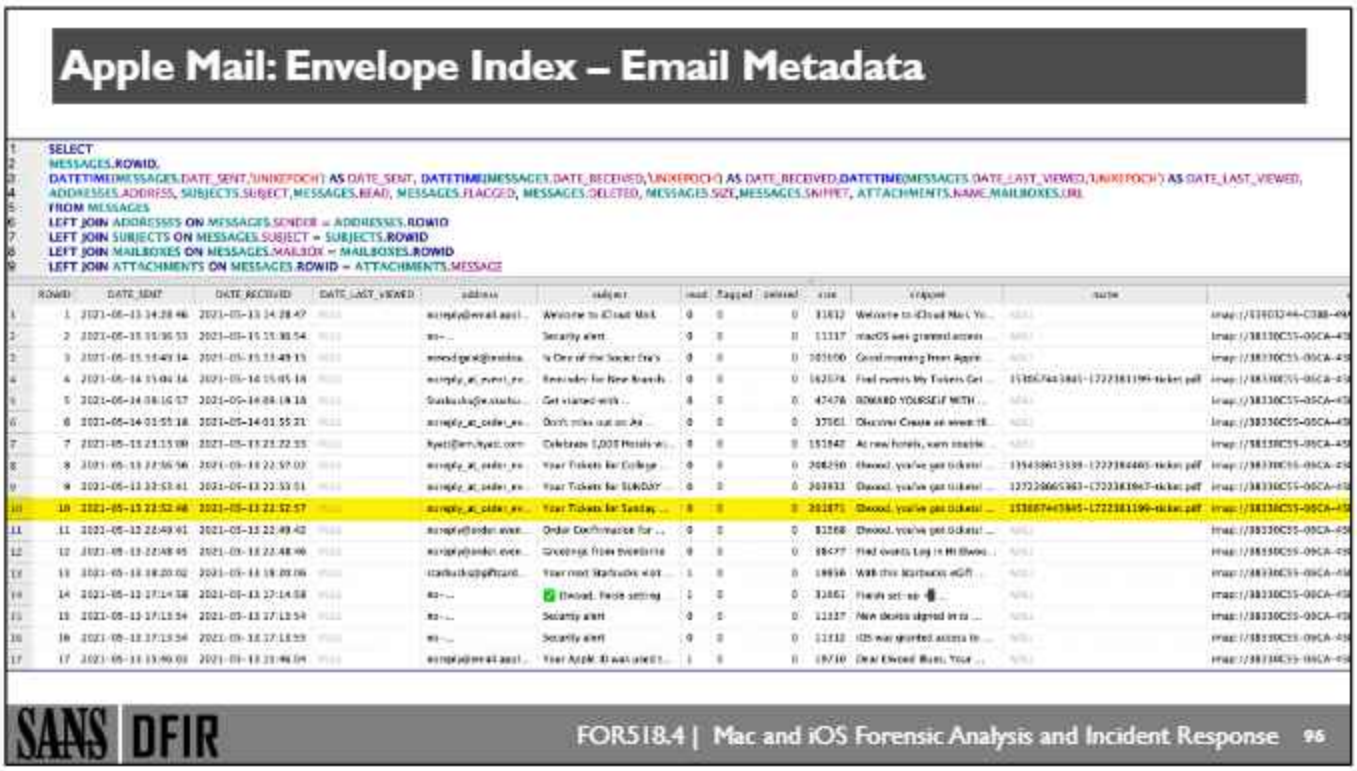
The `Mail` directory on a full file system dump of iOS can contains lots of information. A backup will likely not have most of this information.

Similar to an macOS system, you will have full email messages (at least for what is cached on the device), and detailed message metadata in the `Envelope and Protected Index` databases.

The messages have been moving around over the last few iOS versions but are generally located in the /Mail directory.

The content of email is very different, depending on what type of acquisition you have. Each acquisition type has two areas where email data is found.

The `mboxCache.plist` files contain the organizational structure of each email account. This property list file contains the folder names the email messages are organized into. In these files, you may find a folder of interest, depending on how the user has organized their email.



macOS:

- `~/Library/Mail/`

iOS FFS:

- `/private/var/mobile/Library/Mail/`

The SQLite database `Envelope Index`, located in the `MailData` directory, contains indexed mail data.

- The `addresses` table contains all the indexed email addresses and associated contact names.
- The `attachments` table contains the name of each attachment.
- The `mailboxes` table contains data for each mailbox, including total messages and unread messages.
- The `messages` table contains metadata for each email message, including To, From, Subject, email timestamps, and if the message was read or not.
- The `subjects` table contains the email subject for each email message.

In the screenshot above, the actual email can be found by looking for the `ROWID`.

Emails that have been read will contain a '`1`' in the `read` column, as will messages that have been `flagged` or `deleted`, in their respective columns. The `url` column shows the path to the email account—this will contain the GUID associated with the account (also found in the `Accounts` database). The address column shows the sender's email address.

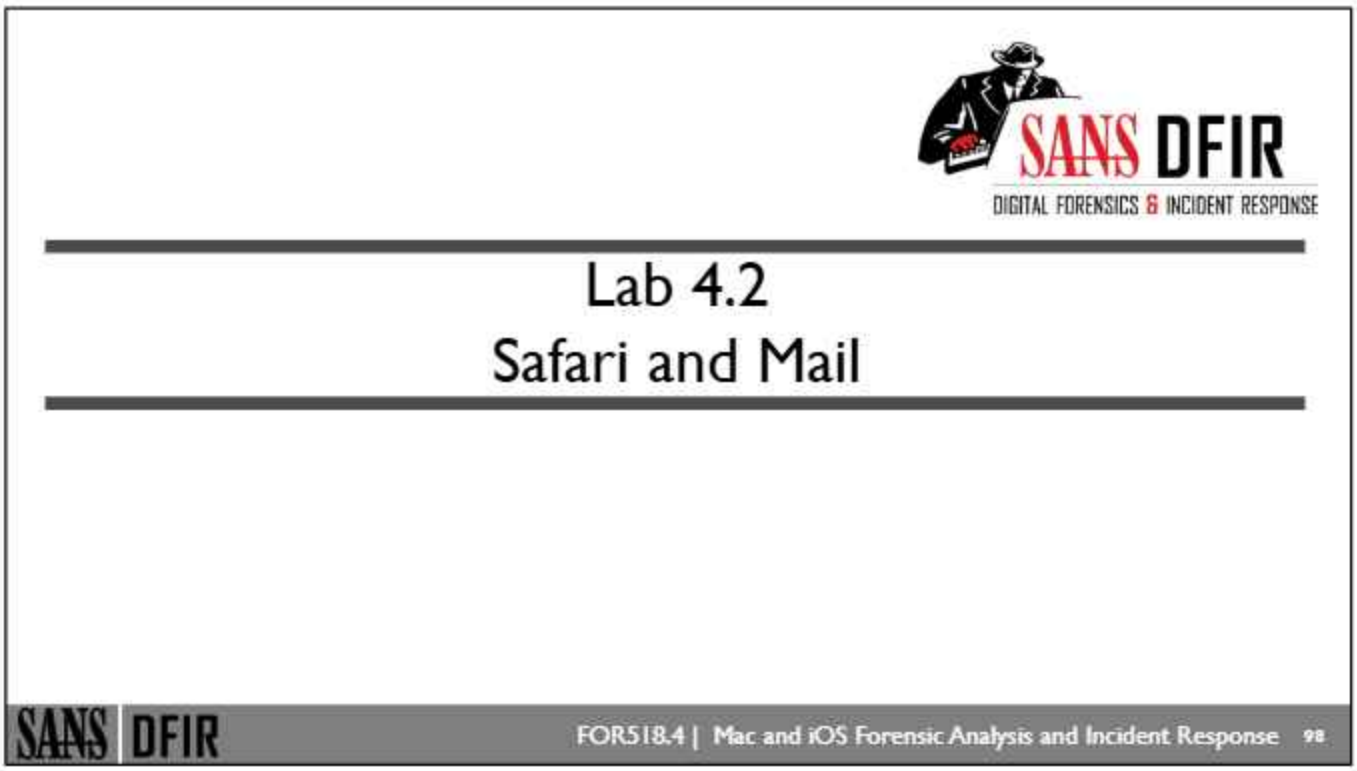
On iOS, some of these details will be in the Protected Index database in the same directory.

© 2022 Sarah Edwards

```
SELECT
MESSAGES.ROWID,
DATETIME(MESSAGES.DATE_SENT,'UNIXEPOCH') AS DATE_SENT, DATETIME(MESSAGES.DATE_RECEIVED,'UNIXEPOCH') AS DATE_RECEIVED,DATETIME(MESSAGES.DATE_LAST_VIEWED,'UNIXEPOCH') AS DATE_LAST_VIEWED,
ADDRESSES.ADDRESS, SUBJECTS.SUBJECT,MESSAGES.READ, MESSAGES.FLAGGED, MESSAGES.DELETED, MESSAGES.SIZE,MESSAGES.SNIPPET, ATTACHMENTS.NAME,MAILBOXES.URL
FROM MESSAGES
LEFT JOIN ADDRESSES ON MESSAGES.SENDER = ADDRESSES.ROWID
LEFT JOIN SUBJECTS ON MESSAGES.SUBJECT = SUBJECTS.ROWID
LEFT JOIN MAILBOXES ON MESSAGES.MAILBOX = MAILBOXES.ROWID
LEFT JOIN ATTACHMENTS ON MESSAGES.ROWID = ATTACHMENTS.MESSAGE
```

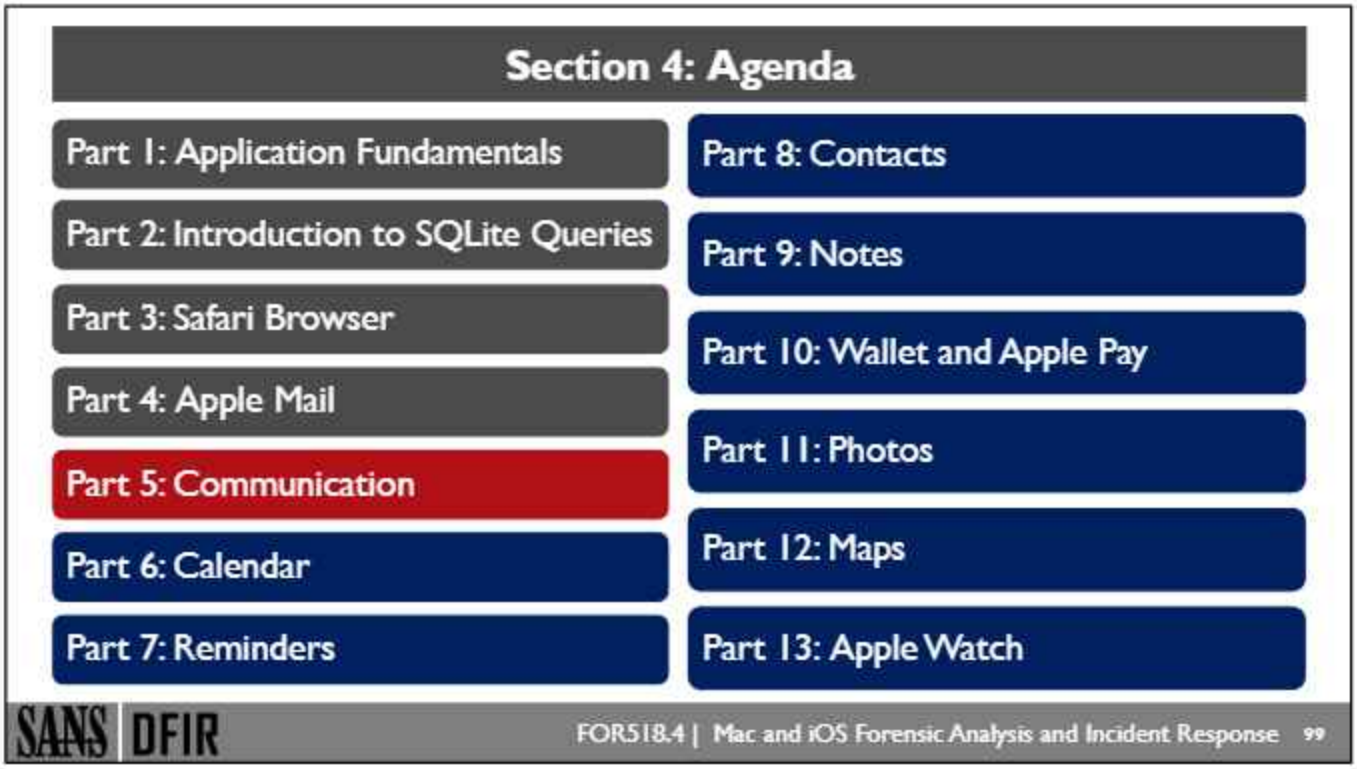
	ROWID	DATE_SENT	DATE_RECEIVED	DATE_LAST_VIEWED	address	subject	read	flagged	deleted	size	snippet	name	
1	1	2021-05-13 14:28:46	2021-05-13 14:28:47	NULL	noreply@email.appl...	Welcome to iCloud Mail.	0	0	0	31812	Welcome to iCloud Mail. Yo...	NULL	imap://53903244-C088-49
2	2	2021-05-15 15:36:53	2021-05-15 15:36:54	NULL	no-...	Security alert	0	0	0	11317	macOS was granted access ...	NULL	imap://38330C55-06CA-45
3	3	2021-05-15 13:49:14	2021-05-15 13:49:15	NULL	newsdigest@insidea...	Is One of the Soviet Era's ...	0	0	0	101690	Good morning from Apple ...	NULL	imap://38330C55-06CA-45
4	4	2021-05-14 15:04:14	2021-05-14 15:05:18	NULL	noreply_at_event_ev...	Reminder for New Brunch...	0	0	0	162074	Find events My Tickets Get ...	153067443845-1722381199-ticket.pdf	imap://38330C55-06CA-45
5	5	2021-05-14 09:16:57	2021-05-14 09:19:18	NULL	Starbucks@e.starbu...	Get started with ...	0	0	0	47478	REWARD YOURSELF WITH ...	NULL	imap://38330C55-06CA-45
6	6	2021-05-14 01:55:18	2021-05-14 01:55:21	NULL	noreply_at_order_ev...	Don't miss out on An ...	0	0	0	37061	Discover Create an event Hi...	NULL	imap://38330C55-06CA-45
7	7	2021-05-13 23:13:00	2021-05-13 23:22:33	NULL	hyatt@em.hyatt.com	Celebrate 1,000 Hotels wi...	0	0	0	151842	At new hotels, earn double ...	NULL	imap://38330C55-06CA-45
8	8	2021-05-13 22:56:56	2021-05-13 22:57:02	NULL	noreply_at_order_ev...	Your Tickets for College ...	0	0	0	208250	Elwood, you've got tickets! ...	135438613539-1722384465-ticket.pdf	imap://38330C55-06CA-45
9	9	2021-05-13 22:53:41	2021-05-13 22:53:51	NULL	noreply_at_order_ev...	Your Tickets for SUNDAY ...	0	0	0	203983	Elwood, you've got tickets! ...	127228661363-1722381947-ticket.pdf	imap://38330C55-06CA-45
10	10	2021-05-13 22:52:48	2021-05-13 22:52:57	NULL	noreply_at_order_ev...	Your Tickets for Sunday ...	0	0	0	201871	Elwood, you've got tickets! ...	153067443845-1722381199-ticket.pdf	imap://38330C55-06CA-45
11	11	2021-05-13 22:49:41	2021-05-13 22:49:42	NULL	noreply@order.even...	Order Confirmation for ...	0	0	0	81568	Elwood, you've got tickets! ...	NULL	imap://38330C55-06CA-45
12	12	2021-05-13 22:48:45	2021-05-13 22:48:46	NULL	noreply@order.even...	Greetings from Eventbrite	0	0	0	38477	Find events Log in Hi Elwoo...	NULL	imap://38330C55-06CA-45
13	13	2021-05-13 18:20:02	2021-05-13 18:20:06	NULL	starbucks@giftcard...	Your next Starbucks visit ...	1	0	0	19856	With this Starbucks eGift ...	NULL	imap://38330C55-06CA-45
14	14	2021-05-13 17:14:58	2021-05-13 17:14:58	NULL	no-...	Elwood, finish setting ...	1	0	0	31661	Finish set-up -[redacted] ...	NULL	imap://38330C55-06CA-45
15	15	2021-05-13 17:13:54	2021-05-13 17:13:54	NULL	no-...	Security alert	0	0	0	11337	New device signed in to ...	NULL	imap://38330C55-06CA-45
16	16	2021-05-13 17:13:54	2021-05-13 17:13:55	NULL	no-...	Security alert	0	0	0	11312	iOS was granted access to ...	NULL	imap://38330C55-06CA-45
17	17	2021-05-13 15:46:03	2021-05-13 15:46:04	NULL	noreply@email.appl...	Your Apple ID was used t...	1	0	0	19710	Dear Elwood Blues, Your ...	NULL	imap://38330C55-06CA-45

© 2022 Sarah Edwards



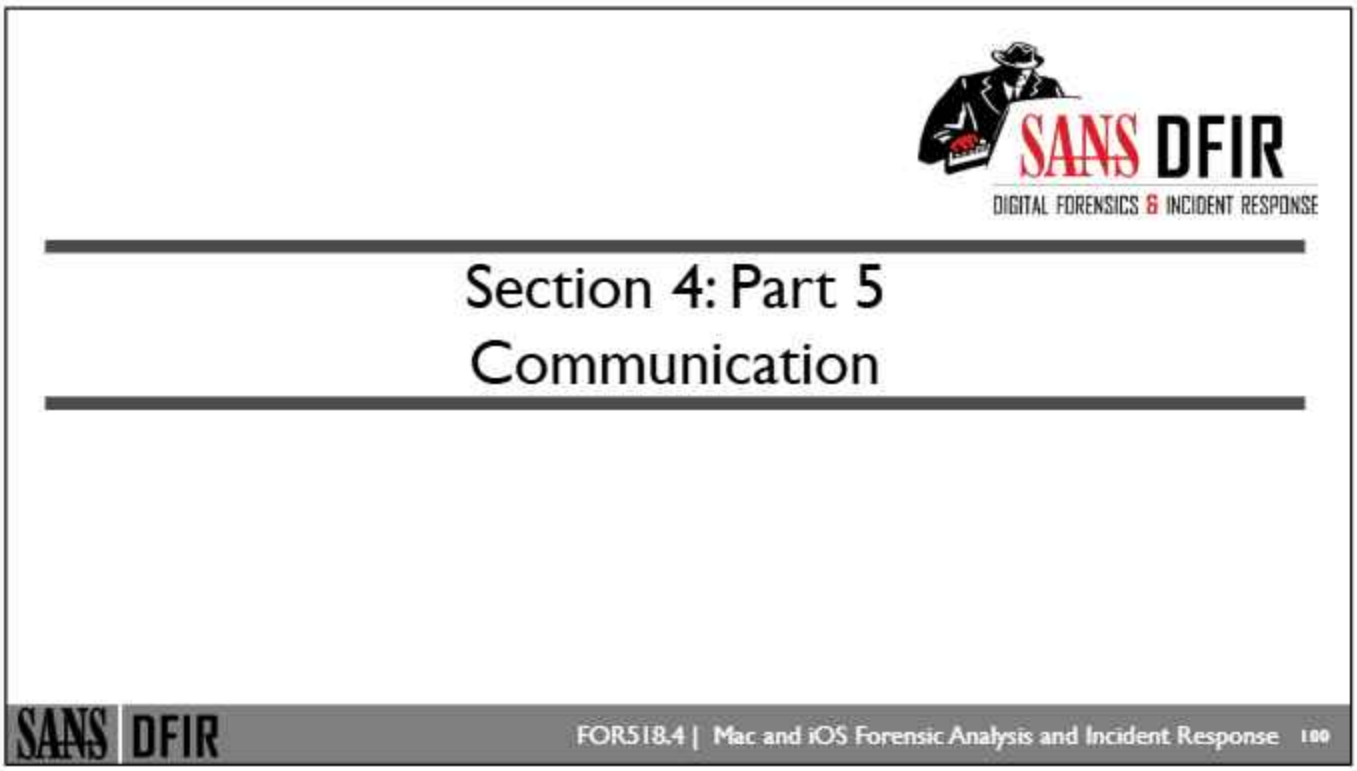
This page intentionally left blank.

© 2022 Sarah Edwards



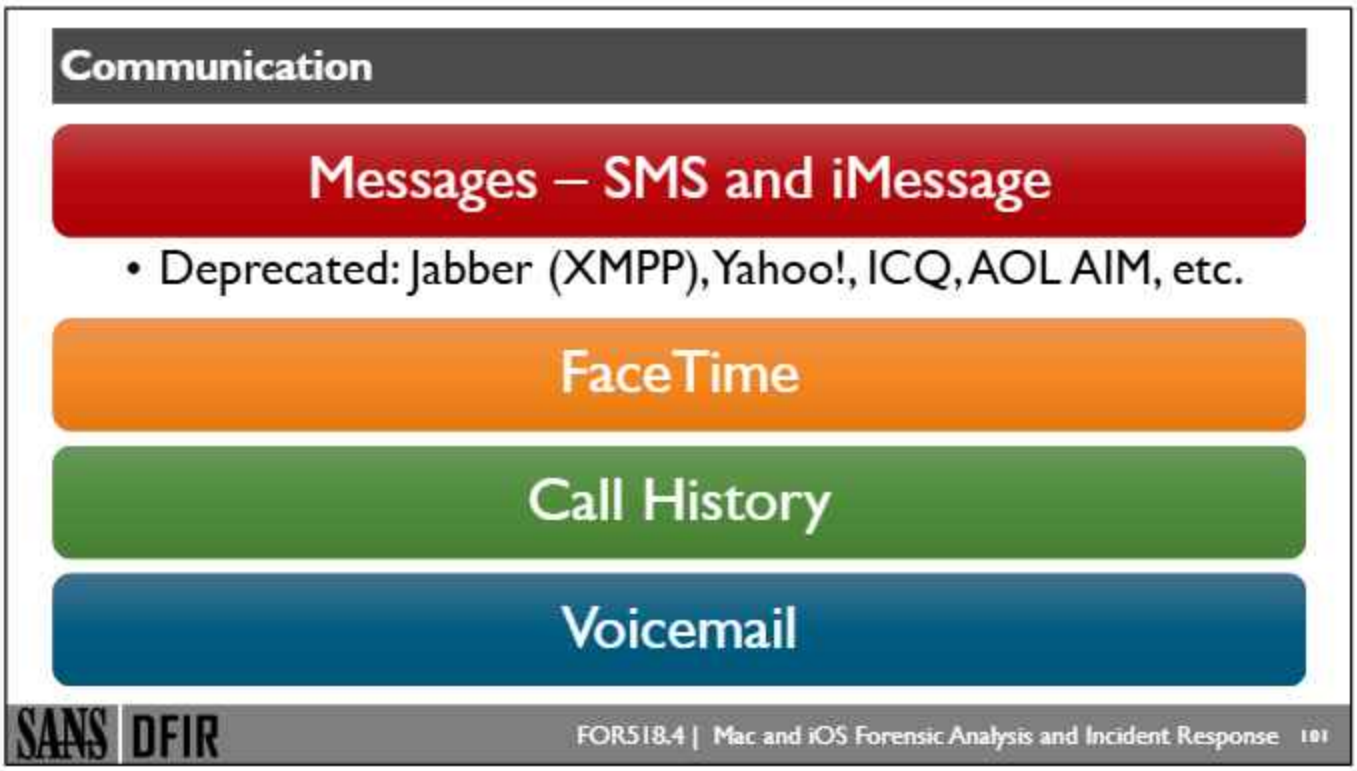
This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.

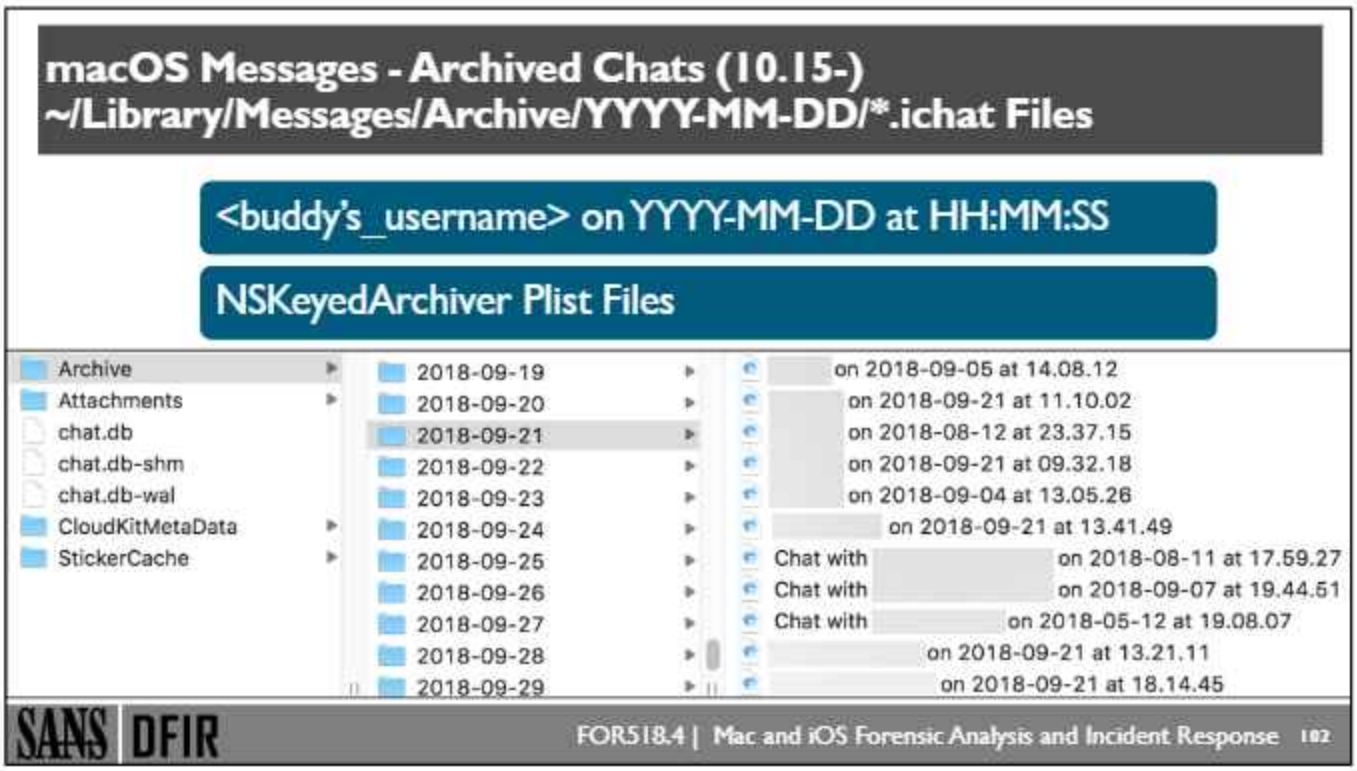
© 2022 Sarah Edwards



Many apps can be used to communicate on Apple devices.

Messages is the native instant messaging application. This application can be used with a variety of instant messaging programs and protocols (some of which are deprecated but still may be found in systems that have been upgraded over time.).

Calls may be found in the Phone or FaceTime applications, and voicemails are also located in its own area of iOS devices.



Messages will also store chat conversations if configured to do so. These files are stored in the `~/Library/Messages/Archive/` directory, also organized by date. These files use the same `iChat` binary property list format saved by date and time.

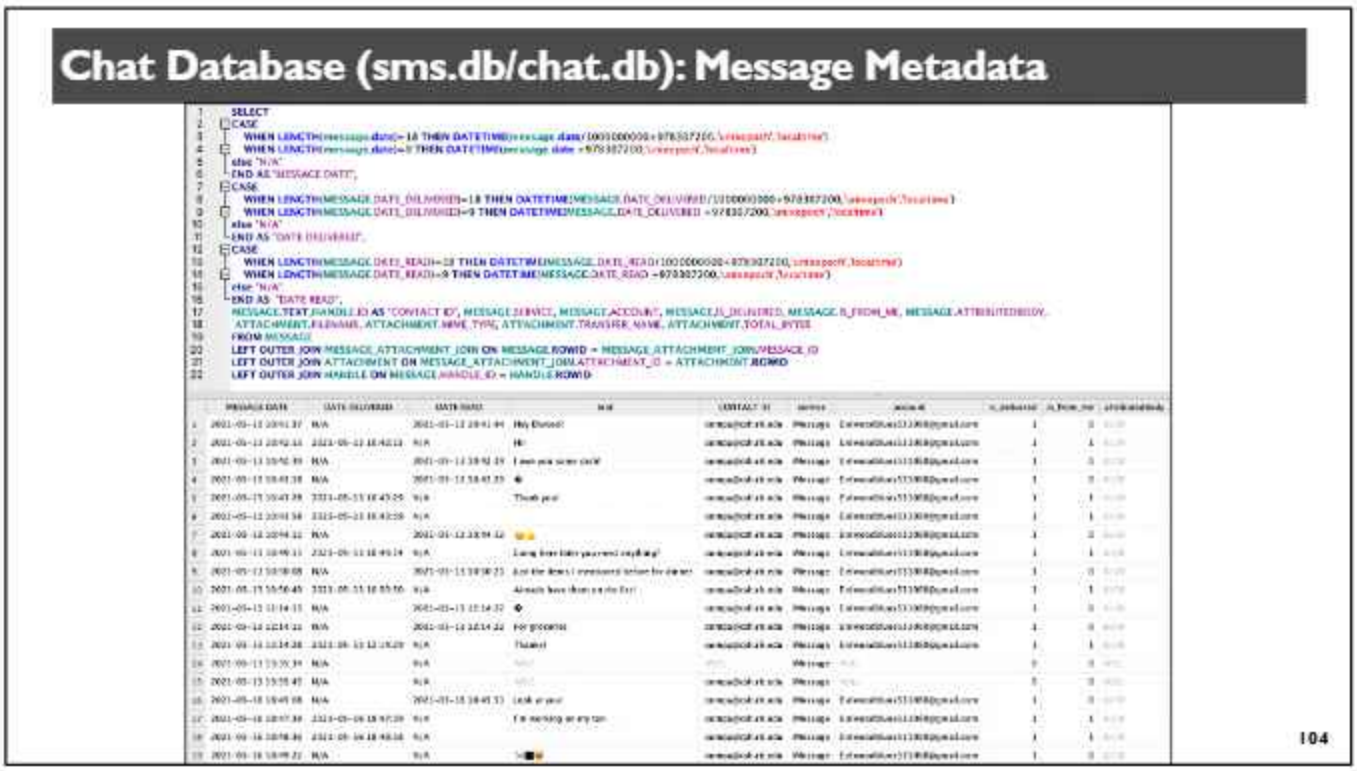
As of macOS 10.15 these files do not appear to be created, instead just the database is used.

© 2022 Sarah Edwards

macOS: `~/Library/Messages/chat.db`
iOS FFS: `/private/var/mobile/Library/SMS/sms.db`
iOS Backup: `/mobile/Library/SMS/sms.db`

The Messages application on iOS can be used for both iMessage and SMS-based messaging. The SMS directory will contain a SQLite database, `SMS.db`, and an `/Attachments` directory. On macOS the database is in the User's Messages directory and named chat.db.

The `SMS.db`/`chat.db` database file contains the message and related metadata, while the `/Attachments` directory contains media attachments.



macOS: `~/Library/Messages/chat.db`
iOS FFS: `/private/var/mobile/Library/SMS/sms.db`
iOS Backup: `/mobile/Library/SMS/sms.db`

A query on the `sms.db` SQLite database file shown below will extract the messages, timestamps, text, contact numbers/emails, originating account, attachment data, whether the message was delivered, and who originated it. The timestamps with "N/A" above originally contained a "0". These timestamps are in Mac Epoch. A specific CASE statement was created to deal with variable lengths of timestamps in the timestamp columns.

The text column may not always have a message contained within in situations where attachments are sent without a message or in situations where other services were used such enabling locations (the NULL entries, lines14 & 15), and Apple Pay was used (Lines 4 & 11).

Some context can be obtained from BLOB data in the `attributedBody` for items like Apple Pay transactions.

Messages may contain emojis. These messages can be reviewed from the binary view to see the Unicode that each emoji uses. This is handy if your SQLite viewer does not support Unicode.

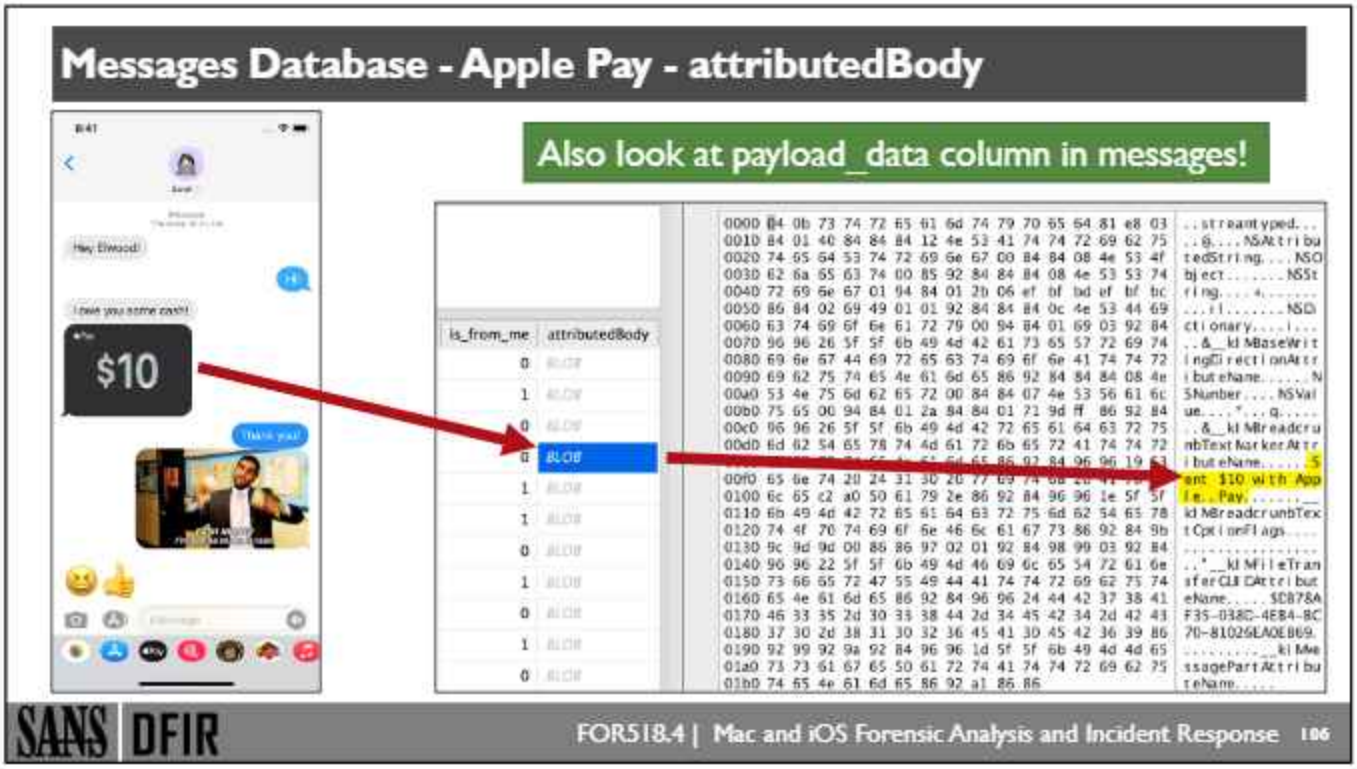
References:
https://apps.timwhitlock.info/emoji/tables/unicode
http://www.unicode.org/emoji/charts/full-emoji-list.html
https://smarterforensics.com/2017/09/time-is-not-on-our-side-when-it-comes-to-messages-in-ios-11/

© 2022 Sarah Edwards

```
1   SELECT
2   CASE
3       WHEN LENGTH(message.date)=18 THEN DATETIME(message.date/1000000000+978307200,'unixepoch','localtime')
4       WHEN LENGTH(message.date)=9 THEN DATETIME(message.date +978307200,'unixepoch','localtime')
5     else "N/A"
6     END AS "MESSAGE DATE",
7   CASE
8       WHEN LENGTH(MESSAGE.DATE_DELIVERED)=18 THEN DATETIME(MESSAGE.DATE_DELIVERED/1000000000+978307200,'unixepoch','localtime')
9       WHEN LENGTH(MESSAGE.DATE_DELIVERED)=9 THEN DATETIME(MESSAGE.DATE_DELIVERED +978307200,'unixepoch','localtime')
10    else "N/A"
11    END AS "DATE DELIVERED",
12  CASE
13      WHEN LENGTH(MESSAGE.DATE_READ)=18 THEN DATETIME(MESSAGE.DATE_READ/1000000000+978307200,'unixepoch','localtime')
14      WHEN LENGTH(MESSAGE.DATE_READ)=9 THEN DATETIME(MESSAGE.DATE_READ +978307200,'unixepoch','localtime')
15    else "N/A"
16    END AS  "DATE READ",
17    MESSAGE.TEXT,HANDLE.ID AS "CONTACT ID", MESSAGE.SERVICE, MESSAGE.ACCOUNT, MESSAGE.IS_DELIVERED, MESSAGE.IS_FROM_ME, MESSAGE.ATTRIBUTEDBODY,
18     ATTACHMENT.FILENAME, ATTACHMENT.MIME_TYPE, ATTACHMENT.TRANSFER_NAME, ATTACHMENT.TOTAL_BYTES
19    FROM MESSAGE
20    LEFT OUTER JOIN MESSAGE_ATTACHMENT_JOIN ON MESSAGE.ROWID = MESSAGE_ATTACHMENT_JOIN.MESSAGE_ID
21    LEFT OUTER JOIN ATTACHMENT ON MESSAGE_ATTACHMENT_JOIN.ATTACHMENT_ID = ATTACHMENT.ROWID
22    LEFT OUTER JOIN HANDLE ON MESSAGE.HANDLE_ID = HANDLE.ROWID
```

	MESSAGE DATE	DATE DELIVERED	DATE READ	text	CONTACT ID	service	account	is_delivered	is_from_me	attributedBody
1	2021-05-13 10:41:37	N/A	2021-05-13 10:41:44	Hey Elwood!	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	0	BLOB
2	2021-05-13 10:42:13	2021-05-13 10:42:13	N/A	Hi!	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	1	BLOB
3	2021-05-13 10:42:39	N/A	2021-05-13 10:42:39	I owe you some cash!	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	0	BLOB
4	2021-05-13 10:43:18	N/A	2021-05-13 10:43:20	￼	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	0	BLOB
5	2021-05-13 10:43:28	2021-05-13 10:43:29	N/A	Thank you!	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	1	BLOB
6	2021-05-13 10:43:58	2021-05-13 10:43:59	N/A		oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	1	BLOB
7	2021-05-13 10:44:11	N/A	2021-05-13 10:44:12	😊👍	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	0	BLOB
8	2021-05-13 10:49:13	2021-05-13 10:49:14	N/A	Going here later you need anything?	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	1	BLOB
9	2021-05-13 10:50:08	N/A	2021-05-13 10:50:21	Just the items I mentioned before for dinner	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	0	BLOB
10	2021-05-13 10:50:49	2021-05-13 10:50:50	N/A	Already have them on the list!	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	1	BLOB
11	2021-05-13 12:14:13	N/A	2021-05-13 12:14:22	￼	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	0	BLOB
12	2021-05-13 12:14:13	N/A	2021-05-13 12:14:22	For groceries	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	0	BLOB
13	2021-05-13 12:14:28	2021-05-13 12:14:29	N/A	Thanks!	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	1	BLOB
14	2021-05-13 13:35:34	N/A	N/A	NULL	NULL	iMessage	NULL	0	0	NULL
15	2021-05-13 13:35:45	N/A	N/A	NULL	oompa@csh.rit.edu	iMessage	NULL	0	0	NULL
16	2021-05-16 18:45:08	N/A	2021-05-16 18:45:51	Look at you!	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	0	BLOB
17	2021-05-16 18:47:39	2021-05-16 18:47:39	N/A	I'm working on my tan	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	1	BLOB
18	2021-05-16 18:48:36	2021-05-16 18:48:38	N/A		oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	1	BLOB
19	2021-05-16 18:49:22	N/A	N/A	🙌🏾🔥😻	oompa@csh.rit.edu	iMessage	E:elwoodblues531060@gmail.com	1	0	BLOB

© 2022 Sarah Edwards



macOS: `~/Library/Messages/chat.db`
iOS FFS: `/private/var/mobile/Library/SMS/sms.db`
iOS Backup: `/mobile/Library/SMS/sms.db`

Apple Pay transaction details are not listed in the text area but rather in `attributedBody` and `payload_data` columns of the messages table.

The `payload_data` contains an embedded binary plist with similar information. The `attributedBody` contains a legacy format blob 'NeXT/Apple typedstream data'.

© 2022 Sarah Edwards

Chat Database (sms.db/chat.db): Chat Attachments

- Photos, Videos, Maps, Safari Links, Locations, GIFs, Animoji, etc.

macOS: `~/Library/Messages/chat.db`
iOS FFS: `/private/var/mobile/Library/SMS/sms.db`
iOS Backup: `/mobile/Library/SMS/sms.db`

The "`filename`" column will hold a path to the attachment on the file system. An attachment does not have to be a picture or video. The "`mime_type`" column can suggest that other types of attachments are available, to include this list:

- "image/jpeg"
- "video/3gpp"
- "text/x-vlocation"
- "text/vcard"
- "image/png"
- "image/tiff"
- "video/quicktime"
- "audio/x-m4a"
- "image/gif"
- "audio/amr"

The `transfer_name` contains the filename and `total_bytes` is the attachment size in bytes.

These attachments can be found in the Attachments directory alongside the database.

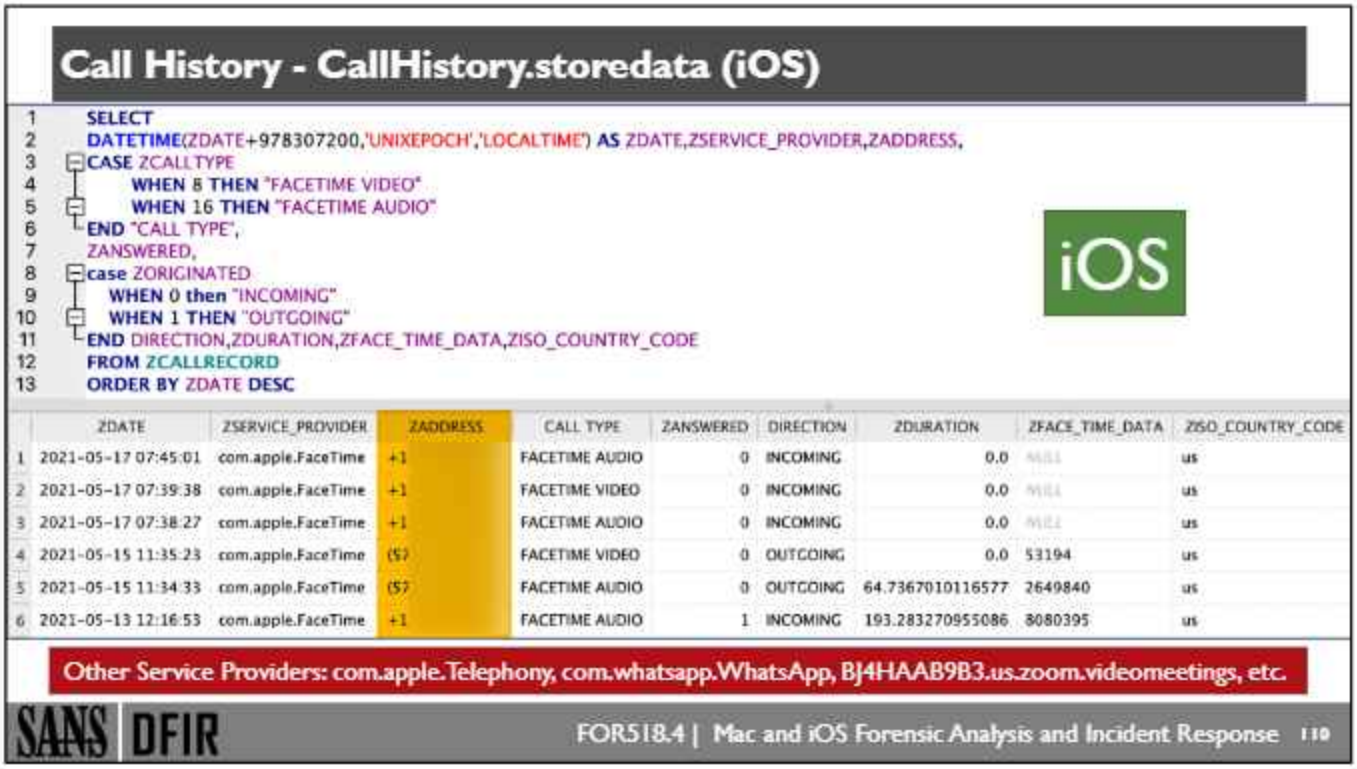
© 2022 Sarah Edwards

filename	mime_type	transfer_name	total_bytes
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
~/Library/SMS/Attachments/57/07/2F11245C-31B9-4D0E-8CAD-617E0371A453/tmp.gif	image/gif	tmp.gif	1640269
NULL	NULL	NULL	NULL
~/Library/SMS/Attachments/74/04/6D5E0AA6-A13D-4AF5-BF07-4234C02EFB7B/MOM's Organic Market.loc.vcf	text/x-vlocation	MOM's Organic Market.loc.vcf	822
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
NULL	NULL	NULL	NULL
~/Library/SMS/Attachments/ea/10/C3298AEF-C921-483D-9E74-ED7E2BB508EE/IMG_8480.heic	image/heic	IMG_8480.heic	2611406
NULL	NULL	NULL	NULL
~/Library/SMS/Attachments/8b/11/94237D39-7475-4866-9847-1582DB97BBB3/tmp.gif	image/gif	tmp.gif	1386166
NULL	NULL	NULL	NULL

© 2022 Sarah Edwards

macOS: `~/Library/Application Support/CallHistoryDB/`
iOS Physical: `/private/var/mobile/Library/CallHistoryDB/`
iOS Backup: `/mobile/Library/CallHistoryDB/`

Phone & FaceTime calls may be performed and received by different applications on macOS and iOS. The example above shows the same call history for each device.



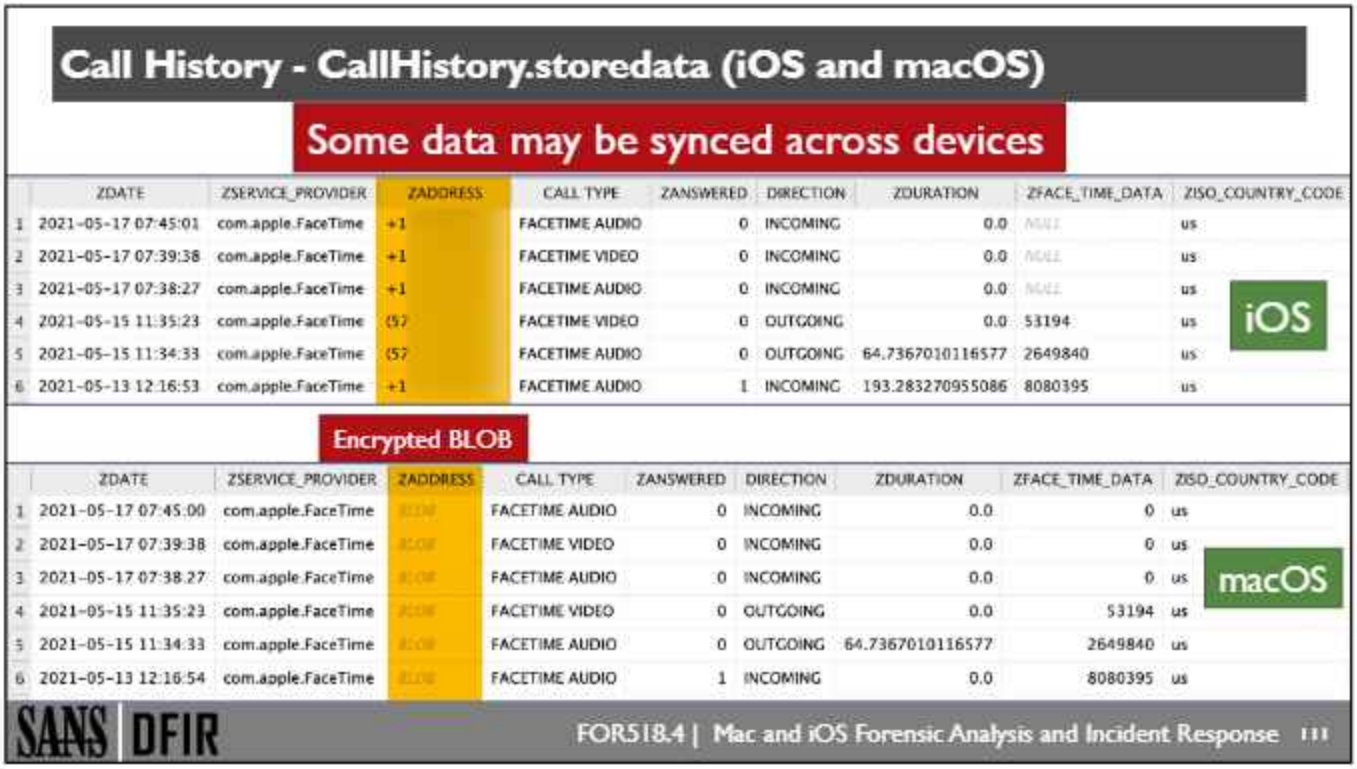
macOS: `~/Library/Application Support/CallHistoryDB/CallHistory.storedata`
iOS Physical: `/private/var/mobile/Library/CallHistoryDB/CallHistory.storedata`
iOS Backup: `/mobile/Library/CallHistoryDB/CallHistory.storedata`

The SQLite database `CallHistory.storedata` located in the directory contains the calls made. This call data is then written to the database.

Each record contains the following data:

- `ZDATE`: Mac Epoch Timestamp
- `ZADDRESS`: Contact phone number or email address (FaceTime)
- `ZANSWERED`: 0 = No, 1 = Yes
- `ZCALLTYPE`: (If duration is 0, call was missed/canceled)
  - 1: Normal Telephony Call
  - 8: FaceTime
  - 16: FaceTime Voice Call
- `ZORIGINATED`: 1 = Outgoing with this user, 0 = Incoming
- `ZDURATION`: Time in seconds of the call
- `ZISO_COUNTRY_CODE`: Country Code (can also view ZLOCATION for regular calls)
- `ZSERVICE_PROVIDER`: Application (`com.apple.Telephony`—Regular Call, may also see `com.apple.FaceTime`, `com.whatsapp.WhatsApp`, `BJ4HAAB9B3.us.zoom.videomeetings`, etc., for other applications)

© 2022 Sarah Edwards



macOS: `~/Library/Application Support/CallHistoryDB/CallHistory.storedata`
iOS Physical: `/private/var/mobile/Library/CallHistoryDB/CallHistory.storedata`
iOS Backup: `/mobile/Library/CallHistoryDB/CallHistory.storedata`

The macOS version of the database is similar, however you may find that the contact information is in an encrypted BLOB.

You may also notice that it is not necessarily a one-for-one copy of the mobile database due to various user interactions, such as taking calls on different devices.

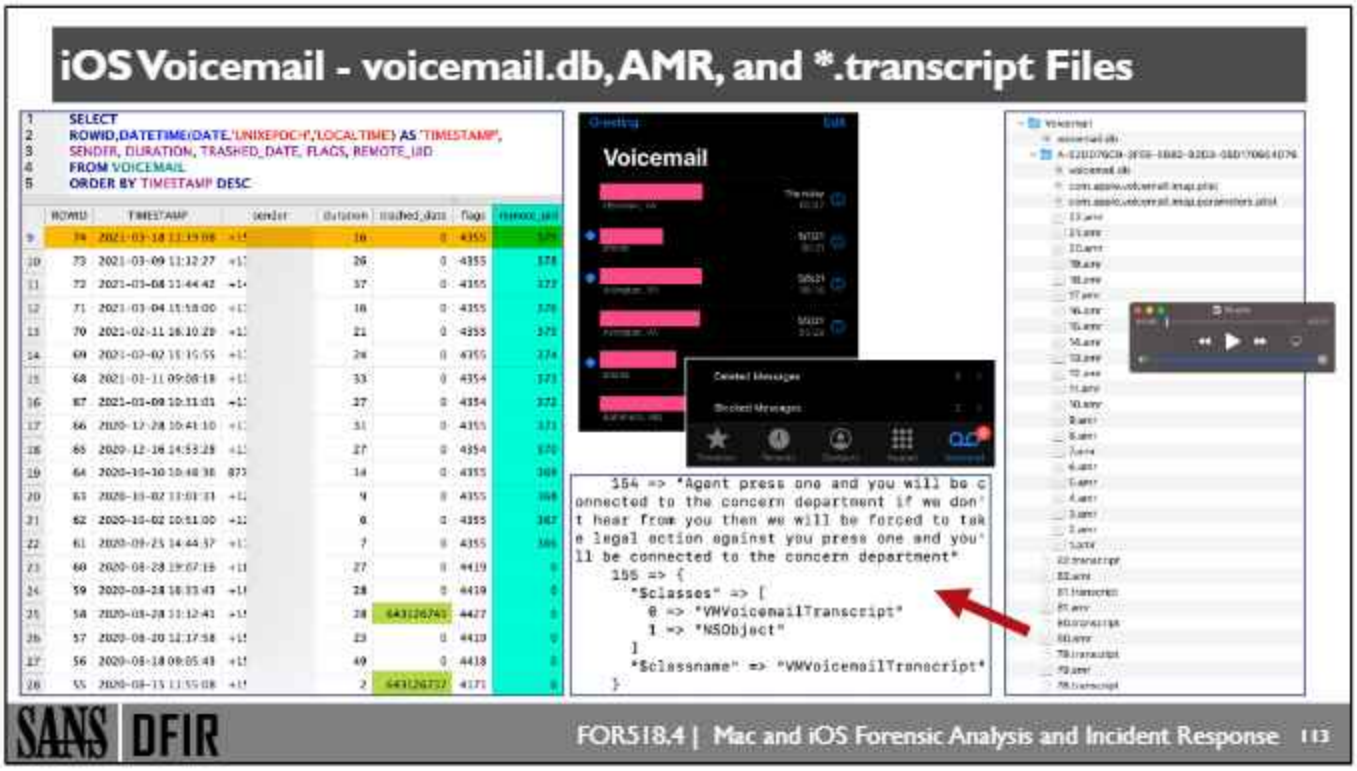
© 2022 Sarah Edwards

	ZDATE	ZSERVICE_PROVIDER	ZADDRESS	CALL TYPE	ZANSWERED	DIRECTION	ZDURATION	ZFACE_TIME_DATA	ZISO_COUNTRY_CODE
1	2021-05-17 07:45:00	com.apple.FaceTime	*BLOB*	FACETIME AUDIO	0	INCOMING	0.0	0	us
2	2021-05-17 07:39:38	com.apple.FaceTime	*BLOB*	FACETIME VIDEO	0	INCOMING	0.0	0	us
3	2021-05-17 07:38:27	com.apple.FaceTime	*BLOB*	FACETIME AUDIO	0	INCOMING	0.0	0	us
4	2021-05-15 11:35:23	com.apple.FaceTime	*BLOB*	FACETIME VIDEO	0	OUTGOING	0.0	53194	us
5	2021-05-15 11:34:33	com.apple.FaceTime	*BLOB*	FACETIME AUDIO	0	OUTGOING	64.7367010116577	2649840	us
6	2021-05-13 12:16:54	com.apple.FaceTime	*BLOB*	FACETIME AUDIO	1	INCOMING	0.0	8080395	us

```
1   SELECT
2   DATETIME(ZDATE+978307200,'UNIXEPOCH','LOCALTIME') AS ZDATE,ZSERVICE_PROVIDER,ZADDRESS,
3   CASE ZCALLTYPE
4         WHEN 8 THEN "FACETIME VIDEO"
5         WHEN 16 THEN "FACETIME AUDIO"
6   END "CALL TYPE",
7   ZANSWERED,
8   case ZORIGINATED
9      WHEN 0 then "INCOMING"
10    WHEN 1 THEN "OUTGOING"
11  END DIRECTION,ZDURATION,ZFACE_TIME_DATA,ZISO_COUNTRY_CODE
12  FROM ZCALLRECORD
13  ORDER BY ZDATE DESC
```

	ZDATE	ZSERVICE_PROVIDER	ZADDRESS	CALL TYPE	ZANSWERED	DIRECTION	ZDURATION	ZFACE_TIME_DATA	ZISO_COUNTRY_CODE
1	2021-05-17 07:45:01	com.apple.FaceTime	+1	FACETIME AUDIO	0	INCOMING	0.0	*NULL*	us
2	2021-05-17 07:39:38	com.apple.FaceTime	+1	FACETIME VIDEO	0	INCOMING	0.0	*NULL*	us
3	2021-05-17 07:38:27	com.apple.FaceTime	+1	FACETIME AUDIO	0	INCOMING	0.0	*NULL*	us
4	2021-05-15 11:35:23	com.apple.FaceTime	(57	FACETIME VIDEO	0	OUTGOING	0.0	53194	us
5	2021-05-15 11:34:33	com.apple.FaceTime	(57	FACETIME AUDIO	0	OUTGOING	64.7367010116577	2649840	us
6	2021-05-13 12:16:53	com.apple.FaceTime	+1	FACETIME AUDIO	1	INCOMING	193.283270955086	8080395	us

© 2022 Sarah Edwards



iOS FFS: `/private/var/mobile/Library/Voicemail/`
Backup: `/mobile/Library/Voicemail/`

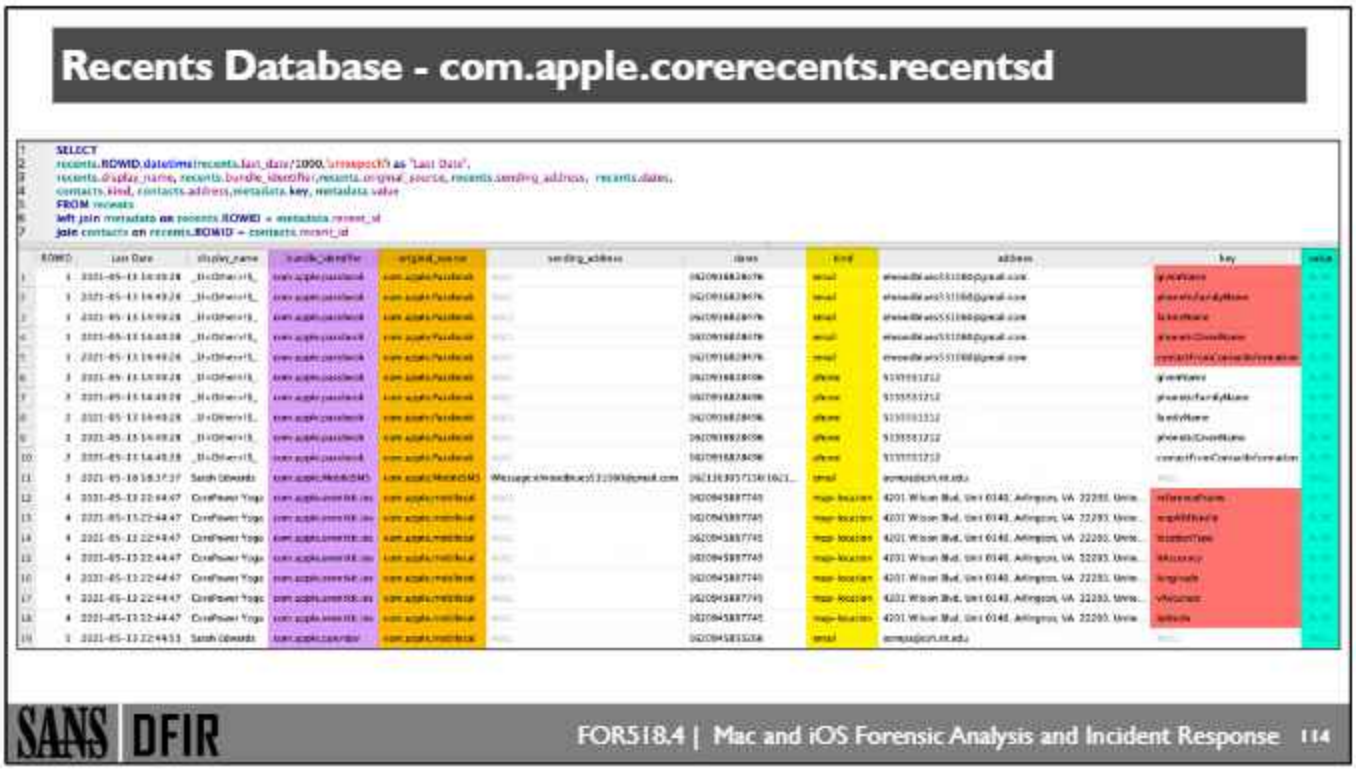
If the iOS device has the visual voicemail functionality—not all cellular carriers implement this—the device will download the voicemail audio files to the phone. The Voicemail directory will contain a `voicemail.db` SQLite database file and downloaded AMR files. Newer versions of iOS may have multiple GUID-named directories with copies of the voicemails as well as multiple `voicemail.db` databases.

The database query performed extracts the voicemail ID, timestamp, sender info, duration, if and when it was trashed, and flag information. If the voicemail is set to be deleted but is still on the device, it will show a timestamp of that date in Mac Epoch. The flags determine if the voicemail is deleted, downloaded, unheard, blocked, etc.

Each voicemail that is downloaded can be listened to using most audio players that support the Adaptive Multi-rate Audio (AMR) audio format. The voicemail filename will use the "`ROWID`" (voicemail ID) as its filename.

If the voicemail has an accompanying `*.transcript` file, you'll find an NSKeyedArchiver plist file with the transcript information.

© 2022 Sarah Edwards



macOS:
`~/Library/Containers/com.apple.corerecents.recentsd/Data/Library/Recents/`
iOS Physical: `/private/var/mobile/Library/Recents`
iOS Backup: `/mobile/Library/Recents`

The `Recents` database on macOS and iOS contains recent interactions across various applications including those for communication, such as email, phone, messages, etc.

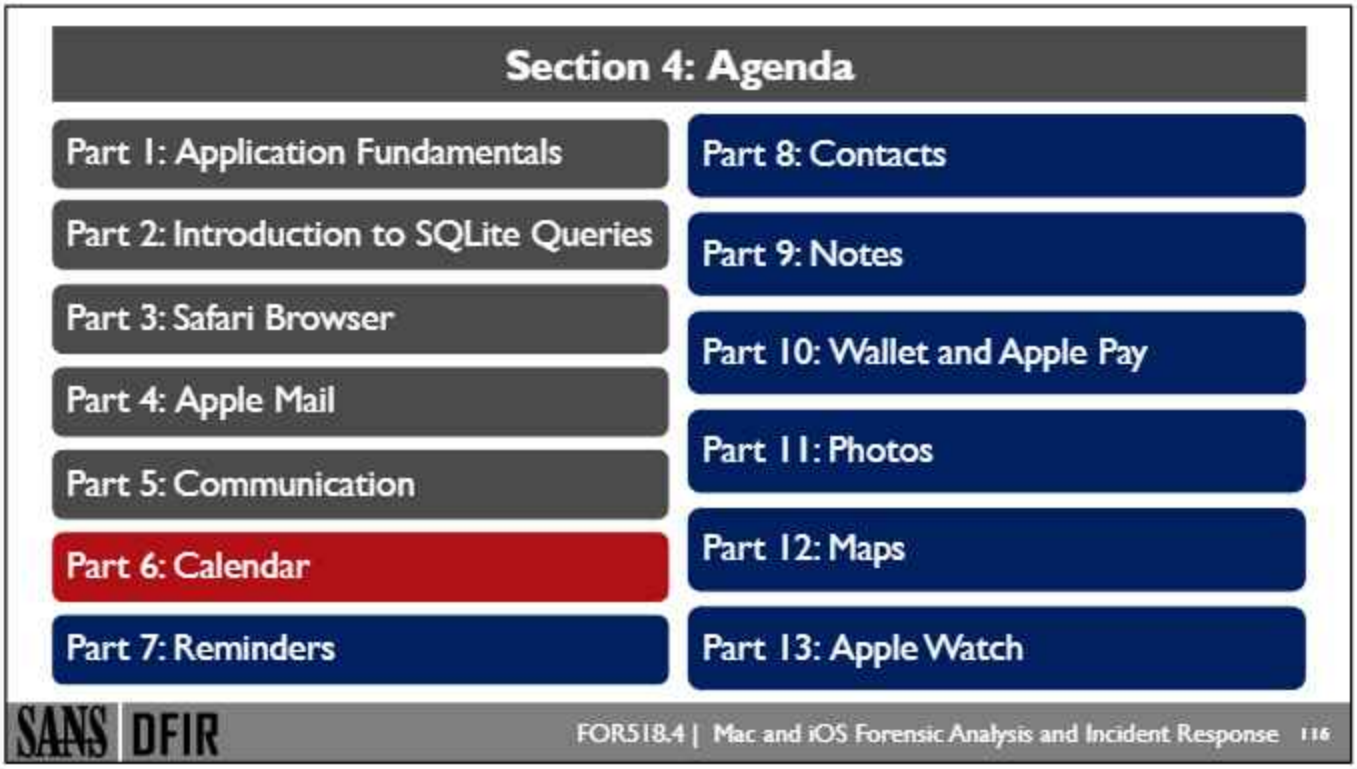
The recent items can show what applications they are associated with and came from, the contact or location, the last few timestamps a communication has occurred, as well as other various keys and values as shown above. The value BLOBs contain an embedded binary plist with different pieces of information depending on the key type.

© 2022 Sarah Edwards

```
SELECT
recents.ROWID,datetime(recents.last_date/1000,'unixepoch') as "Last Date",
recents.display_name, recents.bundle_identifier,recents.original_source, recents.sending_address,  recents.dates,
contacts.kind, contacts.address,metadata.key, metadata.value
FROM recents
left join metadata on recents.ROWID = metadata.recent_id
join contacts on recents.ROWID = contacts.recent_id
```

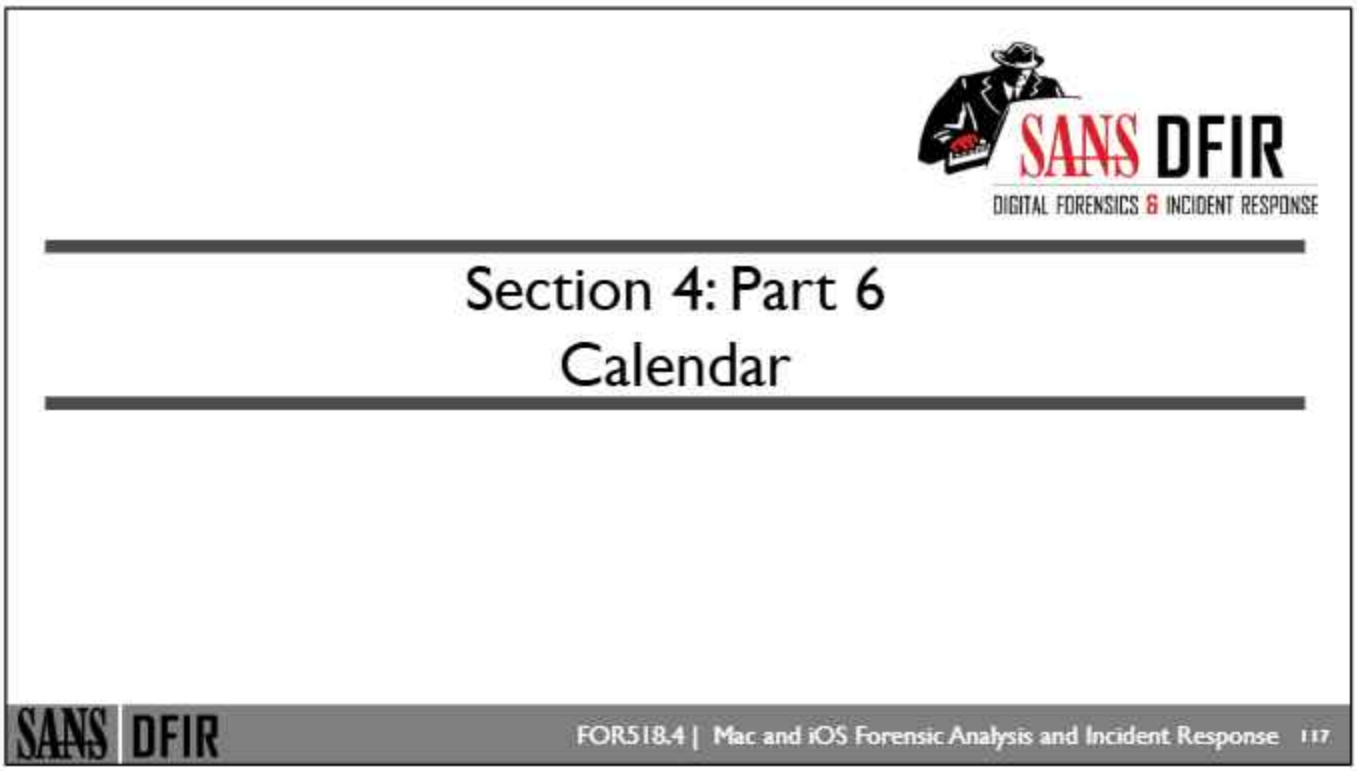
	ROWID	Last Date	display_name	bundle_identifier	original_source	sending_address	dates	kind	address	key	value
1	1	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828476	email	elwoodblues531060@gmail.com	givenName	BLOB
2	1	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828476	email	elwoodblues531060@gmail.com	phoneticFamilyName	BLOB
3	1	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828476	email	elwoodblues531060@gmail.com	familyName	BLOB
4	1	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828476	email	elwoodblues531060@gmail.com	phoneticGivenName	BLOB
5	1	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828476	email	elwoodblues531060@gmail.com	contactFromContactInformation	BLOB
6	2	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828496	phone	5155551212	givenName	BLOB
7	2	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828496	phone	5155551212	phoneticFamilyName	BLOB
8	2	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828496	phone	5155551212	familyName	BLOB
9	2	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828496	phone	5155551212	phoneticGivenName	BLOB
10	2	2021-05-13 14:40:28	_$!<Other>!$_	com.apple.passbook	com.apple.Passbook	NULL	1620916828496	phone	5155551212	contactFromContactInformation	BLOB
11	3	2021-05-18 18:37:37	Sarah Edwards	com.apple.MobileSMS	com.apple.MobileSMS	iMessage:elwoodblues531060@gmail.com	1621363057150:1621...	email	oompa@csh.rit.edu	NULL	NULL
12	4	2021-05-13 22:44:47	CorePower Yoga	com.apple.eventkit.ios	com.apple.mobilecal	NULL	1620945887745	map-location	4201 Wilson Blvd, Unit 0140, Arlington, VA 22203, Unite...	referenceFrame	BLOB
13	4	2021-05-13 22:44:47	CorePower Yoga	com.apple.eventkit.ios	com.apple.mobilecal	NULL	1620945887745	map-location	4201 Wilson Blvd, Unit 0140, Arlington, VA 22203, Unite...	mapKitHandle	BLOB
14	4	2021-05-13 22:44:47	CorePower Yoga	com.apple.eventkit.ios	com.apple.mobilecal	NULL	1620945887745	map-location	4201 Wilson Blvd, Unit 0140, Arlington, VA 22203, Unite...	locationType	BLOB
15	4	2021-05-13 22:44:47	CorePower Yoga	com.apple.eventkit.ios	com.apple.mobilecal	NULL	1620945887745	map-location	4201 Wilson Blvd, Unit 0140, Arlington, VA 22203, Unite...	hAccuracy	BLOB
16	4	2021-05-13 22:44:47	CorePower Yoga	com.apple.eventkit.ios	com.apple.mobilecal	NULL	1620945887745	map-location	4201 Wilson Blvd, Unit 0140, Arlington, VA 22203, Unite...	longitude	BLOB
17	4	2021-05-13 22:44:47	CorePower Yoga	com.apple.eventkit.ios	com.apple.mobilecal	NULL	1620945887745	map-location	4201 Wilson Blvd, Unit 0140, Arlington, VA 22203, Unite...	vAccuracy	BLOB
18	4	2021-05-13 22:44:47	CorePower Yoga	com.apple.eventkit.ios	com.apple.mobilecal	NULL	1620945887745	map-location	4201 Wilson Blvd, Unit 0140, Arlington, VA 22203, Unite...	latitude	BLOB
19	5	2021-05-13 22:44:53	Sarah Edwards	com.apple.calendar	com.apple.mobilecal	NULL	1620945893266	email	oompa@csh.rit.edu	NULL	NULL

© 2022 Sarah Edwards



This page intentionally left blank.

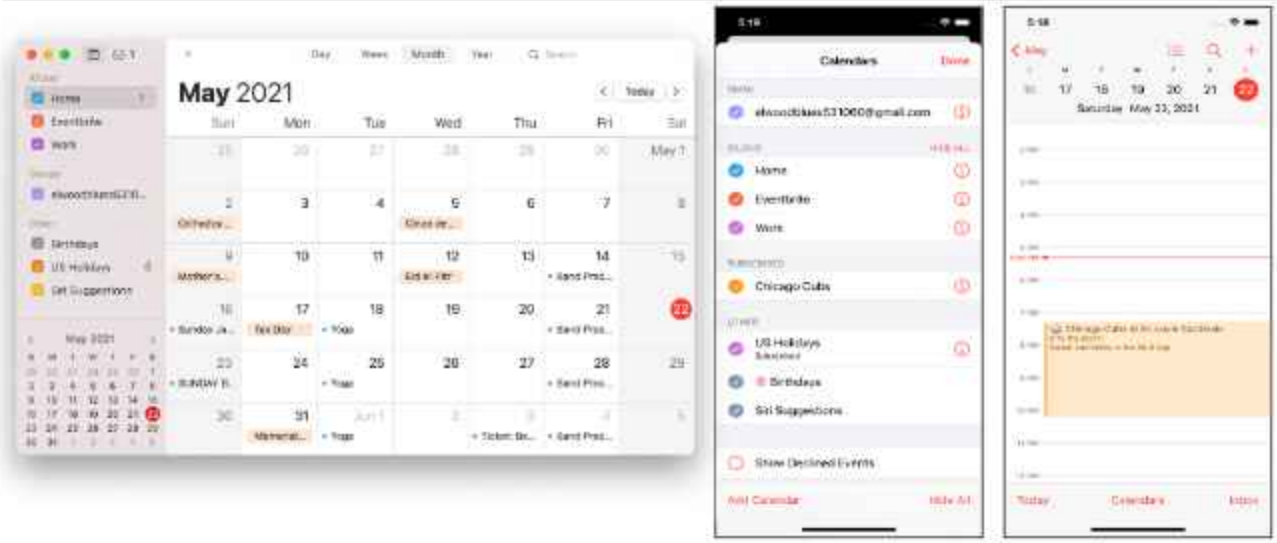
© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards

Calendar – com.apple.iCal and com.apple.mobilecal [1]
*/Library/Calendar



macOS: `~/Library/Calendars/`
iOS Physical: `/private/var/mobile/Library/Calendar/`
iOS Backup: `/mobile/Library/Calendar/`

The native calendar application is Calendar. These items can be synced from a variety of accounts, including iCloud, Google and apps such as Eventbrite. It can include both personal calendars and shared calendars.

© 2022 Sarah Edwards

Calendars – com.apple.iCal and com.apple.mobilecal [2]
*/Library/Calendar

Sources

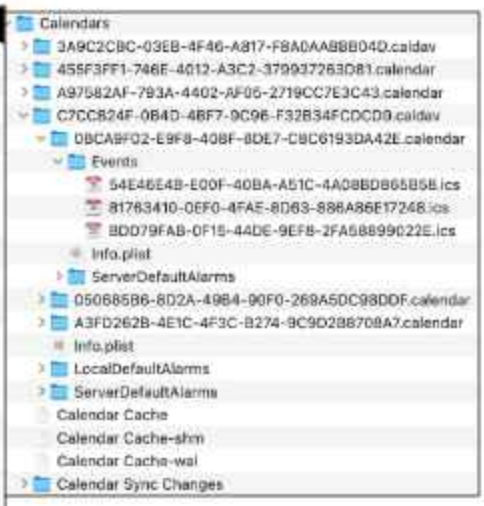
- Synced or Local
- iCloud, Apps, Email Providers, etc.

ICS Files on macOS

- Separate Events
- Synced, downloaded via email invites, shared events, etc.

Databases

- macOS: 'Calendar Cache'
- iOS: Calendar.sqlitedb



macOS: `~/Library/Calendars/`
iOS Physical: `/private/var/mobile/Library/Calendar/`
iOS Backup: `/mobile/Library/Calendar/`

On macOS, each calendar is saved as a separate directory, which is named after the calendar GUID and ends with a `.calendar` or `.caldav` extension. Each calendar directory contains an `Events` directory and an `Info.plist`.

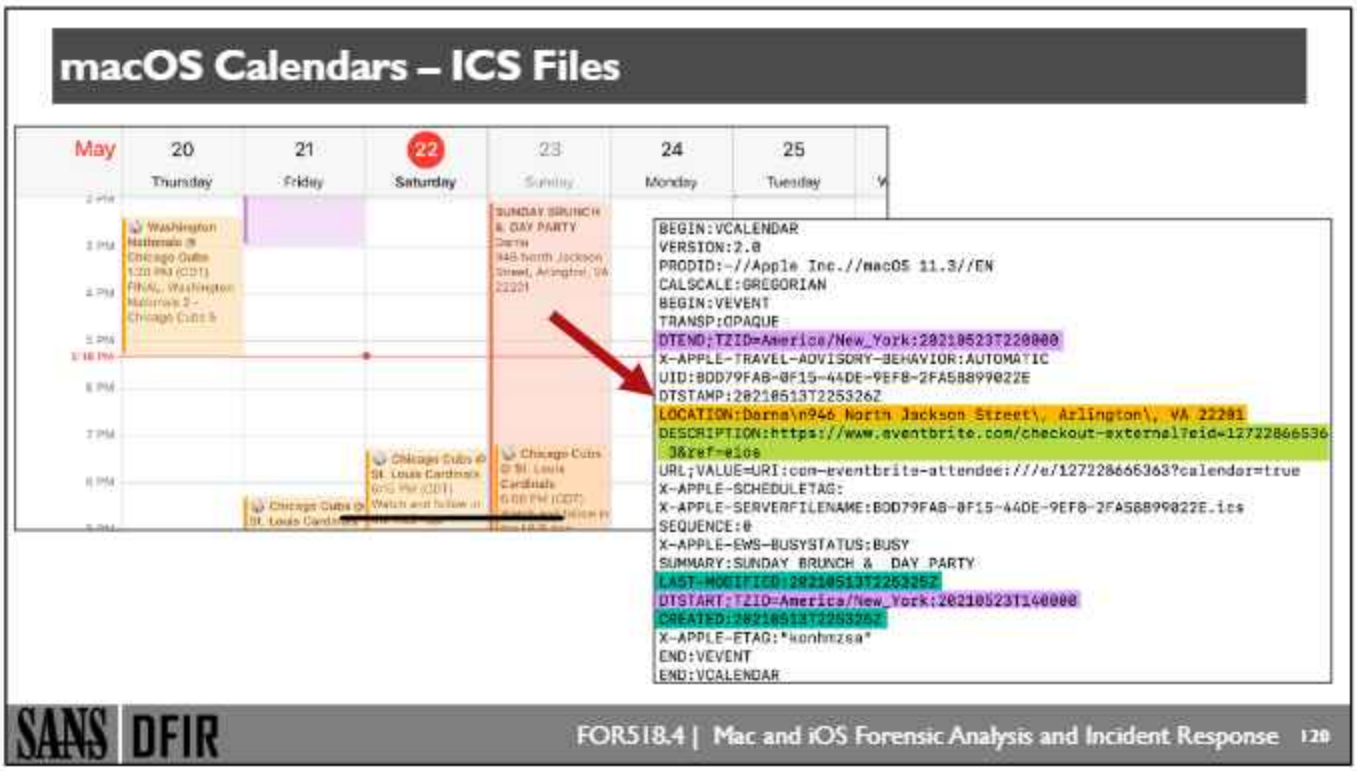
The `Events` directory contains the calendar `.ics` files. These files contain the calendar entries. Each `.ics` file contains information for a single calendar event.

The `Info.plist` contains information about the calendar, such as calendar name, GUI preferences, and other configurable items such as is it a local or synced calendar (calDAV). CalDAV is a standard that allows multiple applications, such as iCal or Google Calendar, to save the same information and keep it synced between clients.

The CalDAV `Info.plist` file may contain more information than a normal calendar `Info.plist` file.

- Account Information, such as Google Calendars or iCloud accounts
- Time Zone Information
- Last Sync Date
- Sync Settings

© 2022 Sarah Edwards



Each `.ics` file contained within a calendar is a calendar event. Shown above, it may contain:

- Create Date
- Event Start Time
- Event End Time
- Unique ID
- Time Zone Information
- Event Summary
- Locations
- Alarms
- Etc.

CalDAV calendar events may have additional information, depending on the client the event was created with.

© 2022 Sarah Edwards

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//Apple Inc.//macOS 11.3//EN
CALSCALE:GREGORIAN
BEGIN:VEVENT
TRANSP:OPAQUE
DTEND;TZID=America/New_York:20210523T220000
X-APPLE-TRAVEL-ADVISORY-BEHAVIOR:AUTOMATIC
UID:BDD79FAB-0F15-44DE-9EF8-2FA58899022E
DTSTAMP:20210513T225326Z
LOCATION:Darna\n946 North Jackson Street\, Arlington\, VA 22201
DESCRIPTION:https://www.eventbrite.com/checkout-external?eid=12722866536
 3&ref=eios
URL;VALUE=URI:com-eventbrite-attendee:///e/127228665363?calendar=true
X-APPLE-SCHEDULETAG:
X-APPLE-SERVERFILENAME:BDD79FAB-0F15-44DE-9EF8-2FA58899022E.ics
SEQUENCE:0
X-APPLE-EWS-BUSYSTATUS:BUSY
SUMMARY:SUNDAY BRUNCH &  DAY PARTY
LAST-MODIFIED:20210513T225325Z
DTSTART;TZID=America/New_York:20210523T140000
CREATED:20210513T225325Z
X-APPLE-ETAG:"konhmzsa"
END:VEVENT
END:VCALENDAR
```

macOS Calendar Cache (iOS Calendar.sqlitedb) - Calendar Events

- Much, much more information
  - Locations, Notes, Shared events, and contacts, etc.
- Reminders - macOS 10.14- and iOS 12-

SANS DFIR FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 122

The `Calendar Cache` (or `Calendar.sqlitedb` on iOS) SQLite database contains information for the Calendars. These databases have slightly different table names and have also changed column names over time but contain generally the same data.

The `ZCALENDARITEM` table contains calendar event information. Each tuple contains data like that found in the `.ics` files.

These calendar items or reminders contain a variety of data, partially shown here for brevity.

Items included are:

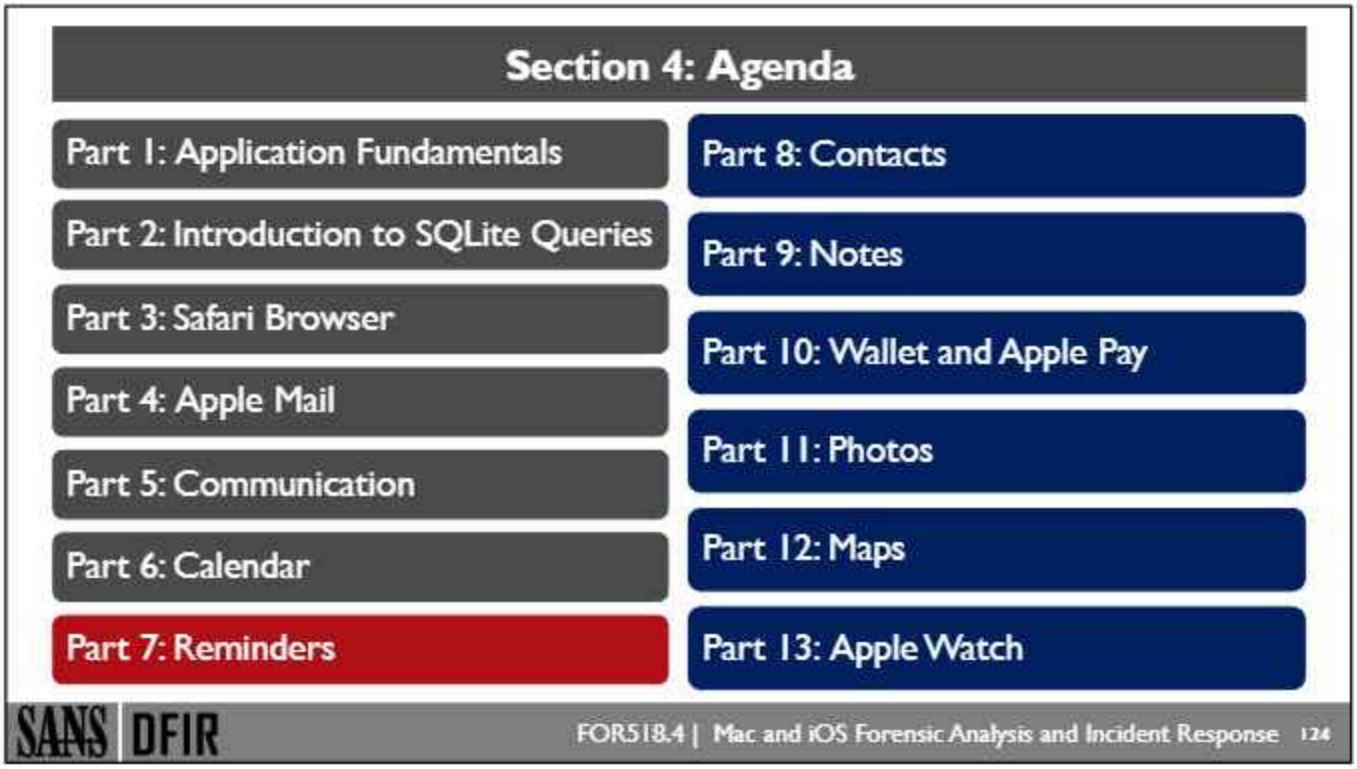
- Summary/item name
- Start and end dates
- If the calendar item is an all-day event
- What calendar it belongs to
- When the event was last modified/created
- … so much more!

© 2022 Sarah Edwards

```
SELECT
DATETIME(ZCALENDARITEM.ZCREATIONDATE+978307200,'UNIXEPOCH','LOCALTIME') AS "CREATED",DATETIME(ZCALENDARITEM.ZDATESTAMP+978307200,'UNIXEPOCH','LOCALTIME') AS "LAST MODIFIED",
DATETIME(ZCALENDARITEM.ZSTARTDATE+978307200,'UNIXEPOCH','LOCALTIME') AS "EVENT START",DATETIME(ZCALENDARITEM.ZENDDATE+978307200,'UNIXEPOCH','LOCALTIME') AS "EVENT END",
ZCALENDARITEM.ZTIMEZONE, ZNODE.ZTITLE AS "CALENDAR",
ZCALENDARITEM.ZTITLE AS "EVENT", ZCALENDARITEM.ZRECURRENCERULE, ZCALENDARITEM.ZISALLDAY,ZCALENDARITEM.ZNOTES, ZCALENDARITEM.ZURLSTRING
FROM ZCALENDARITEM
LEFT JOIN ZNODE ON ZCALENDARITEM.ZCALENDAR = ZNODE.Z_PK
```

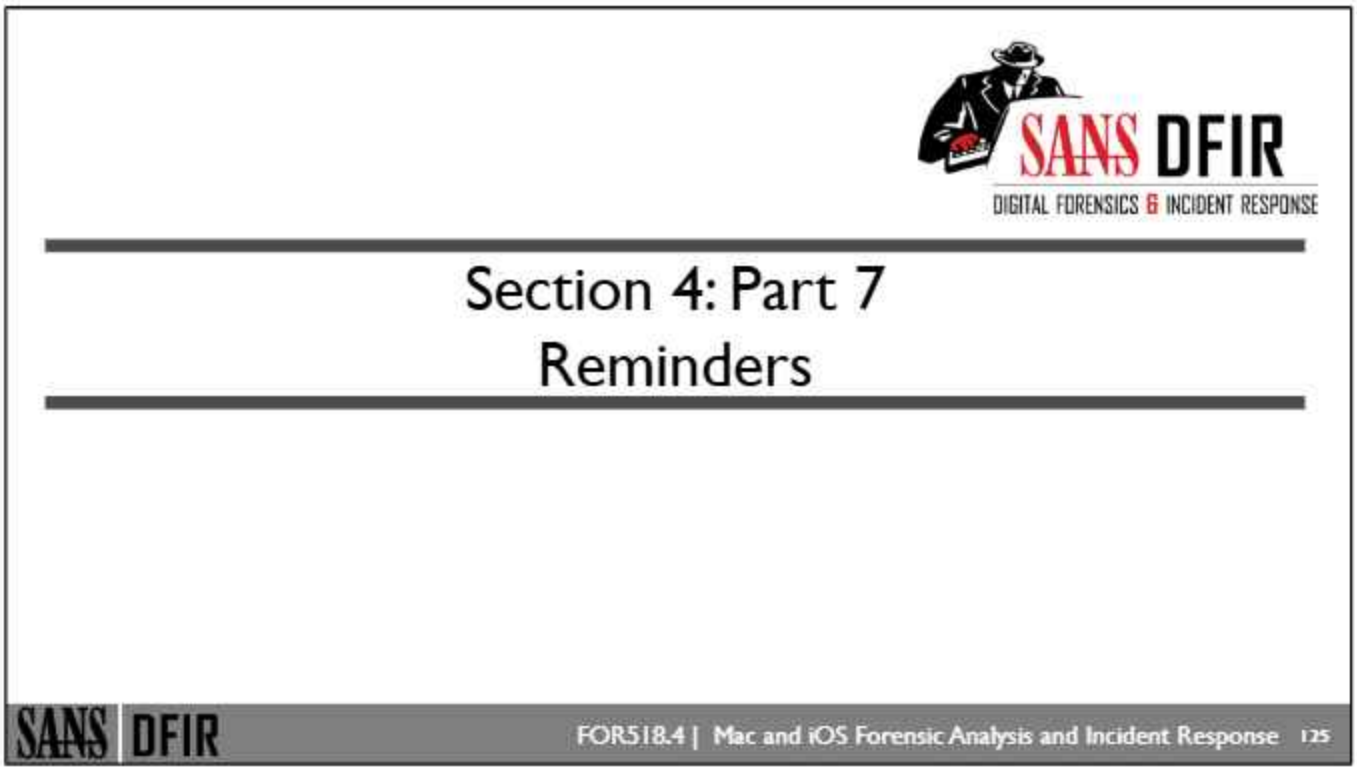
	CREATED	LAST MODIFIED	EVENT START	EVENT END	ZTIMEZONE	CALENDAR	EVENT	ZRECURRENCERULE	ZISALLDAY	ZNOTES	ZURLSTRING
1	2021-05-13 12:07:00	2021-04-01 13:21:45	2020-09-27 20:00:00	2020-09-28 20:00:00	NULL	US Holidays	Yom Kippur	NULL	1	NULL	NULL
2	2021-05-13 12:07:00	2021-04-01 13:21:43	2020-02-16 19:00:00	2020-02-17 19:00:00	NULL	US Holidays	President's Day	FREQ=YEARLY;COUNT=3;BYMONTH=2;BYDAY=3MO	1	NULL	NULL
3	2021-05-13 12:07:00	2021-04-01 13:21:43	2020-06-18 20:00:00	2020-06-19 20:00:00	NULL	US Holidays	Juneteenth	FREQ=YEARLY;COUNT=3	1	NULL	NULL
4	2021-05-13 12:07:00	2021-04-01 13:21:43	2020-07-14 20:00:00	2020-07-15 20:00:00	NULL	US Holidays	Tax Day	NULL	1	NULL	NULL
5	2021-05-13 12:07:00	2021-04-01 13:21:45	2022-08-07 20:00:00	2022-08-08 20:00:00	NULL	US Holidays	Ashura	NULL	1	The exact date of this holiday is ...	NULL
6	2021-05-13 18:39:13	2021-05-13 18:39:26	2021-05-14 12:00:00	2021-05-14 15:00:00	America/New_York	Work	Band Practice	FREQ=WEEKLY	0	NULL	
7	2021-05-13 18:39:57	2021-05-13 18:40:01	2021-07-11 20:00:00	2021-07-16 20:00:00	NULL	Home	Vacation	NULL	1	NULL	
8	2021-05-13 18:42:43	2021-05-13 18:44:53	2021-05-18 08:00:00	2021-05-18 09:00:00	America/New_York	Home	Yoga	FREQ=WEEKLY	0	NULL	https://www.corepoweryoga.com/yoga-...
9	2021-05-13 18:52:32	2021-05-13 18:52:34	2021-05-16 11:00:00	2021-05-16 14:00:00	America/New_York	Eventbrite	Sunday Jazz Brunch	NULL	0	https://www.eventbrite.com/...	com-eventbrite-attendee://ev...
10	2021-05-13 18:53:25	2021-05-13 18:53:26	2021-05-23 14:00:00	2021-05-23 22:00:00	America/New_York	Eventbrite	SUNDAY BRUNCH &...	NULL	0	https://www.eventbrite.com/...	com-eventbrite-attendee://ev...
11	2021-05-13 18:58:15	2021-05-13 18:58:18	2021-06-26 08:00:00	2021-06-26 13:00:00	America/New_York	Eventbrite	College Park Farme...	NULL	0	https://www.eventbrite.com/...	com-eventbrite-attendee://ev...
12	2021-05-13 18:59:35	2021-05-13 18:59:36	2021-06-03 19:00:00	2021-06-03 20:00:00	GMT-0400	Work	Ticket: Baking from...	NULL	0	Reservation Number: 1722377648...	message://...

© 2022 Sarah Edwards

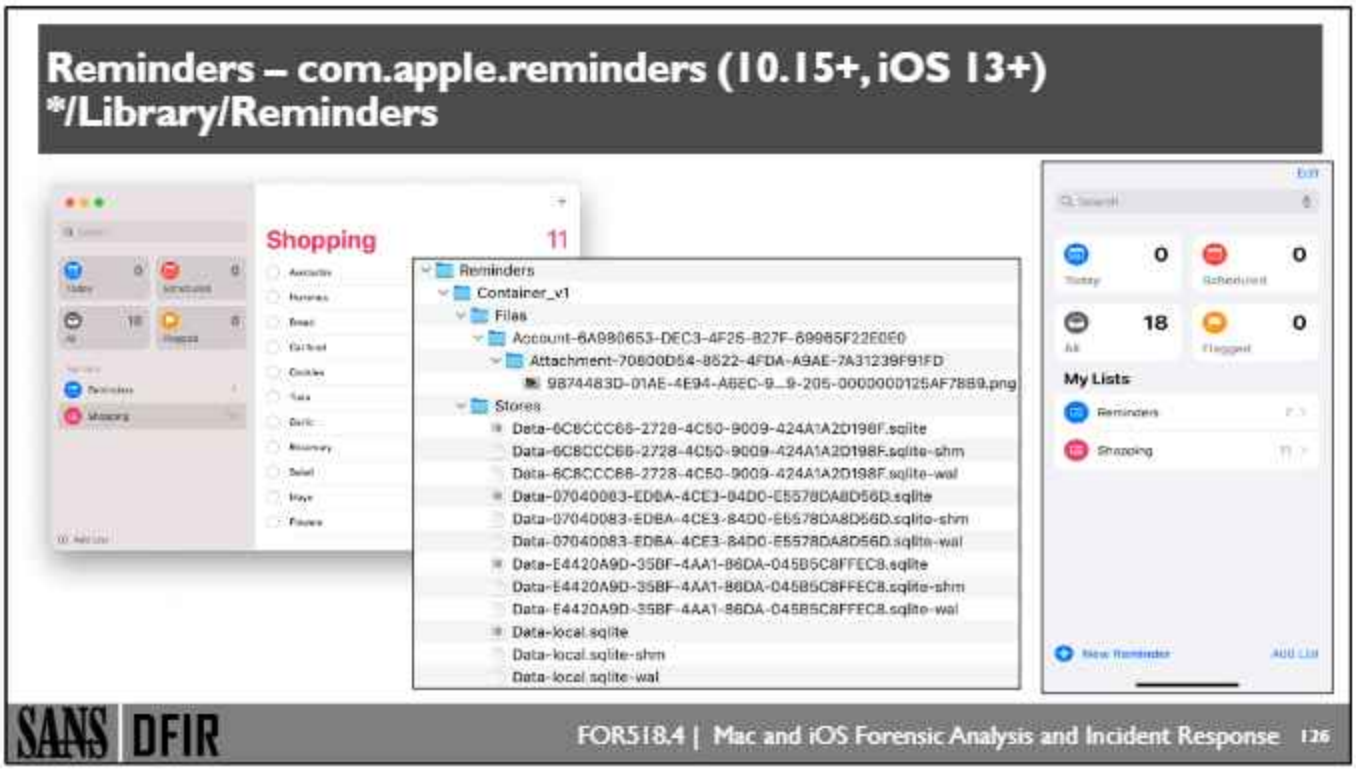


This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.

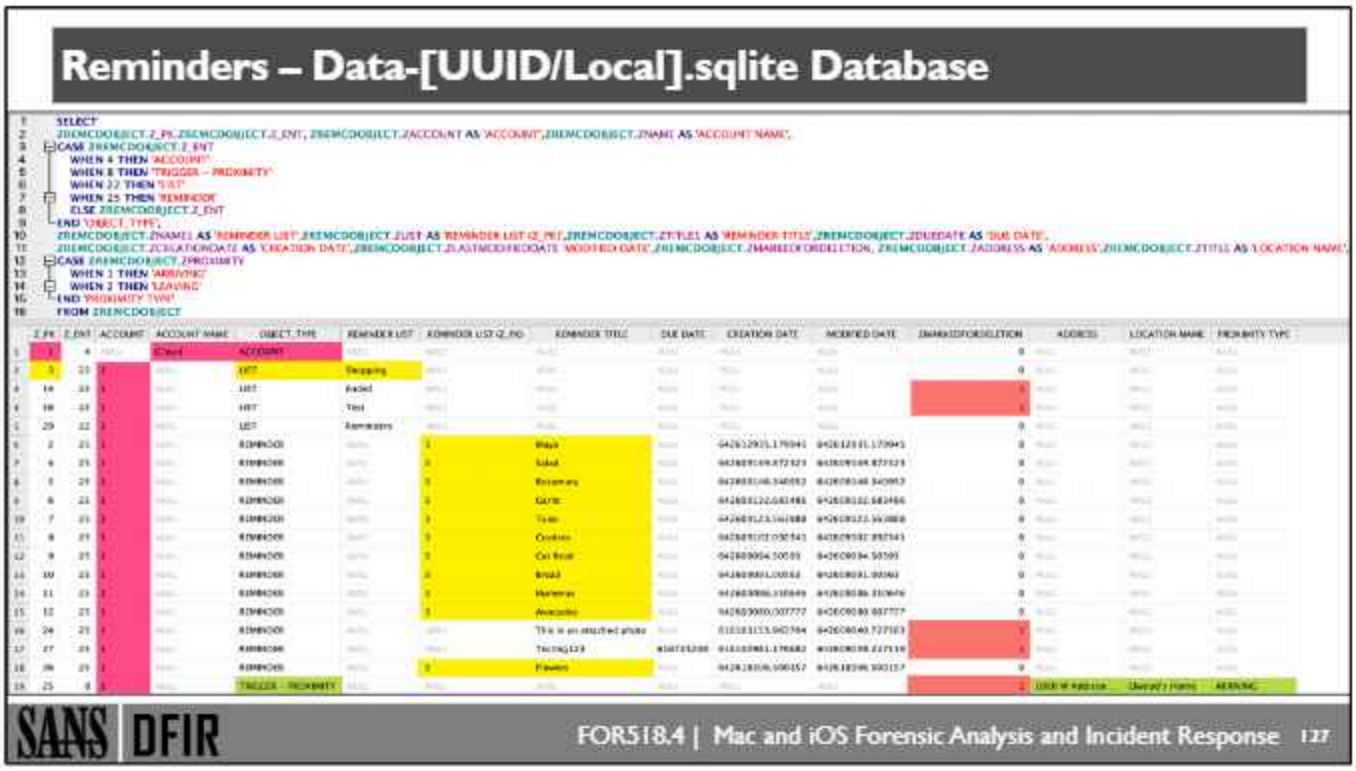


macOS: `~/Library/Reminders/`
iOS Physical: `/private/var/mobile/Library/Reminders/`
iOS Backup: `/mobile/Library/Reminders/`

In macOS 10.15 and iOS 13, the Reminders are now stored in a separate database, `Data-*.sqlite`.

Each `Data-<GUID>.sqlite` database contains the Reminders from a certain source; Local, iCloud, etc.

© 2022 Sarah Edwards



macOS: `~/Library/Reminders/`
iOS Physical: `/private/var/mobile/Library/Reminders/`
iOS Backup: `/mobile/Library/Reminders/`

Every object in this database is a different Z_ENT value. A few objects are listed in the query above. The items labeled 'REMINDER' is a Z_ENT = 25, for this version of the database. These entity numbers will change for different versions of the databases.

Each object will have a different Z_ENT (like the Notes Application) (i.e., Account, Lists, Reminders, Alarms, Triggers (Proximity, Temporal, Action), Shared Information, etc.).

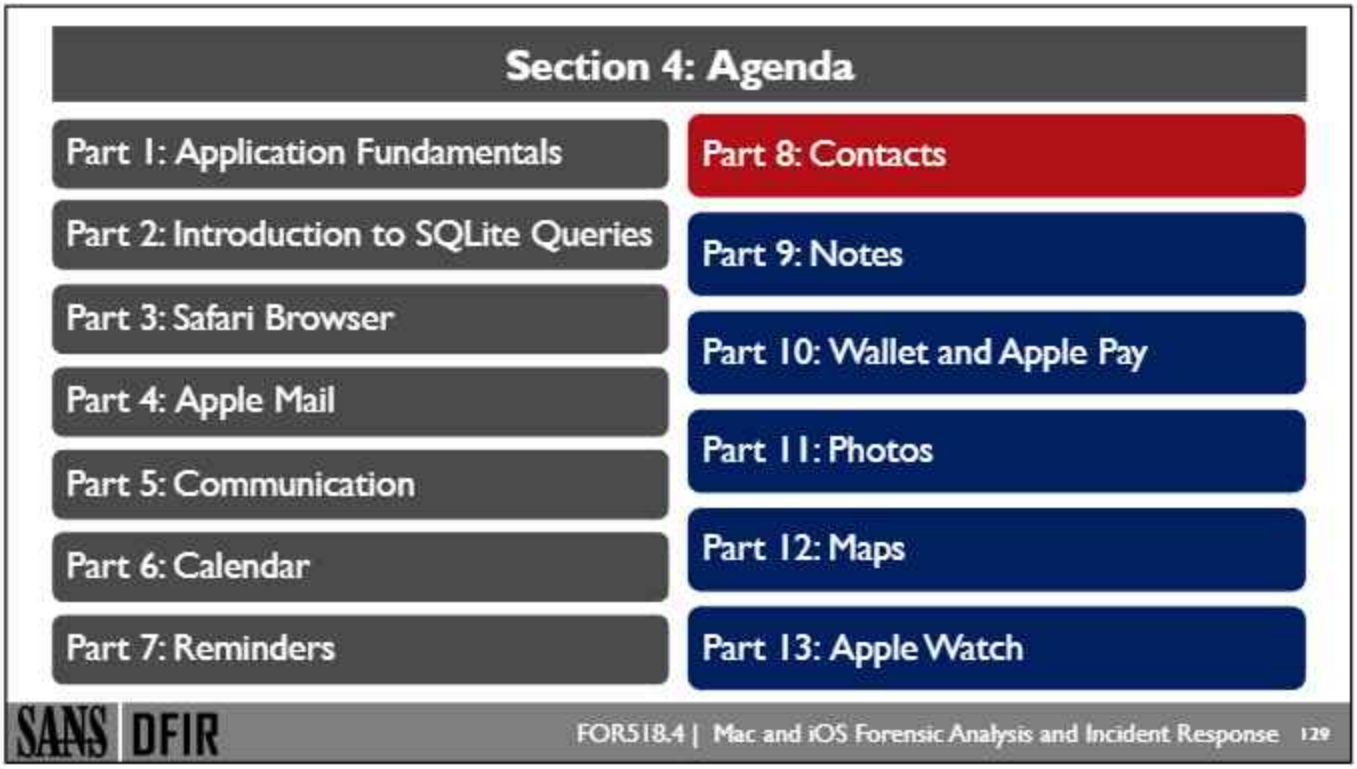
Each object will have a different set of data in the columns. A few columns in this table for the reminder object are shown in the screenshot.

© 2022 Sarah Edwards

```
SELECT
ZREMCDOBJECT.Z_PK,ZREMCDOBJECT.Z_ENT, ZREMCDOBJECT.ZACCOUNT AS 'ACCOUNT',ZREMCDOBJECT.ZNAME AS 'ACCOUNT NAME',
CASE ZREMCDOBJECT.Z_ENT
    WHEN 4 THEN 'ACCOUNT'
    WHEN 8 THEN 'TRIGGER - PROXIMITY'
    WHEN 22 THEN 'LIST'
    WHEN 25 THEN 'REMINDER'
    ELSE ZREMCDOBJECT.Z_ENT
END 'OBJECT_TYPE',
ZREMCDOBJECT.ZNAME1 AS 'REMINDER LIST',ZREMCDOBJECT.ZLIST AS 'REMINDER LIST (Z_PK)',ZREMCDOBJECT.ZTITLE1 AS 'REMINDER TITLE',ZREMCDOBJECT.ZDUEDATE AS 'DUE DATE',
ZREMCDOBJECT.ZCREATIONDATE AS 'CREATION DATE',ZREMCDOBJECT.ZLASTMODIFIEDDATE 'MODIFIED DATE',ZREMCDOBJECT.ZMARKEDFORDELETION, ZREMCDOBJECT.ZADDRESS AS 'ADDRESS',ZREMCDOBJECT.ZTITLE AS 'LOCATION NAME',
CASE ZREMCDOBJECT.ZPROXIMITY
    WHEN 1 THEN 'ARRIVING'
    WHEN 2 THEN 'LEAVING'
END 'PROXIMITY TYPE'
FROM ZREMCDOBJECT
```

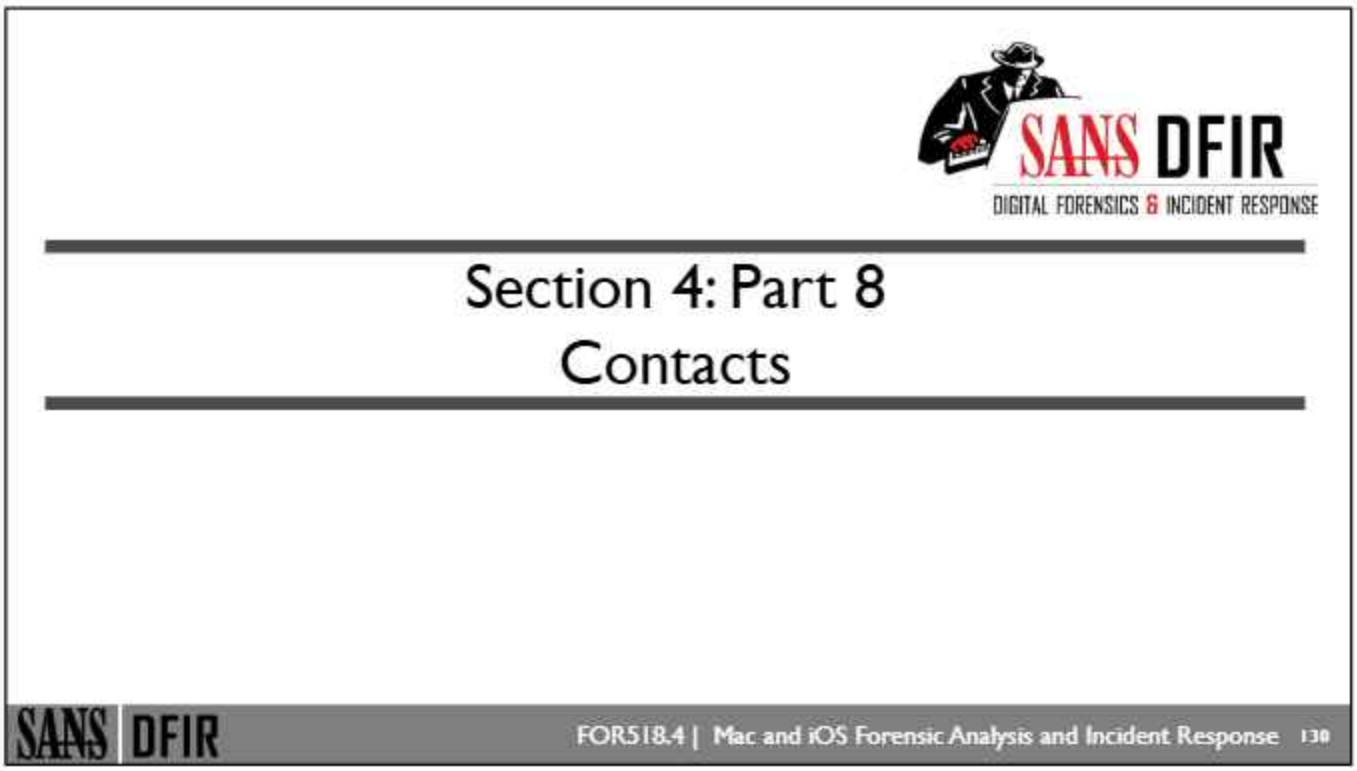
	Z_PK	Z_ENT	ACCOUNT	ACCOUNT NAME	OBJECT_TYPE	REMINDER LIST	REMINDER LIST (Z_PK)	REMINDER TITLE	DUE DATE	CREATION DATE	MODIFIED DATE	ZMARKEDFORDELETION	ADDRESS	LOCATION NAME	PROXIMITY TYPE
1	1	4	NULL	iCloud	ACCOUNT	NULL	NULL	NULL	NULL	NULL	NULL	0	NULL	NULL	NULL
2	3	22	1	NULL	LIST	Shopping	NULL	NULL	NULL	NULL	NULL	0	NULL	NULL	NULL
3	14	22	1	NULL	LIST	Ended	NULL	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL
4	16	22	1	NULL	LIST	Test	NULL	NULL	NULL	NULL	NULL	1	NULL	NULL	NULL
5	29	22	1	NULL	LIST	Reminders	NULL	NULL	NULL	NULL	NULL	0	NULL	NULL	NULL
6	2	25	1	NULL	REMINDER	NULL	3	Mayo	NULL	642612935.179945	642612935.179945	0	NULL	NULL	NULL
7	4	25	1	NULL	REMINDER	NULL	3	Salad	NULL	642609149.872323	642609149.872323	0	NULL	NULL	NULL
8	5	25	1	NULL	REMINDER	NULL	3	Rosemary	NULL	642609146.340952	642609146.340952	0	NULL	NULL	NULL
9	6	25	1	NULL	REMINDER	NULL	3	Garlic	NULL	642609132.683486	642609132.683486	0	NULL	NULL	NULL
10	7	25	1	NULL	REMINDER	NULL	3	Tuna	NULL	642609123.563888	642609123.563888	0	NULL	NULL	NULL
11	8	25	1	NULL	REMINDER	NULL	3	Cookies	NULL	642609102.092341	642609102.092341	0	NULL	NULL	NULL
12	9	25	1	NULL	REMINDER	NULL	3	Cat food	NULL	642609094.50593	642609094.50593	0	NULL	NULL	NULL
13	10	25	1	NULL	REMINDER	NULL	3	Bread	NULL	642609091.00563	642609091.00563	0	NULL	NULL	NULL
14	11	25	1	NULL	REMINDER	NULL	3	Hummus	NULL	642609088.350646	642609088.350646	0	NULL	NULL	NULL
15	12	25	1	NULL	REMINDER	NULL	3	Avocados	NULL	642609080.007777	642609080.007777	0	NULL	NULL	NULL
16	24	25	1	NULL	REMINDER	NULL	NULL	This is an attached photo	NULL	616103153.963704	642609040.727563	1	NULL	NULL	NULL
17	27	25	1	NULL	REMINDER	NULL	NULL	Testing123	616723200	616102981.126682	642609039.227519	1	NULL	NULL	NULL
18	36	25	1	NULL	REMINDER	NULL	3	Flowers	NULL	642618306.500157	642618306.500157	0	NULL	NULL	NULL
19	25	8	1	NULL	TRIGGER - PROXIMITY	NULL	NULL	NULL	NULL	NULL	NULL	1	1060 W Addison	Elwood's Home	ARRIVING

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards

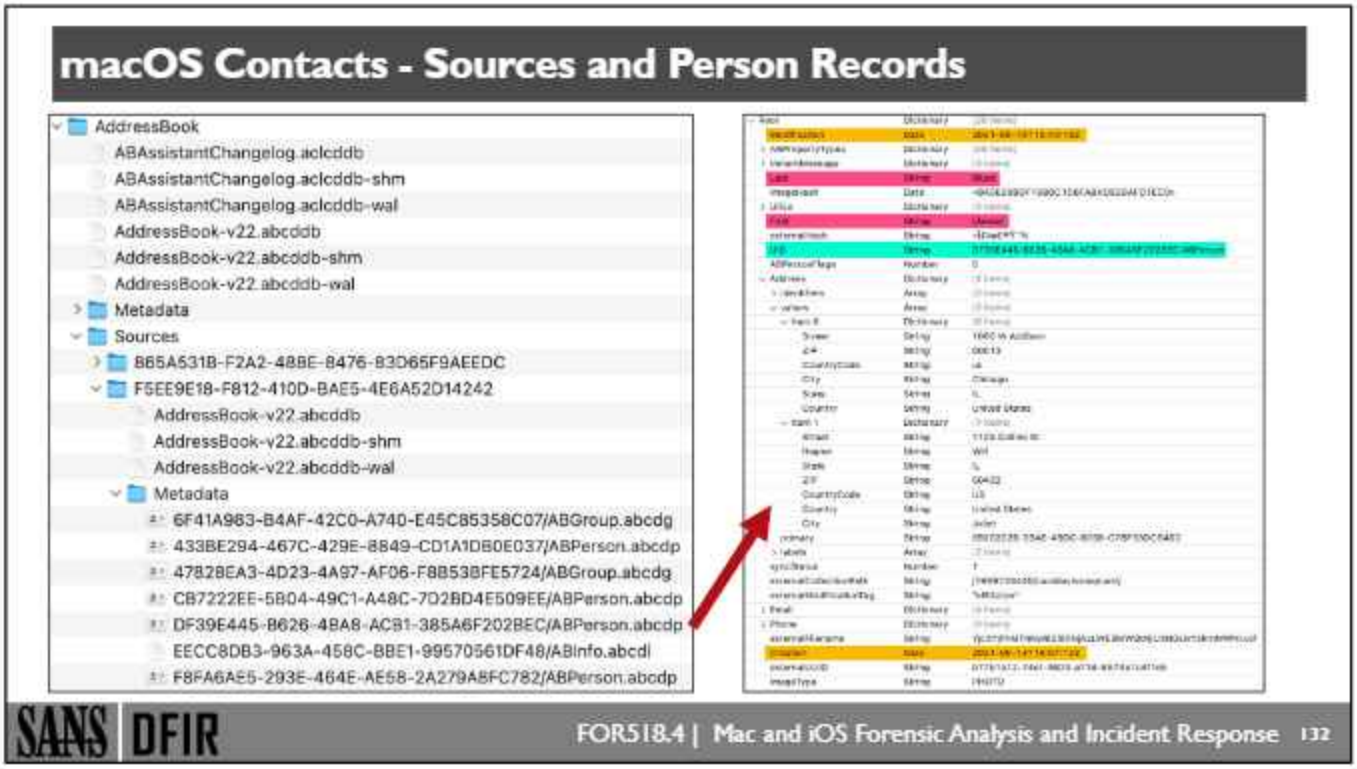


This page intentionally left blank.

© 2022 Sarah Edwards

macOS: `~/Library/Application Support/AddressBook/`
iOS: `[/private/var/]mobile/Library/AddressBook/`

The Address Book, or Contacts application, holds the user's contact information. This database gets populated by the user but also from other sources if the user associates their email and social media accounts with the operating system such as iCloud and Google above.



macOS: `~/Library/Application Support/AddressBook/`
iOS: `[/private/var/]mobile/Library/AddressBook/`

Each Address Book database under Sources will have contact information separated out. One could be the contacts for Google, iCloud—or another source. These sources can be determined by matching the GUID up in the `Accounts` database.

The Metadata directory contains a file for each person, group, or subscription. Each file is named accordingly: "`p`" for person, "`g`" for group, and "`s`" for subscription. Each file is a binary property list containing the information for that particular Address Book entry.

Much of the same information found in the Metadata directory may be found in the SQLite database `AddressBook-v22.abcddb`.

To view the raw Address Book records with Xcode, you will need to rename them as `.plist`; otherwise, they will show up in a view similar to what is found in the Contacts application.

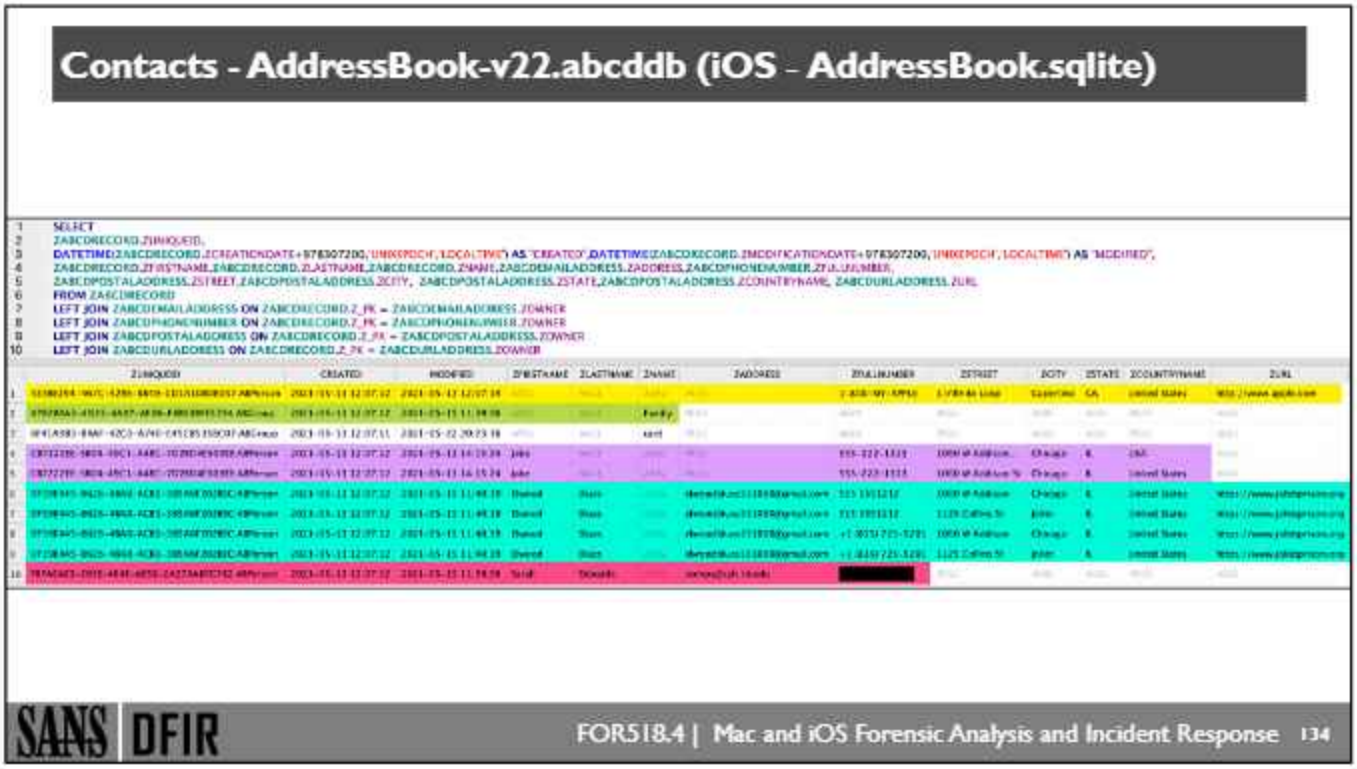
Each `Person` record has its own GUID in the `UID` key. This GUID may be used in the `Group` record if the `Person` record is part of a group. The `Creation` key contains the date the record was created, while the `Modification` date contains the date when the contact was last modified. Each `Person` record contains the phone, address, email, or other information the user has supplied. If the contact has multiple entries under the "`values`" key, such as multiple phone numbers, the "`primary`" key will determine which of these is listed as the primary contact number. If you are looking for a person of interest, it is highly recommended to search not just for their names, but also for their Contacts UID. This can pinpoint information in many other applications, files, logs, databases, etc.

A Group record contains all the GUIDs of Personal records that are a part of that group. Each group also has its own GUID that can be referenced by various applications.

© 2022 Sarah Edwards

Root	Dictionary	(20 items)
Modification	Date	2021-05-15T15:40:19Z
ABPropertyTypes	Dictionary	(50 items)
InstantMessage	Dictionary	(4 items)
Last	String	Blues
imageHash	Data	<9A5E989DF19B0C1C6FABAD52BAFD1EC0>
URLs	Dictionary	(4 items)
First	String	Elwood
externalHash	String	+İ£¤øE®Ý˜%
UID	String	DF39E445-B626-4BA8-ACB1-385A6F202BEC:ABPerson
ABPersonFlags	Number	0
Address	Dictionary	(4 items)
identifiers	Array	(2 items)
values	Array	(2 items)
Item 0	Dictionary	(6 items)
Street	String	1060 W Addison
ZIP	String	60613
CountryCode	String	us
City	String	Chicago
State	String	IL
Country	String	United States
Item 1	Dictionary	(7 items)
Street	String	1125 Collins St
Region	String	Will
State	String	IL
ZIP	String	60432
CountryCode	String	US
Country	String	United States
City	String	Joliet
primary	String	69202026-39AE-49DC-8058-C7BF33DCB40C
labels	Array	(2 items)
syncStatus	Number	1
externalCollectionPath	String	/1989733445/carddavhome/card/
externalModificationTag	String	"kl92zlzw"
Email	Dictionary	(4 items)
Phone	Dictionary	(4 items)
externalFilename	String	Yjc3YjFhMTItNzRlZi00NjAzLWE3MWQtNjU3NGUxYzRmMWNi.vcf
Creation	Date	2021-05-13T16:07:12Z
externalUUID	String	b77b1a12-74ef-4603-a71d-6574e1c4f1cb
imageType	String	PHOTO

© 2022 Sarah Edwards



macOS: `~/Library/Application Support/AddressBook/AddressBook-v22.abcddb`
iOS Physical: `/private/var//mobile/Library/AddressBook/AddressBook.sqlite`
iOS Backup: `/mobile/Library/AddressBook/AddressBook.sqlite`

The Contacts application stores its data in SQLite databases, `AddressBook.sqlite` on iOS and `AddressBook-v22.abcddb` on macOS. The table and column names from iOS and macOS will be slightly different but will contain the same type of information.

The databases contain contacts information including names, emails, addresses, phone numbers, social media accounts, etc.

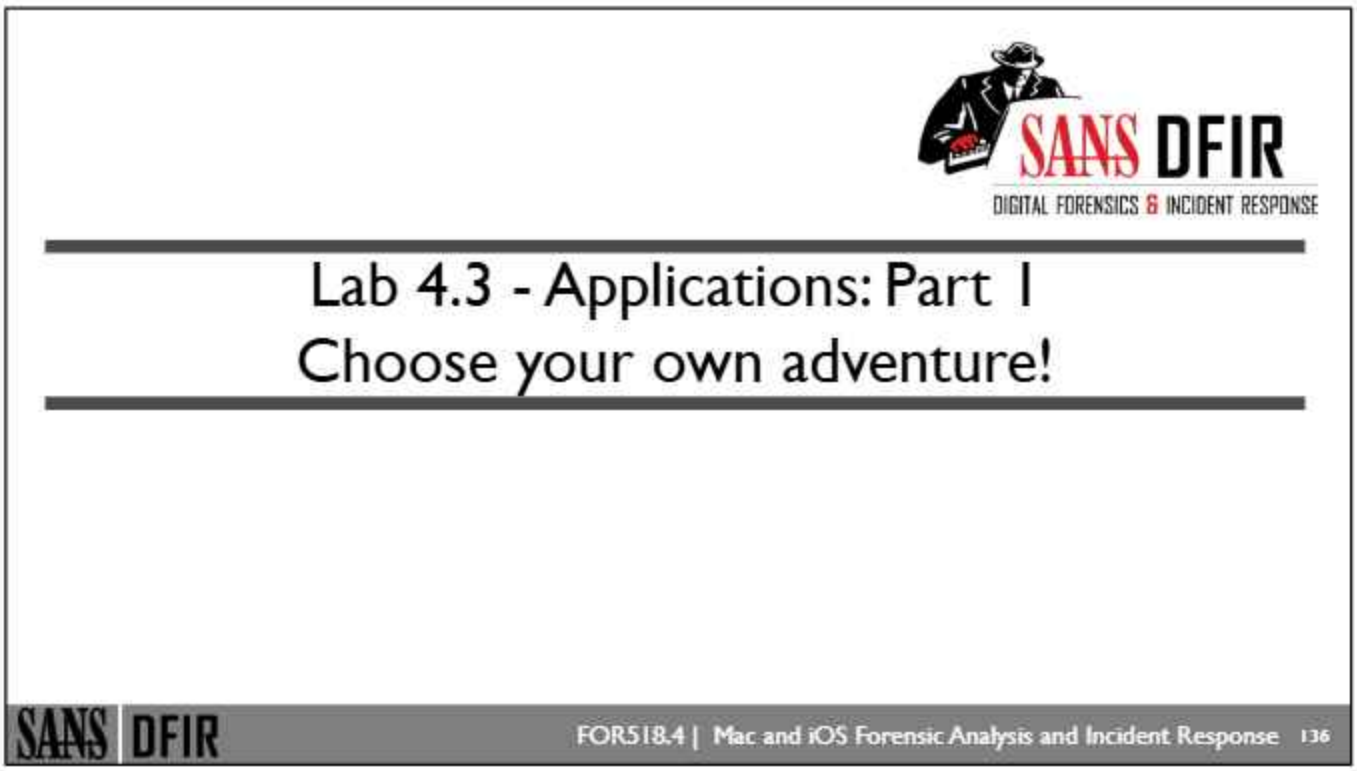
The `ABCDRECORD` table (`ABPerson` on iOS) stores the contact name, organization, job title, etc. This table also stores the contact creation and modification dates. There are many more tables that hold additional information. In the example, the email addresses, phone numbers, URLs, and addresses were extracted from other tables.

© 2022 Sarah Edwards

```sql
SELECT
ZABCDRECORD.ZUNIQUEID,
DATETIME(ZABCDRECORD.ZCREATIONDATE+978307200,'UNIXEPOCH','LOCALTIME') AS "CREATED",DATETIME(ZABCDRECORD.ZMODIFICATIONDATE+978307200,'UNIXEPOCH','LOCALTIME') AS "MODIFIED",
ZABCDRECORD.ZFIRSTNAME,ZABCDRECORD.ZLASTNAME,ZABCDRECORD.ZNAME,ZABCDEMAILADDRESS.ZADDRESS,ZABCDPHONENUMBER.ZFULLNUMBER,
ZABCDPOSTALADDRESS.ZSTREET,ZABCDPOSTALADDRESS.ZCITY,  ZABCDPOSTALADDRESS.ZSTATE,ZABCDPOSTALADDRESS.ZCOUNTRYNAME, ZABCDURLADDRESS.ZURL
FROM ZABCDRECORD
LEFT JOIN ZABCDEMAILADDRESS ON ZABCDRECORD.Z_PK = ZABCDEMAILADDRESS.ZOWNER
LEFT JOIN ZABCDPHONENUMBER ON ZABCDRECORD.Z_PK = ZABCDPHONENUMBER.ZOWNER
LEFT JOIN ZABCDPOSTALADDRESS ON ZABCDRECORD.Z_PK = ZABCDPOSTALADDRESS.ZOWNER
LEFT JOIN ZABCDURLADDRESS ON ZABCDRECORD.Z_PK = ZABCDURLADDRESS.ZOWNER
```

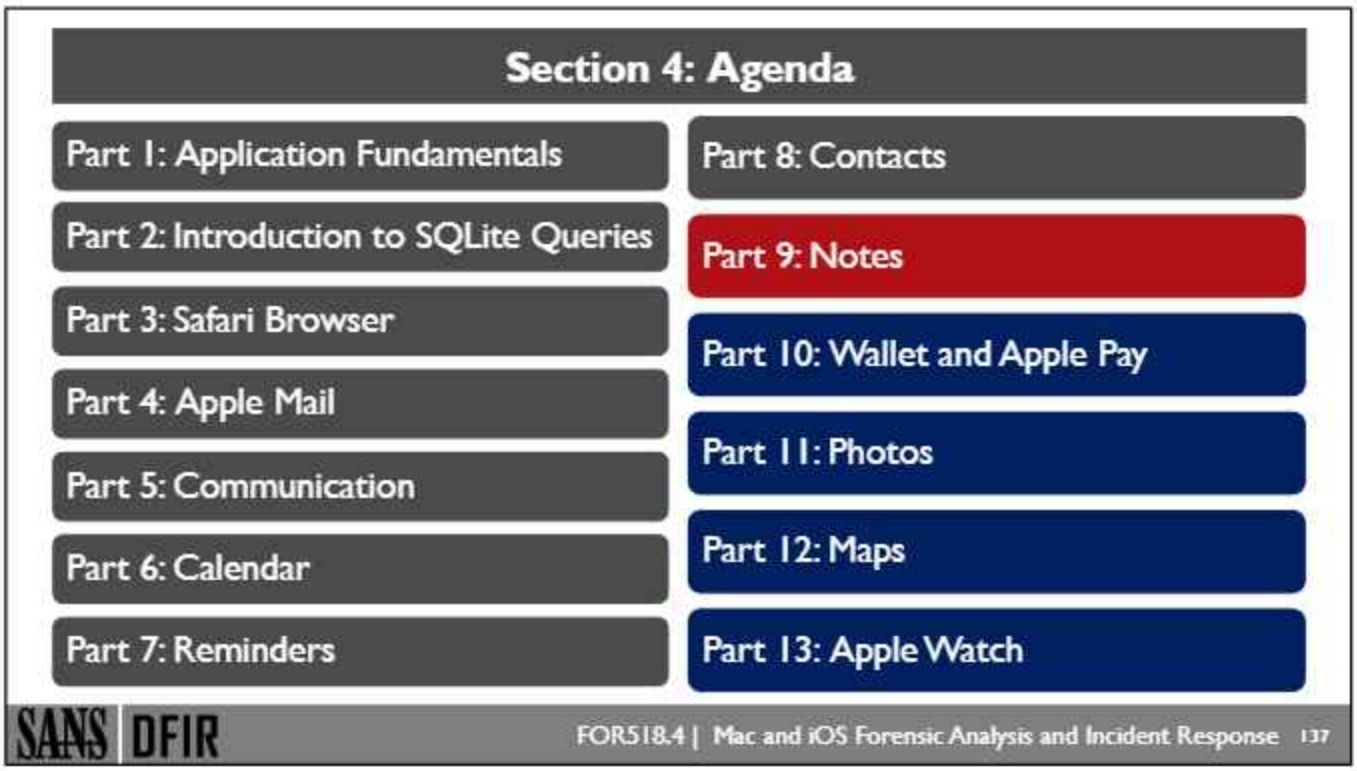
	ZUNIQUEID	CREATED	MODIFIED	ZFIRSTNAME	ZLASTNAME	ZNAME	ZADDRESS	ZFULLNUMBER	ZSTREET	ZCITY	ZSTATE	ZCOUNTRYNAME	ZURL
1	43380294-467C-4290-8849-CD1A1DB00037:ABPerson	2021-05-13 12:07:12	2021-05-13 12:07:14	NULL	NULL	NULL	NULL	1-800-MY-APPLE	1 Infinite Loop	Cupertino	CA	United States	http://www.apple.com
2	478286A1-4023-4A97-AF05-F8E538FE5724:ABGroup	2021-05-13 12:07:12	2021-05-15 11:39:06	NULL	NULL	Family	NULL	NULL	NULL	NULL	NULL	NULL	NULL
3	6F41A983-84AF-42C0-A740-E45C85358C07:ABGroup	2021-05-13 12:07:11	2021-05-22 20:23:36	NULL	NULL	card	NULL	NULL	NULL	NULL	NULL	NULL	NULL
4	C87222EE-5804-49C1-A48C-7D28D4E509EE:ABPerson	2021-05-13 12:07:12	2021-05-13 14:15:24	Jake	NULL	NULL	NULL	555-222-1313	1060 W Addison...	Chicago	IL	USA	NULL
5	C87222EE-5804-49C1-A48C-7D28D4E509EE:ABPerson	2021-05-13 12:07:12	2021-05-13 14:15:24	Jake	NULL	NULL	NULL	555-222-1313	1060 W Addison St	Chicago	IL	United States	NULL
6	DF19E445-8626-48A8-ACB1-385A0F2028EC:ABPerson	2021-05-13 12:07:12	2021-05-15 11:40:19	Elwood	Blues	NULL	elwoodblues31060@gmail.com	555 5551212	1060 W Addison	Chicago	IL	United States	https://www.jolietprison.org
7	DF19E445-8626-48A8-ACB1-385A0F2028EC:ABPerson	2021-05-13 12:07:12	2021-05-15 11:40:19	Elwood	Blues	NULL	elwoodblues31060@gmail.com	555 5551212	1125 Collins St	Joliet	IL	United States	https://www.jolietprison.org
8	DF19E445-8626-48A8-ACB1-385A0F2028EC:ABPerson	2021-05-13 12:07:12	2021-05-15 11:40:19	Elwood	Blues	NULL	elwoodblues31060@gmail.com	+1 (815) 723-5201	1060 W Addison	Chicago	IL	United States	https://www.jolietprison.org
9	DF19E445-8626-48A8-ACB1-385A0F2028EC:ABPerson	2021-05-13 12:07:12	2021-05-15 11:40:19	Elwood	Blues	NULL	elwoodblues31060@gmail.com	+1 (815) 723-5201	1125 Collins St	Joliet	IL	United States	https://www.jolietprison.org
10	FBFA6AE5-293E-4646-AE58-2A273A8FC782:ABPerson	2021-05-13 12:07:12	2021-05-15 11:39:06	Sarah	Edwards	NULL	oompa@csh.rit.edu	[redacted]	NULL	NULL	NULL	NULL	NULL

© 2022 Sarah Edwards



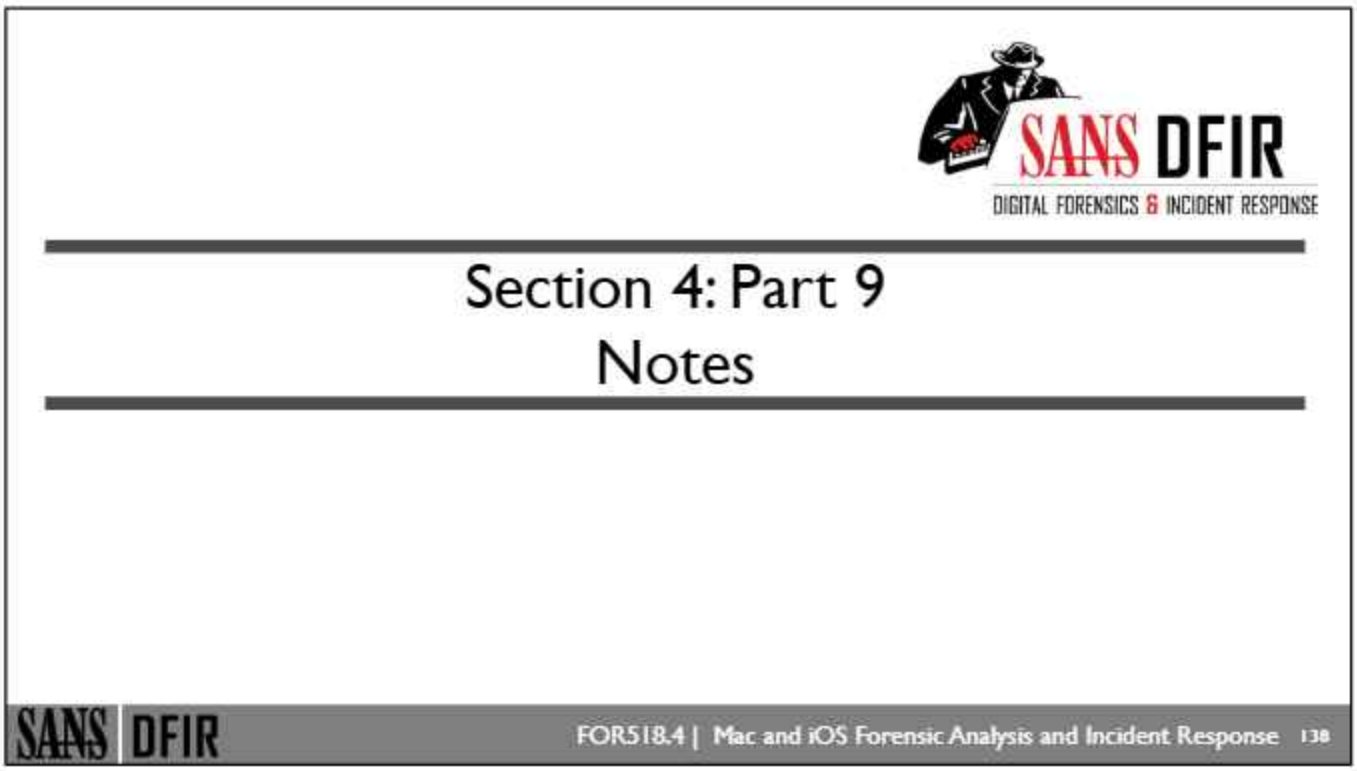
This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards

iOS FFS:

- `/private/var/mobile/Containers/Shared/AppGroup/<GUID>`
- `/private/var/mobile/Library/Notes (Legacy)`

iOS Backup:

- `/mobile/Applications/com.apple.notes`
- `/mobile/Applications/Notes (Legacy)`

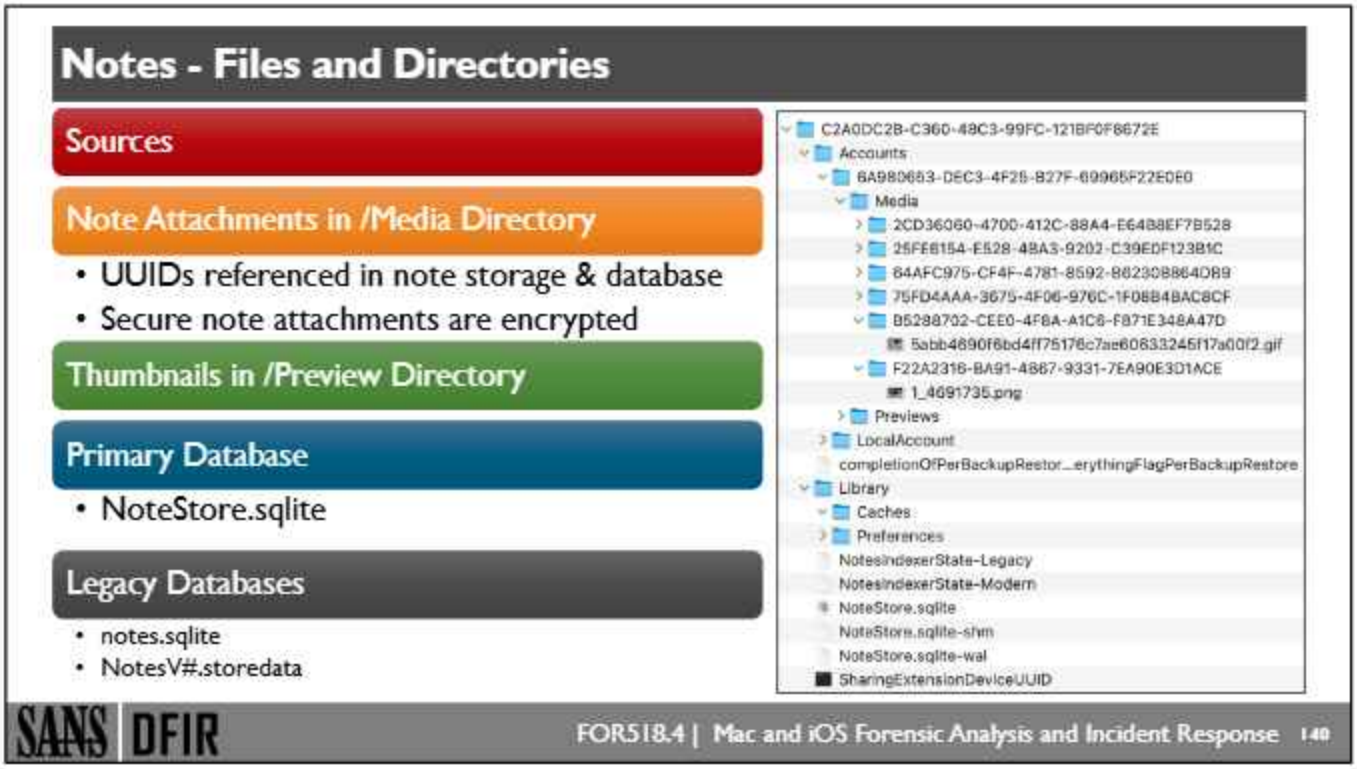
macOS:

- `~/Library/Group Containers/group.com.apple.notes/`
- `~/Library/Containers/com.apple.Notes/ (Legacy)`

The Notes application allows a user to create notes of various types on macOS, iCloud.com, or on iOS devices. If the user selects one of the syncing services, these notes will be pushed to all devices associated with these accounts.

The iCloud service will sync all iCloud notes to all devices the user has specified to use iCloud services. The user may also choose to create local device notes that do not get synced ("On My Mac").

© 2022 Sarah Edwards



iOS FFS:
- `/private/var/mobile/Containers/Shared/AppGroup/<GUID>`
- `/private/var/mobile/Library/Notes (Legacy)`

iOS Backup:
- `/mobile/Applications/com.apple.notes`
- `/mobile/Applications/Notes (Legacy)`

macOS:
- `~/Library/Group Containers/group.com.apple.notes/`
- `~/Library/Containers/com.apple.Notes/ (Legacy)`

The Legacy iCloud/Local Notes were stored in a simpler database named `notes.sqlite`. On macOS, these databases had a version number associated with them. The newer database file is `NoteStore.sqlite`. Each version of the database may have slight differences in it.

© 2022 Sarah Edwards

NotesStore.sqlite - Note Containers and Folders

- Folders: ZICCLOUDSYNCINGOBJECT.Z_ENT = 11 and 12

```
SELECT
ZICCLOUDSYNCINGOBJECT.Z_PK,ZICCLOUDSYNCINGOBJECT.Z_ENT,
CASE ZICCLOUDSYNCINGOBJECT.Z_ENT
  WHEN 4 THEN "ATTACHMENT METADATA"
  WHEN 5 THEN "ATTACHMENT THUMBNAILS"
  WHEN 8 THEN "FILE"
  WHEN 9 THEN "NOTE"
  WHEN 11 THEN "FOLDER"
WHEN 12 THEN "FOLDER"
  ELSE ZICCLOUDSYNCINGOBJECT.Z_ENT
  END AS "OBJECT TYPE",
ZICCLOUDSYNCINGOBJECT.ZNAME,ZICCLOUDSYNCINGOBJECT.ZTITLE2,
ZICCLOUDSYNCINGOBJECT.ZACCOUNTNAMEFORACCOUNTLISTSORTING
FROM ZICCLOUDSYNCINGOBJECT
WHERE "OBJECT TYPE" = "FOLDER"
```

	Z_PK	Z_ENT	OBJECT TYPE	ZNAME	ZTITLE2	ZACCOUNTNAMEFORACCOUNTLISTSORTING
1	1	12	FOLDER	NULL	Notes	3_On My iPhone
2	2	12	FOLDER	NULL	Recently Deleted	3_On My iPhone
3	3	11	FOLDER	On My iPhone	NULL	3_On My iPhone
4	20	12	FOLDER	NULL	Notes	1_iCloud
5	22	12	FOLDER	NULL	Recently Deleted	1_iCloud
6	23	11	FOLDER	iCloud	NULL	1_iCloud

In the notes database, everything is mostly located in one table, `ZICCLOUDSYNCINGOBJECT`. Each object (i.e., Note, Folder, Attachment Metadata, File, etc.,) has a different entity number or Z_ENT. This query extracts each note folder and its associated details. We will be looking at Elwood's iCloud notes in Z_PK 20.

For this version of the database, Z_ENT of 11 is an overall note container (i.e., iCloud, or local notes "On My iPhone/Mac"). Z_ENT of 12 are note folders underneath these containers.

In future slides, we will see recently deleted notes are in Z_PK 22, and the iCloud note container, Z_PK 23.

Table: Z_PRIMARYKEY

	Z_ENT	Z_NAME
	Filter	Filter
1	1	ICAuthor
2	2	ICCloudState
3	3	ICCloudSyncingObject
4	4	ICAttachment
5	5	ICAttachmentPreviewImage
6	6	ICDeviceMigrationState
7	7	ICLegacyTombstone
8	8	ICMedia
9	9	ICNote
10	10	ICNoteContainer
11	11	ICAccount
12	12	ICFolder
13	13	ICDDevice
14	14	ICGroup
15	15	ICLocation
16	16	ICAttachmentLocation
17	17	ICNoteChange
18	18	ICNoteData
19	19	ICPerson
20	20	ICSearchIndexTransaction
21	21	ICServerChangeToken
22	22	NextId

© 2022 Sarah Edwards

```
SELECT
ZICCLOUDSYNCINGOBJECT.Z_PK,ZICCLOUDSYNCINGOBJECT.Z_ENT,
CASE ZICCLOUDSYNCINGOBJECT.Z_ENT
WHEN 4 THEN "ATTACHMENT METADATA"
WHEN 5 THEN "ATTACHMENT THUMBNAILS"
WHEN 8 THEN "FILE"
WHEN 9 THEN "NOTE"
WHEN 11 THEN "FOLDER"
WHEN 12 THEN "FOLDER"
ELSE ZICCLOUDSYNCINGOBJECT.Z_ENT
END AS "OBJECT TYPE",
ZICCLOUDSYNCINGOBJECT.ZNAME,ZICCLOUDSYNCINGOBJECT.ZTITLE2,
ZICCLOUDSYNCINGOBJECT.ZACCOUNTNAMEFORACCOUNTLISTSORTING
FROM ZICCLOUDSYNCINGOBJECT
WHERE "OBJECT TYPE" = "FOLDER"
```

	Z_PK	Z_ENT	OBJECT TYPE	ZNAME	ZTITLE2	ZACCOUNTNAMEFORACCOUNTLISTSORTING
1	1	12	FOLDER	*NULL*	Notes	3_On My iPhone
2	2	12	FOLDER	*NULL*	Recently Deleted	3_On My iPhone
3	3	11	FOLDER	On My iPhone	*NULL*	3_On My iPhone
4	20	12	FOLDER	*NULL*	Notes	1_iCloud
5	22	12	FOLDER	*NULL*	Recently Deleted	1_iCloud
6	23	11	FOLDER	iCloud	*NULL*	1_iCloud

© 2022 Sarah Edwards

NotesStore.sqlite - Notes

- Notes: Z_ENT = 9
- Secure Notes: Password-Protected Notes and Password Hint
- ZDATA = Note Contents


```
SELECT
ZICCLOUDSYNCINGOBJECT.Z_PK, ZICCLOUDSYNCINGOBJECT.Z_ENT,
CASE ZICCLOUDSYNCINGOBJECT.Z_ENT
  WHEN 4 THEN "ATTACHMENT METADATA"
  WHEN 5 THEN "ATTACHMENT THUMBNAILS"
  WHEN 8 THEN "FILE"
  WHEN 9 THEN "NOTE"
  WHEN 11 THEN "FOLDER"
WHEN 12 THEN "FOLDER"
  ELSE ZICCLOUDSYNCINGOBJECT.Z_ENT
  END AS "OBJECT TYPE",
  ZICCLOUDSYNCINGOBJECT.ZCREATIONDATE1,ZICCLOUDSYNCINGOBJECT.ZMODIFICATIONDATE1,ZICCLOUDSYNCINGOBJECT.ZLASTOPENEDDATE,ZICCLOUDSYNCINGOBJECT.ZMARKEDFORDELETION,
  ZICCLOUDSYNCINGOBJECT.ZTITLE1 AS "NOTE TITLE",ZICCLOUDSYNCINGOBJECT.ZSNIPPET,ZICCLOUDSYNCINCOBJECT.ZISPASSWORDPROTECTED AS PROTECTED,
  ZICCLOUDSYNCINGOBJECT.ZPASSWORDHINT,ZICCLOUDSYNCINGOBJECT.ZFOLDER,ZICCLOUDSYNCINGOBJECT.ZACCOUNT3,ZICCLOUDSYNCINGOBJECT.ZISPINNED,ZICNOTEDATA.ZDATA
  FROM ZICCLOUDSYNCINGOBJECT
  LEFT JOIN ZICNOTEDATA ON ZICCLOUDSYNCINGOBJECT.ZNOTEDATA == ZICNOTEDATA.Z_P
  WHERE "OBJECT TYPE" = "NOTE"
```

	Z_PK	Z_ENT	OBJECT TYPE	ZCREATIONDATE1	ZMODIFICATIONDATE1	ZLASTOPENEDDATE	ZMARKEDFORDELETION	NOTE TITLE	ZSNIPPET	PROTECTED	ZPASSWORDHINT	ZFOLDER	ZACCOUNT3	ZISPINNED	ZDATA
1	38	9	NOTE	642609284.550799	642609284.550799	642609285.352613	1	New Note	[redacted]	0	[redacted]	22	23	0	[redacted]
2	39	9	NOTE	642609175.562432	642609198.296603	643494580.629072	0	Secret stuff!	[redacted]	1	my normal password	20	23	0	[redacted]
3	40	9	NOTE	642605136.948026	643493927.571788	643482604.41471	0	Photos of me!		0	[redacted]	20	23	0	[redacted]
4	45	9	NOTE	642608668.768133	643482533.781124	643494497.261452	0	Things to do	Sleep all day	0	[redacted]	20	23	0	[redacted]

This query extracts the Note data to include timestamps, title, snippets, and whether or not it is encrypted and/or marked for deletion. The highlighted row contains a note that is named "`New Note`" and is not encrypted. If it was, the Protected column would show a "`1`". The note body is stored in a GZIP archive in the ZDATA column in the BLOB.

Next, we will look at the iCloud notes. The GUI shows three notes, one regular note, one shared, and one locked.

Notes for this version of the database use Z_ENT = 9. The notes shown contain metadata such as a variety of timestamps (i.e., created, modified, last viewed), if the note is marked for deletion, title, snippet, protection status and password hint, pinned status, and what folder/account it is associated with.

One note in the screenshot has been marked for deletion with a '1' in the associated column. Items that are marked for deletion will get cleaned up after a period of time.

The 'Secret Stuff!' note is a secure, protected note, with a password hint shown. Secure notes that have been synced from multiple devices may have different password hints. The cryptographic material required to unencrypted these notes are stored in the database, and not other common Apple places such as the keychain.

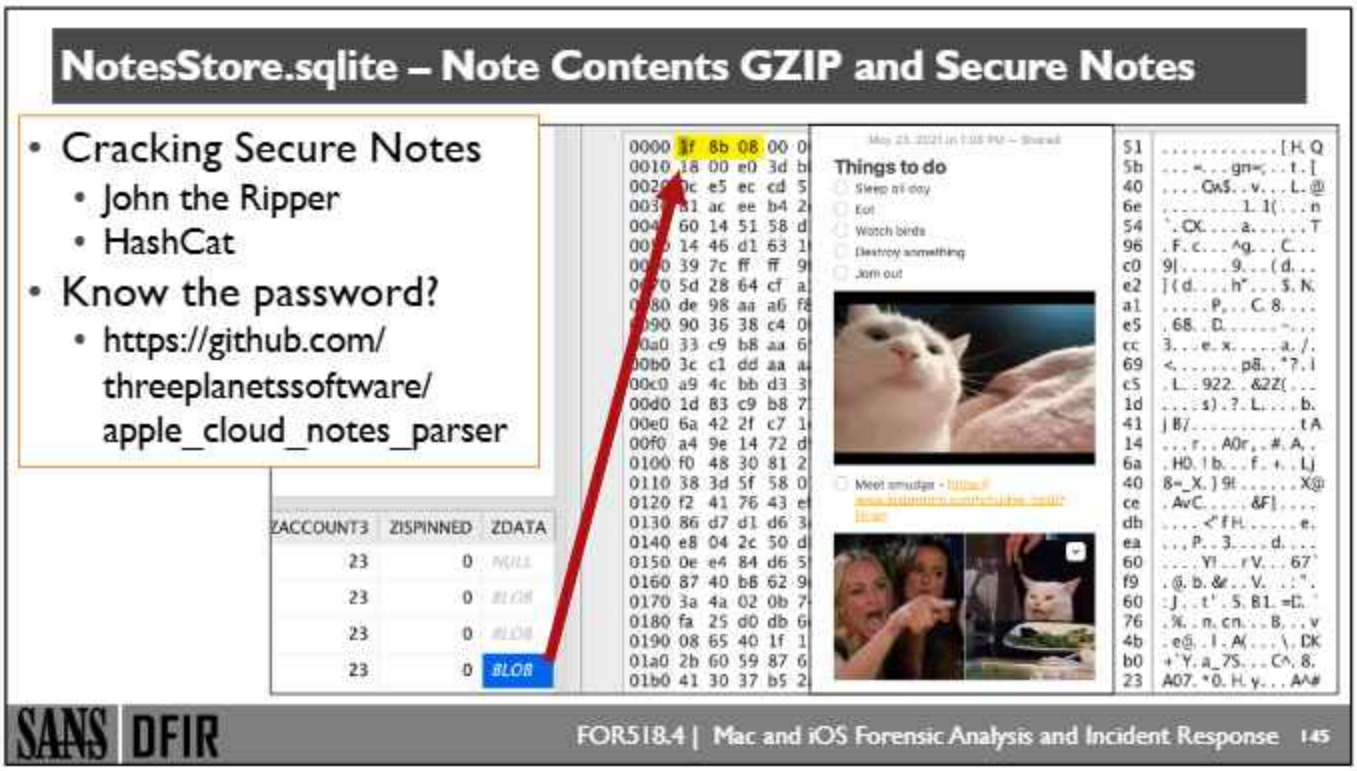
The note of interest for us is Z_PK =45. This note is in the folder Z_PK = 20, in the container Z_PK 23 discussed on the previous slide. The note contents can be extracted from the ZDATA column.

© 2022 Sarah Edwards

```
1  SELECT
2    ZICCLOUDSYNCINGOBJECT.Z_PK, ZICCLOUDSYNCINGOBJECT.Z_ENT,
3    CASE ZICCLOUDSYNCINGOBJECT.Z_ENT
4      WHEN 4 THEN "ATTACHMENT METADATA"
5      WHEN 5 THEN "ATTACHMENT THUMBNAILS"
6      WHEN 8 THEN "FILE"
7      WHEN 9 THEN "NOTE"
8      WHEN 11 THEN "FOLDER"
9      WHEN 12 THEN "FOLDER"
10     ELSE ZICCLOUDSYNCINGOBJECT.Z_ENT
11   END AS "OBJECT TYPE",
12   ZICCLOUDSYNCINGOBJECT.ZCREATIONDATE1,ZICCLOUDSYNCINGOBJECT.ZMODIFICATIONDATE1,ZICCLOUDSYNCINGOBJECT.ZLASTOPENEDDATE,ZICCLOUDSYNCINGOBJECT.ZMARKEDFORDELETION,
13   ZICCLOUDSYNCINGOBJECT.ZTITLE1 AS 'NOTE TITLE',ZICCLOUDSYNCINGOBJECT.ZSNIPPET,ZICCLOUDSYNCINGOBJECT.ZISPASSWORDPROTECTED AS PROTECTED,
14   ZICCLOUDSYNCINGOBJECT.ZPASSWORDHINT,ZICCLOUDSYNCINGOBJECT.ZFOLDER,ZICCLOUDSYNCINGOBJECT.ZACCOUNT3,ZICCLOUDSYNCINGOBJECT.ZISPINNED,ZICNOTEDATA.ZDATA
15   FROM ZICCLOUDSYNCINGOBJECT
16   LEFT JOIN ZICNOTEDATA ON ZICCLOUDSYNCINGOBJECT.ZNOTEDATA == ZICNOTEDATA.Z_P
17   WHERE "OBJECT TYPE" = "NOTE"
```

	Z_PK	Z_ENT	OBJECT TYPE	ZCREATIONDATE1	ZMODIFICATIONDATE1	ZLASTOPENEDDATE	ZMARKEDFORDELETION	NOTE TITLE	ZSNIPPET	PROTECTED	ZPASSWORDHINT	ZFOLDER	ZACCOUNT3	ZISPINNED	ZDATA
1	38	9	NOTE	642609284.550799	642609284.550799	642609285.352613	1	New Note	NULL	0	NULL	22	23	0	NULL
2	39	9	NOTE	642609175.562432	642609198.296609	643494500.629072	0	Secret stuff!	NULL	1	my normal password	20	23	0	BLOB
3	40	9	NOTE	642605156.948926	643493927.571789	643482604.41471	0	Photos of me!		0	NULL	20	23	0	BLOB
4	41	9	NOTE	642608669.768133	643482533.781124	643494497.261452	0	Things to do	Sleep all day	0	NULL	20	23	0	BLOB

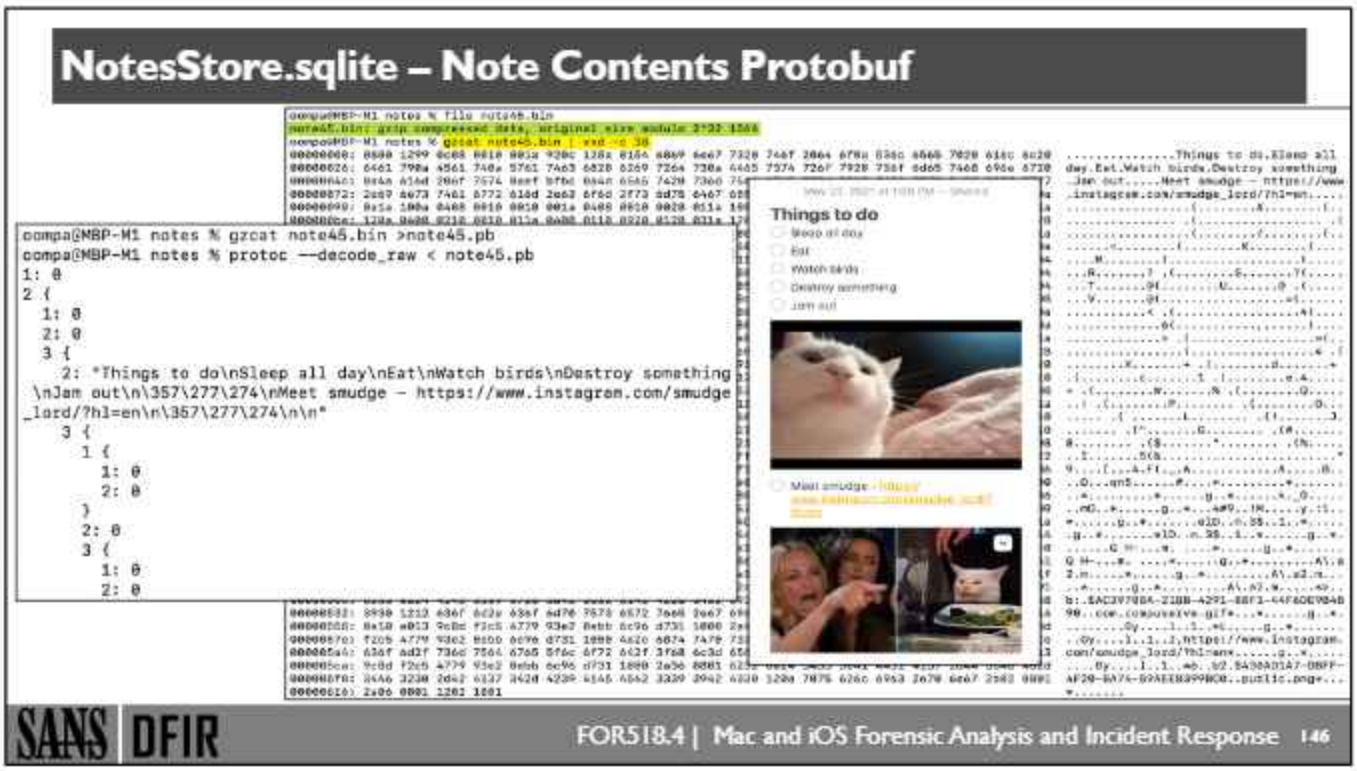
© 2022 Sarah Edwards



The ZDATA column contains the note contents, either encrypted or unencrypted. Unencrypted notes will be stored as GZIP archives with the signature 0x1F8B08. If unencrypted, the note's ZDATA BLOB can be extracted and unarchived from the compress GZIP format to reveal its contents.

If the note is encrypted, other tools may be required to reveal the contents of these BLOBs, such as Apple Cloud Notes parser from Jon Baumann of Ciofeca Forensics (https://www.ciofecaforensics.com) https://github.com/threeplanetssoftware/apple_cloud_notes_parser. This parser requires a known password, and that can be brute forced using John the Ripper or HashCat.

© 2022 Sarah Edwards



Once unarchived, the note content can be viewed as shown above. The contents are stored in a protobuf format (https://developers.google.com/protocol-buffers) which can be dumped using a variety of tools, such as protoc (shown), or CyberChef.

The key item for reviewing protobufs is knowing their template structure or .proto file. In most cases we do not have access to this, therefore we need to create our own. However, without knowing the structure, some items can be seen in the protobuf BLOB in the screenshot:

- Text of the note
- References to attachment files
- Embedded URLs

A fantastic resource on this process was completed by Jon Baumann of Ciofeca Forensics (https://www.ciofecaforensics.com) in a series of blog articles starting with this one: https://www.ciofecaforensics.com/2020/01/10/apple-notes-revisited/.

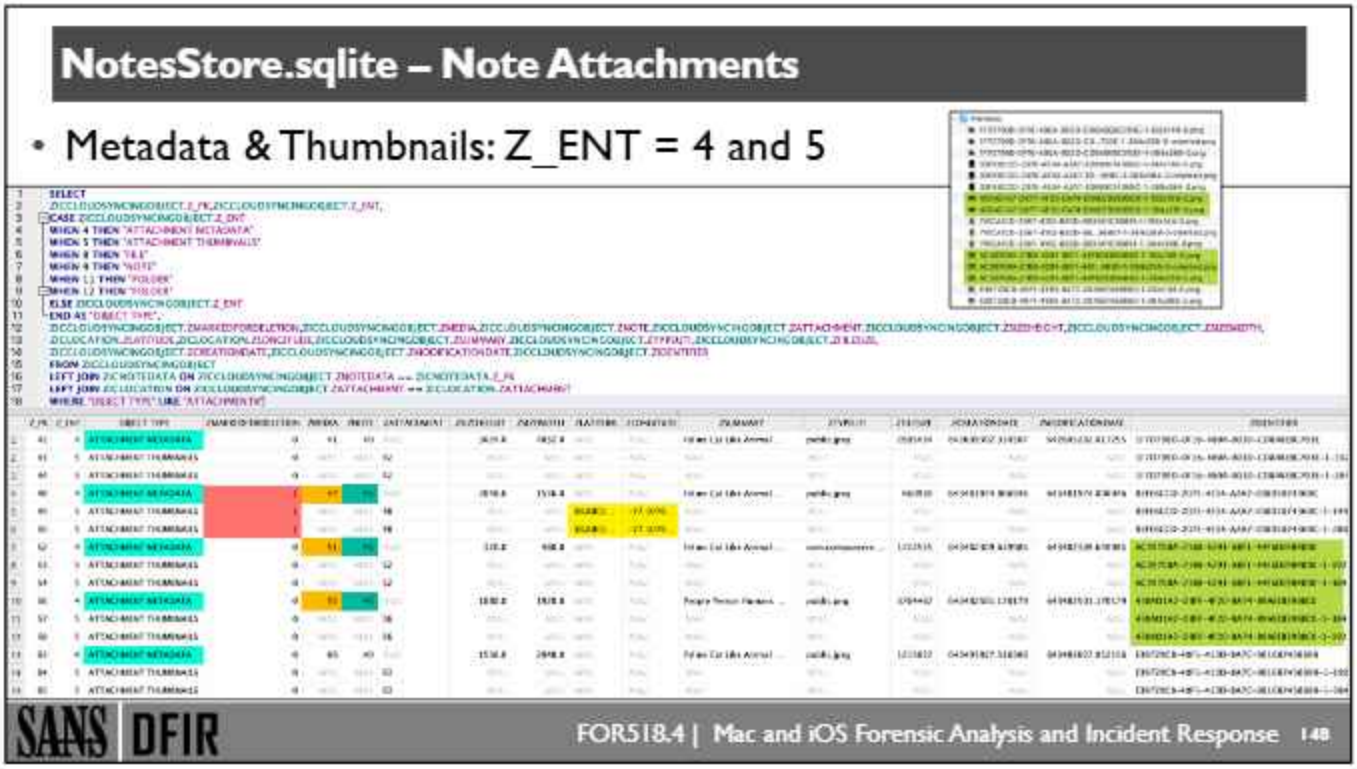
Many Apple databases include some sort of protobuf structure, a selection of which can be found at http://www.mac4n6.com/blog/2019/9/27/just-call-me-buffy-the-proto-slayer-an-initial-look-into-protobuf-data-in-mac-and-ios-forensics.

© 2022 Sarah Edwards

```
oompa@MBP-M1 notes % gzcat note45.bin >note45.pb
oompa@MBP-M1 notes % protoc --decode_raw < note45.pb
1: 0
2 {
  1: 0
  2: 0
  3 {
    2: "Things to do\nSleep all day\nEat\nWatch birds\nDestroy something
 \nJam out\n\357\277\274\nMeet smudge - https://www.instagram.com/smudge
_lord/?hl=en\n\357\277\274\n\n"
    3 {
      1 {
        1: 0
```

```
oompa@MBP-M1 notes % file note45.bin
note45.bin: gzip compressed data, original size
oompa@MBP-M1 notes % gzcat note45.bin | xxd -c
00000000: 0800 1299 0c08 0010 001a 920c 128a 01
00000026: 6461 798a 4561 740a 5761 7463 6820 62
0000004c: 0a4a 616d 206f 7574 0aef bfbc 0a4d 65
00000072: 2e69 6e73 7461 6772 616d 2e63 6f6d 2f
00000098: 0a1a 100a 0408 0010 0010 001a
000000be: 120a 0408 0210 0010 011a 0408
000000e4: 041a 100a 0408 0210 0410 1e1a
0000010a: 100a 0408 0110 3c10 0b1a 0408
00000130: 0408 0110 4d10 021a 0408 0110
00000156: 0801 1052 1001 1a04 0801 103f
0000017c: 0801 1054 1001 1a04 0801 1040
000001a2: 0801 1056 1001 1a04 0801 1040
000001c8: 0110 9901 1004 1a04 0801 103c
000001ee: 0508 0110 a601 1003 1a04 0801
00000214: 0508 0110 a501 1001 1a04 0801
0000023a: 110a 0508 0110 da01 1002 1a04
00000260: 181a 120a 0408 0110 5810 0b1a
00000286: 0128 1a1a 120a 0408 0110 6310
000002ac: 2b20 0128 1c1a 120a 0408 0110
000002d2: 0110 2120 0128 1e1a 120a 0408
000002f8: 0408 0110 1d20 0128 201a 120a
0000031e: 011a 0408 0110 1920 0128 221a
00000344: 3810 041a 0408 0110 0b20 0128
0000036a: 0110 4910 011a 0408 0110 3528
00000390: 390a 1b0a 105b a21f dc34 0e46
000003b6: 8808 44f4 8c03 716e 3510 a1dd
000003dc: 1801 2a88 0801 1204 0800 1801
00000402: 11d1 6d4f 1000 2a1e 0804 121a
00000428: 2a1e 080b 121a 0867 1801 2a14
0000044e: 0867 1801 2a14 0a10 85ef 0311
00000474: 0a10 841a edc0 5120 482d afa4
0000049a: 5120 482d afa4 ea6d 9420 ff93
000004c0: 3215 6dc6 141f 1000 2a1e 0804
000004e6: 1000 2a1e 0801 121a 0867 1801
0000050c: 623a 0a24 4143 3339 3730 3841
00000532: 3930 1212 636f 6d2e 636f 6d70 7573 6572 7665 2e67 6966 2a02 0801 2a1e 080e 121a 0867 1801 2a14
00000558: 0a10 e013 9c8d f2c5 4779 93e2 0ebb 6c96 d731 1000 2a4c 082c 121a 0867 1801 2a14 0a10 e013 9c8d
0000057e: f2c5 4779 93e2 0ebb 6c96 d731 1000 4a2c 6874 7470 733a 2f2f 7777 772e 696e 7374 6167 7261 6d2e
000005a4: 636f 6d2f 736d 7564 6765 5f6c 6f72 642f 3f68 6c3d 656e 2a1e 0801 121a 0867 1801 2a14 0a10 e013
000005ca: 9c8d f2c5 4779 93e2 0ebb 6c96 d731 1000 2a36 0801 6232 0a24 3433 3041 4431 4137 2d44 3846 462d
000005f0: 3446 3230 2d42 4137 342d 4239 4145 4542 3339 3942 4330 120a 7075 626c 6963 2e70 6e67 2a02 0801
00000616: 2a06 0801 1202 1801
```

© 2022 Sarah Edwards



Looking for attachment metadata for Note Z_PK =45, we need to now filter for attachment objects, Z_ENT 4 and 5.

Z_ENT 4 = Contains attachment metadata
Z_ENT 5 = Contains attachment thumbnail information

Attachment metadata contains details for each attachment for a note, including those that are marked for deletion as shown above. Note 45 appears to contain three attachments. However, only two are shown in previous screenshots of the note. A photo was attached to it and later removed, but details are still available. This photo also contained location coordinates (extracted from EXIF).

Other metadata includes summary information. Apple notes attempts to determine what was included in the photo, such as people, objects, and cats! File metadata such as type, height, width, file size, and timestamps are also stored.

The attachment thumbnail information contains the identifier for the thumbnail attachment that can be located in the Previews directory.

© 2022 Sarah Edwards

```
SELECT
ZICCLOUDSYNCINGOBJECT.Z_PK,ZICCLOUDSYNCINGOBJECT.Z_ENT,
CASE ZICCLOUDSYNCINGOBJECT.Z_ENT
WHEN 4 THEN "ATTACHMENT METADATA"
WHEN 5 THEN "ATTACHMENT THUMBNAILS"
WHEN 8 THEN "FILE"
WHEN 9 THEN "NOTE"
WHEN 11 THEN "FOLDER"
WHEN 12 THEN "FOLDER"
ELSE ZICCLOUDSYNCINGOBJECT.Z_ENT
END AS "OBJECT TYPE",
ZICCLOUDSYNCINGOBJECT.ZMARKEDFORDELETION,ZICCLOUDSYNCINGOBJECT.ZMEDIA,ZICCLOUDSYNCINGOBJECT.ZNOTE,ZICCLOUDSYNCINGOBJECT.ZATTACHMENT,ZICCLOUDSYNCINGOBJECT.ZSIZEHEIGHT,ZICCLOUDSYNCINGOBJECT.ZSIZEWIDTH,
ZICLOCATION.ZLATITUDE,ZICLOCATION.ZLONGITUDE,ZICCLOUDSYNCINGOBJECT.ZSUMMARY,ZICCLOUDSYNCINGOBJECT.ZTYPEUTI,ZICCLOUDSYNCINGOBJECT.ZFILESIZE,
ZICCLOUDSYNCINGOBJECT.ZCREATIONDATE,ZICCLOUDSYNCINGOBJECT.ZMODIFICATIONDATE,ZICCLOUDSYNCINGOBJECT.ZIDENTIFIER
FROM ZICCLOUDSYNCINGOBJECT
LEFT JOIN ZICNOTEDATA ON ZICCLOUDSYNCINGOBJECT.ZNOTEDATA == ZICNOTEDATA.Z_PK
LEFT JOIN ZICLOCATION ON ZICCLOUDSYNCINGOBJECT.ZATTACHMENT == ZICLOCATION.ZATTACHMENT
WHERE "OBJECT TYPE" LIKE "ATTACHMENT%"
```

	Z_PK	Z_ENT	OBJECT TYPE	ZMARKEDFORDELETION	ZMEDIA	ZNOTE	ZATTACHMENT	ZSIZEHEIGHT	ZSIZEWIDTH	ZLATITUDE	ZLONGITUDE	ZSUMMARY	ZTYPEUTI	ZFILESIZE	ZCREATIONDATE	ZMODIFICATIONDATE	ZIDENTIFIER
1	42	4	ATTACHMENT METADATA	0	41	40	NULL	1024.0	4032.0	NULL	NULL	Feline Cat Like Animal ...	public.jpeg	2695434	642609302.334507	642615202.017255	1F7D795D-0F16-480A-803D-C08A808C703E
2	43	5	ATTACHMENT THUMBNAILS	0	NULL	NULL	42	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	1F7D795D-0F16-480A-803D-C08A808C703E-1-192
3	44	5	ATTACHMENT THUMBNAILS	0	NULL	NULL	42	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	1F7D795D-0F16-480A-803D-C08A808C703E-1-384
4	48	4	ATTACHMENT METADATA	1	47	40	NULL	2048.0	1536.0	NULL	NULL	Feline Cat Like Animal ...	public.jpeg	660030	643481974.806946	643481974.806946	03F06CCD-207E-4E34-A2A7-E0893074309C
5	49	5	ATTACHMENT THUMBNAILS	1	NULL	NULL	48	NULL	NULL	38.8801...	-77.1076...	NULL	NULL	NULL	NULL	NULL	03F06CCD-207E-4E34-A2A7-E0893074309C-1-144
6	50	5	ATTACHMENT THUMBNAILS	1	NULL	NULL	48	NULL	NULL	38.8801...	-77.1076...	NULL	NULL	NULL	NULL	NULL	03F06CCD-207E-4E34-A2A7-E0893074309C-1-288
7	52	4	ATTACHMENT METADATA	0	51	40	NULL	320.0	480.0	NULL	NULL	Feline Cat Like Animal ...	com.compuserve...	1722555	643482309.639981	643482309.619385	AC39708A-21B8-4291-88F1-44F6D0984090
8	53	5	ATTACHMENT THUMBNAILS	0	NULL	NULL	52	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	AC39708A-21B8-4291-88F1-44F6D0984090-1-192
9	54	5	ATTACHMENT THUMBNAILS	0	NULL	NULL	52	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	AC39708A-21B8-4291-88F1-44F6D0984090-1-384
10	56	4	ATTACHMENT METADATA	0	55	40	NULL	1080.0	1920.0	NULL	NULL	People Person Humans ...	public.png	3764467	643482501.170179	643482501.170179	430AD1A7-38FF-4F20-8A74-89AEE81098C0
11	57	5	ATTACHMENT THUMBNAILS	0	NULL	NULL	56	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	430AD1A7-38FF-4F20-8A74-89AEE81098C0-1-384
12	58	5	ATTACHMENT THUMBNAILS	0	NULL	NULL	56	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	430AD1A7-38FF-4F20-8A74-89AEE81098C0-1-192
13	63	4	ATTACHMENT METADATA	0	66	40	NULL	1536.0	2048.0	NULL	NULL	Feline Cat Like Animal ...	public.jpeg	1215627	643493907.516300	643482627.052156	E39729CB-40F1-413D-8A7C-08168745B690
14	64	5	ATTACHMENT THUMBNAILS	0	NULL	NULL	63	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	E39729CB-40F1-413D-8A7C-08168745B690-1-192
15	65	5	ATTACHMENT THUMBNAILS	0	NULL	NULL	63	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	E39729CB-40F1-413D-8A7C-08168745B690-1-384

NotesStore.sqlite – Note Files

Files: Z_ENT = 8


```
1   SELECT
2   ZICCLOUDSYNCINGOBJECT.Z_PK,ZICCLOUDSYNCINGOBJECT.Z_ENT,
3   CASE ZICCLOUDSYNCINGOBJECT.Z_ENT
4   WHEN 4 THEN "ATTACHMENT METADATA"
5   WHEN 5 THEN "ATTACHMENT THUMBNAILS"
6   WHEN 8 THEN "FILE"
7   WHEN 9 THEN "NOTE"
8   WHEN 11 THEN "FOLDER"
9   WHEN 12 THEN "FOLDER"
10  ELSE ZICCLOUDSYNCINGOBJECT.Z_ENT
11  END AS "OBJECT TYPE",
12  ZICCLOUDSYNCINGOBJECT.ZMARKEDFORDELETION,ZICCLOUDSYNCINGOBJECT.ZATTACHMENT1,
13  ZICCLOUDSYNCINGOBJECT.ZIDENTIFIER,ZICCLOUDSYNCINGOBJECT.ZFILENAME
14  FROM ZICCLOUDSYNCINGOBJECT
15  WHERE "OBJECT TYPE" = "FILE"
```

	Z_PK	Z_ENT	OBJECT TYPE	ZMARKEDFORDELETION	ZATTACHMENT1	ZIDENTIFIER	ZFILENAME
1	41	8	FILE	0	42	25FE6154-E528-4BA3-9202-C39E0F12381C	image.jpeg
2	47	8	FILE	1	48	2CD36060-4700-412C-88A4-E64B8EF7B528	B788D696-A747-4765-B522-95F5AAEAEC2D.jpeg
3	51	8	FILE	0	52	B5288702-CEE0-4F8A-A1C6-F871E348A47D	5abb4690f6bd4ff75176c7ae60633245f17a00f2.gif
4	55	8	FILE	0	56	F22A2316-BA91-4B67-9331-7EA90E3D1ACE	1_4691735.png
5	66	8	FILE	0	63	75FD4AAA-3675-4F06-976C-1F08B4BAC8CF	7A789A98-A46D-4C29-AB8E-4EFEEBAA7A64.jpeg

Next, we will find the actual attachment files. These are Z_ENT = 8.

In the previous slide it showed which attachments belonged to note 45 in the ZNOTE column, we can find those using the Z_PKs of those entries as shown above.

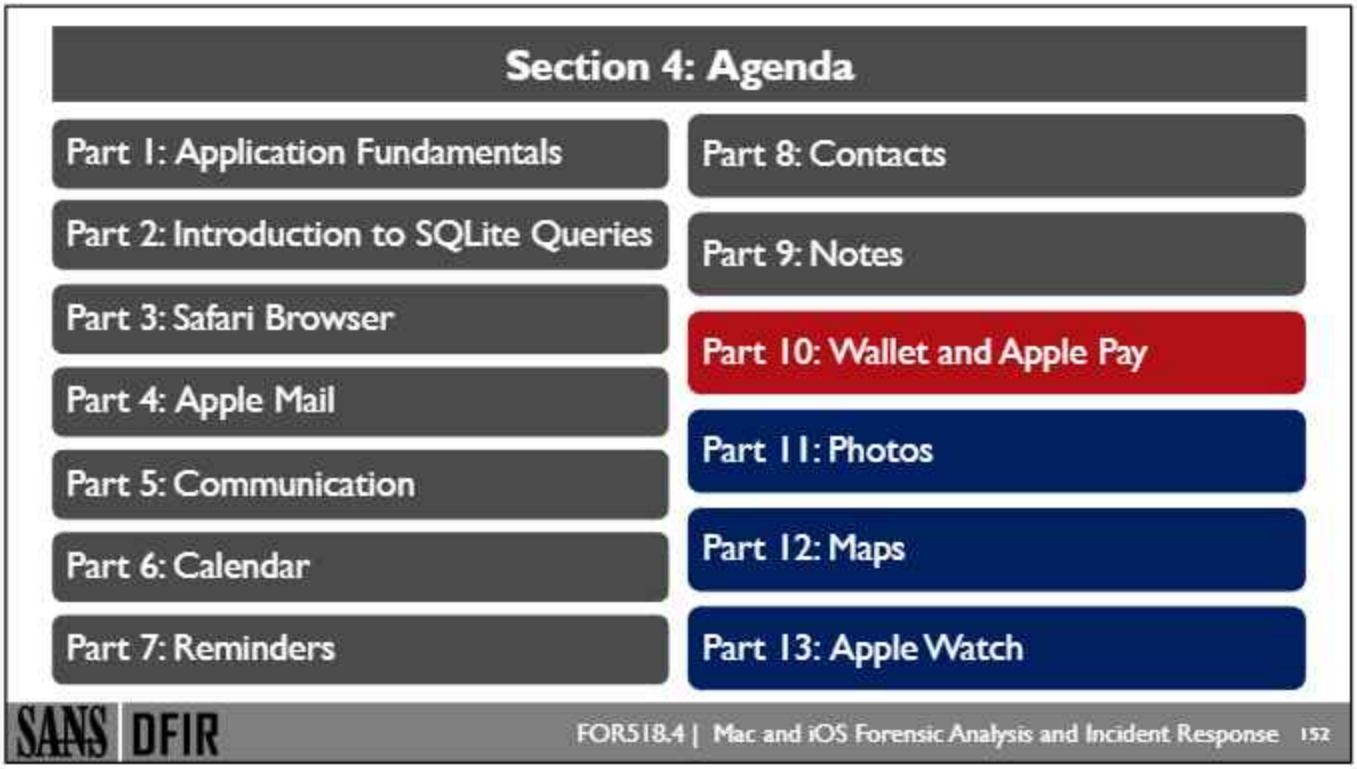
The database contains the UUID identifiers and filenames associated. These files can then be found in the Media directory. This includes the photo that was recently removed from the note.

© 2022 Sarah Edwards

```
SELECT
ZICCLOUDSYNCINGOBJECT.Z_PK,ZICCLOUDSYNCINGOBJECT.Z_ENT,
CASE ZICCLOUDSYNCINGOBJECT.Z_ENT
WHEN 4 THEN "ATTACHMENT METADATA"
WHEN 5 THEN "ATTACHMENT THUMBNAILS"
WHEN 8 THEN "FILE"
WHEN 9 THEN "NOTE"
WHEN 11 THEN "FOLDER"
WHEN 12 THEN "FOLDER"
ELSE ZICCLOUDSYNCINGOBJECT.Z_ENT
END AS "OBJECT TYPE",
ZICCLOUDSYNCINGOBJECT.ZMARKEDFORDELETION,ZICCLOUDSYNCINGOBJECT.ZATTACHMENT1,
ZICCLOUDSYNCINGOBJECT.ZIDENTIFIER,ZICCLOUDSYNCINGOBJECT.ZFILENAME
FROM ZICCLOUDSYNCINGOBJECT
WHERE "OBJECT TYPE" = "FILE"
```

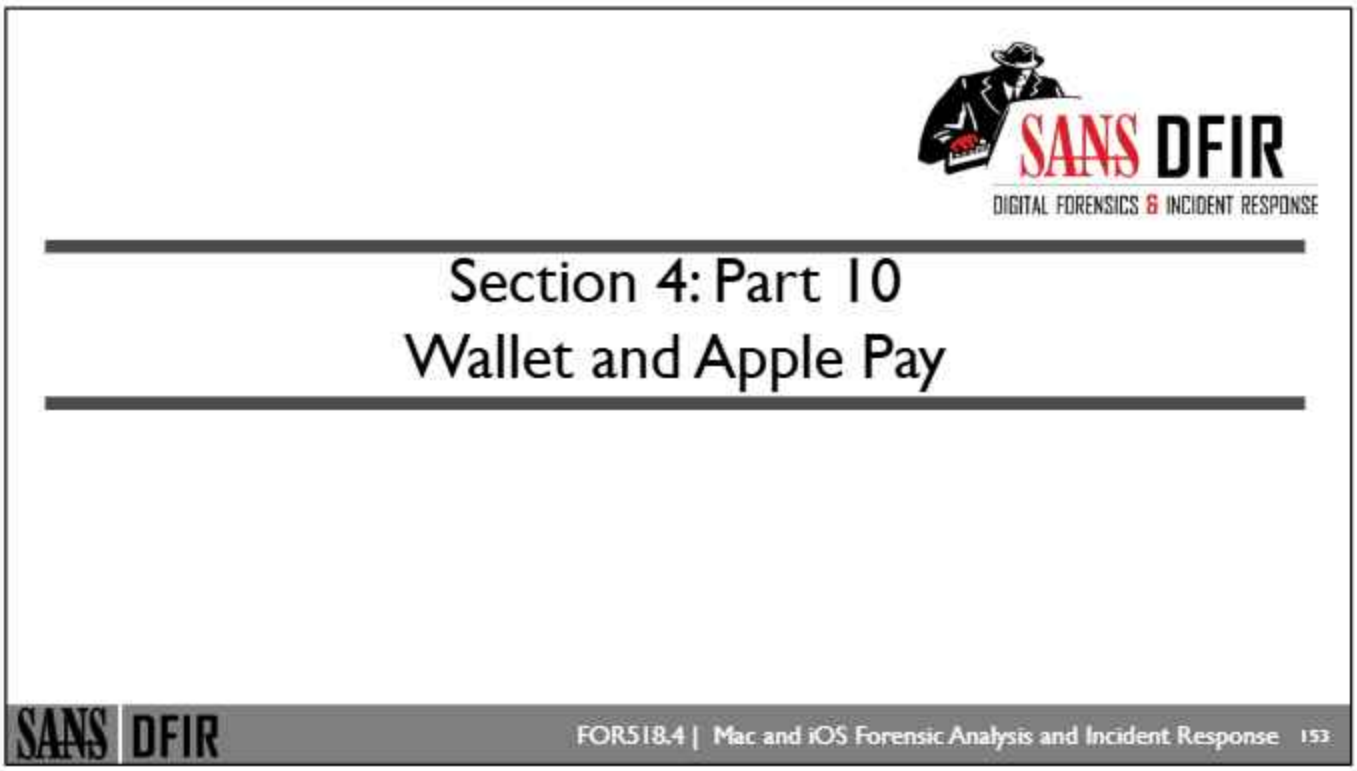
	Z_PK	Z_ENT	OBJECT TYPE	ZMARKEDFORDELETION	ZATTACHMENT1	ZIDENTIFIER	ZFILENAME
1	41	8	FILE	0	42	25FE6154-E528-4BA3-9202-C39E0F123B1C	Image.jpeg
2	47	8	FILE	1	48	2CD36060-4700-412C-88A4-E64B8EF7B528	B788D696-A747-4765-B522-95F5AAEAEC2D.jpeg
3	51	8	FILE	0	52	B5288702-CEE0-4F8A-A1C6-F871E348A47D	5abb4690f6bd4ff75176c7ae60633245f17a00f2.gif
4	55	8	FILE	0	56	F22A2316-BA91-4B67-9331-7EA90E3D1ACE	1_4691735.png
5	66	8	FILE	0	63	75FD4AAA-3675-4F06-976C-1F08B4BAC8CF	7A7B9A9B-A46D-4C29-AB8E-4EFEEBAA7A64.jpeg

© 2022 Sarah Edwards

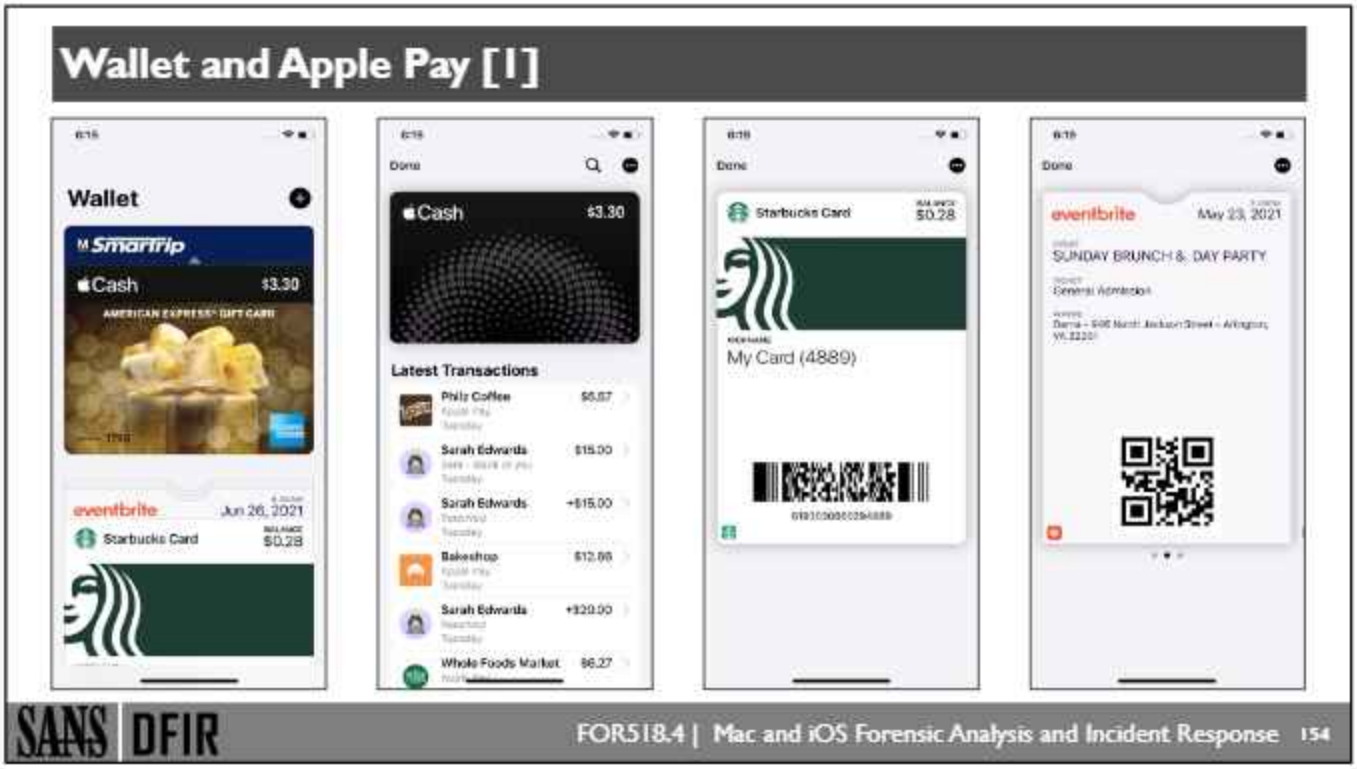


This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.



iOS:

- `/private/var/mobile/Library/Passes/`
  `(iCloud) /private/var/mobile/Library/Mobile Documents/com~apple~shoebox/UbiquitousCards/`

macOS:

- `~/Library/Passes/`
- `(iCloud) ~/Library/Mobile Documents/com~apple~shoebox/UbiquitousCards/`

The Wallet application can be used to keep track of various tickets and cards. If the user selects to add a credit card to the Apple Pay portion of this application, it can be used to purchase items. The cards may also include non-credit cards like the Washington, DC Metro card. The passes section can be used to hold tickets, reward cards, etc.

© 2022 Sarah Edwards

Wallet and Apple Pay [2]

Data Locations

- Primary Directory: Passes/
- iCloud Synced Data: UbiquitousCards/

Files

- /Cards - Cards and Passes
- /Mobile Documents/com~apple~shoebox/UbiquitousCards - iCloud Synced Passes
- passes23.sqlite - Main Database

iOS Physical:

- `/private/var/mobile/Library/Passes/`
- `(iCloud) /private/var/mobile/Library/Mobile Documents/com~apple~shoebox/UbiquitousCards/`

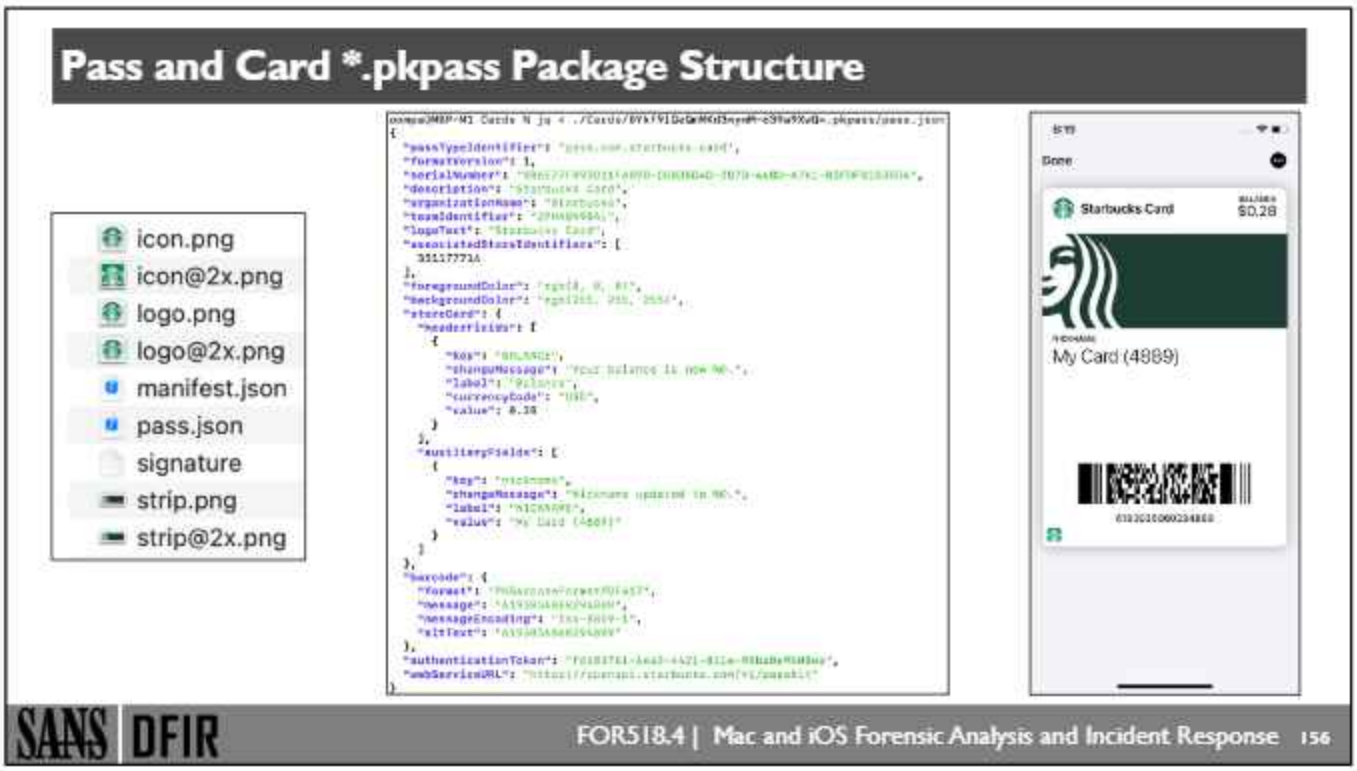
macOS:

- `~/Library/Passes/`
- `(iCloud) ~/Library/Mobile Documents/com~apple~shoebox/UbiquitousCards/`

The `Cards` directory may contain information about Apple Pay credit cards (also transit cards), and the passes. Some of the passes may be synced using iCloud to all devices signed into the user's iCloud account. These will be found in the `UbiquitiousCards` directory.

The primary database that keeps track of the cards, passes, and transactions (amongst a myriad of other items!) is the `passes23.sqlite` database.

© 2022 Sarah Edwards

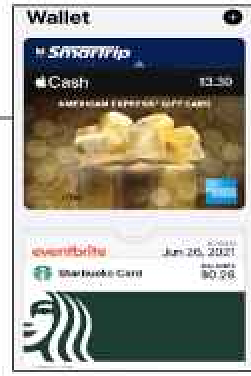


Each card or pass is stored in a `*.pkpass` directory. This directory is in a "Package" format. This directory may include lots of files and pictures of various logos associated with that card. The actual pass data is stored in the pass.json file.

The `pass.json` file can be reviewed in any text editor. Sometimes the developer of the "pass" doesn't include whitespace, so reviewing them can be difficult. Look for an online or offline JSON file formatter. In the screenshot above, jq is being used.

© 2022 Sarah Edwards

passes23.sqlite – Passes and Cards




```sql
SELECT
    DATETIME(PASS.INGESTED_DATE+978307200,'UNIXEPOCH') AS 'INGESTED DATE',
    DATETIME(PASS.MODIFIED_DATE+978307200,'UNIXEPOCH') AS 'MODIFIED DATE',
    CASE PASS.PASS_FLAVOR
        WHEN 0 THEN 'PASS'
        WHEN 1 THEN 'CARD'
    END 'TYPE',
    PASS.UNIQUE_ID AS 'UNIQUE ID',PASS.ORGANIZATION_NAME AS 'ORGANIZATION NAME',
    PASS_GROUP.PASS_TYPE_ID AS 'PASS TYPE',PASS_GROUP.GROUP_ORDER AS 'GROUP ORDER',
    PASS.PRIMARY_ACCOUNT_SUFFIX AS 'ACCOUNT SUFFIX',
    PASS.ISSUER_COUNTRY_CODE AS 'ISSUER COUNTRY CODE'
FROM PASS
LEFT JOIN LOCATION_SOURCE ON LOCATION_SOURCE.URI LIKE '%'||PASS.UNIQUE_ID||'%'
LEFT JOIN LOCATION ON LOCATION.LOCATION_SOURCE_PID == LOCATION_SOURCE.PID
LEFT JOIN PASS_GROUP ON PASS.GROUP_PID == PASS_GROUP.PID
ORDER BY 'GROUP ORDER'
```

	INGESTED DATE	MODIFIED DATE	TYPE	UNIQUE ID	ORGANIZATION NAME	PASS TYPE	GROUP ORDER	ACCOUNT SUFFIX	ISSUER COUNTRY CODE
1	2021-05-13 16:45:55	2021-05-13 16:46:03	CARD	CxYCdnI6ZhkBWmyjVGxuW5sxu28=	Washington Metropolitan Area Transit Authority	paymentpass.com.apple	0	NULL	US
2	2021-05-13 14:28:09	2021-05-13 14:40:38	CARD	ja1b5pqfUKWn2H1s2wtMOHjKGhg=	Apple	paymentpass.com.apple	1	NULL	US
3	2021-05-13 16:08:25	2021-05-13 16:40:33	CARD	hhgHM1ydi4QGHzydBMRuL2XHdCY=	American Express	paymentpass.com.apple	2	1795	US
4	2021-05-13 22:52:42	2021-05-13 22:52:42	PASS	TPQ7KWAnjLWVBQmxVcxQx5SIHfg=	Sunday Jazz Brunch	pass.eventbrite.ticket	3	NULL	NULL
5	2021-05-13 22:54:03	2021-05-13 22:54:03	PASS	mbG2YuZ4isptoqW7V8EDMOjRco=	SUNDAY BRUNCH & DAY PARTY	pass.eventbrite.ticket	3	NULL	NULL
6	2021-05-13 22:56:21	2021-05-13 22:56:21	PASS	~fxgsuJWkDyF5Xmj2o8iqsySljeY=	College Park Farmer's Market @ Paint Branch Parkway	pass.eventbrite.ticket	3	NULL	NULL
7	2021-05-13 22:46:16	2021-05-24 19:05:32	PASS	0YkF9QeQmM4d1mymW-G39a9RwG=	Starbucks	pass.com.starbucks.card	4	NULL	NULL

The query above is extracting details about the cards and passes associated with Wallet. This includes the following timestamps:

- Ingested Date – When the card or pass was initially placed in the Wallet app
- Modified Date – If a pass has been modified (i.e., Airline ticket, gate change or seat upgrade)

Other information includes the type of item (Card or Pass), the unique ID (this matches the *.pkpass files), and the organization it is associated with. The country code for credit cards is also available.

The Group Order shows the order index of the cards and passes in the UI.

passes23.sqlite – Transactions

	TRANSACTION DATE	STATUS	TRANSACTION TYPE	AMOUNT	CURRENCY CODE	CARD TYPE	PAYMENT TYPE	NAME	MERCHANT NAME	MERCHANT CATEGORY	PASS ID
1	2021-05-13 14:41:14	COMPLETED	APPLE CASH PEER PAYMENT	10.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
2	2021-05-13 16:14:00	COMPLETED	APPLE CASH PEER PAYMENT	15.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
3	2021-05-13 16:45:41	COMPLETED	PURCHASE	12.0	USD	Discover	APPLE CASH	NULL	SMARTRIP WASHINGTON DC	Commuter Transport, Ferries	ja2nSpqTURWrzR1z2rHMOryjGhg=
4	2021-05-13 17:13:40	COMPLETED	PURCHASE	5.0	USD	Discover	APPLE CASH	Starbucks	STARBUCKS CARD ECIFT	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
5	2021-05-13 17:21:43	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=
6	2021-05-13 17:46:56	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=
7	2021-05-13 22:40:41	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=
8	2021-05-13 22:57:12	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=
9	2021-05-17 11:52:55	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=
10	2021-05-17 11:53:18	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=
11	2021-05-17 11:54:32	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=
12	2021-05-18 17:54:49	NOT COMPLETED	PURCHASE	NULL	NULL	American Express	CREDIT CARD	NULL	NULL	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=
13	2021-05-18 17:55:24	NOT COMPLETED	PURCHASE	NULL	NULL	American Express	CREDIT CARD	NULL	NULL	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=
14	2021-05-18 17:56:06	COMPLETED	PURCHASE	6.27	USD	Discover	APPLE CASH	Whole Foods Market	WHOLEFDS ABL 10042	Grocery Stores, Supermarkets	ja2nSpqTURWrzR1z2rHMOryjGhg=
15	2021-05-18 17:59:43	COMPLETED	APPLE CASH PEER PAYMENT	20.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
16	2021-05-18 18:09:33	COMPLETED	PURCHASE	12.88	USD	Discover	APPLE CASH	NULL	SQ *BAKESHOP	Miscellaneous General Merchandise	ja2nSpqTURWrzR1z2rHMOryjGhg=
17	2021-05-18 18:13:47	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
18	2021-05-18 18:14:05	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
19	2021-05-18 18:14:25	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
20	2021-05-18 18:15:10	COMPLETED	APPLE CASH PEER PAYMENT	15.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
21	2021-05-18 18:15:34	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
22	2021-05-18 18:37:32	COMPLETED	APPLE CASH PEER PAYMENT	15.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja2nSpqTURWrzR1z2rHMOryjGhg=
23	2021-05-18 18:43:44	TRANSIT	TRANSIT	NULL	NULL	CalcdMFareDBSrve	TRANSIT	NULL	NULL	NULL	[illegible]
24	2021-05-18 18:48:43	COMPLETED	PURCHASE	5.57	USD	Discover	APPLE CASH	Philz Coffee	SQ *PHILZ COFFEE	Eating Places, Restaurants	ja2nSpqTURWrzR1z2rHMOryjGhg=
25	2021-05-23 17:04:01	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	HWgRM1y6r4QCHuyd5MRuz2RhdCY=

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 158

The transactions are shown above (the SQL query is on the following page).

These transactions include the following:

- Time of the transaction
- Status – i.e., Did the transaction go through? Many 'non-completed' entries in this data due to lack of money on the American Express gift card and Wi-Fi issues.
- Transaction Type – Peer Payments being done through Apple Pay or 'regular' transactions
- Amount and currency information
- Card information
- Merchant information
- Unique ID for pass/card (same as *.pkpass)

It is worth noting that not all transactions may be saved in this database and only Apple Card transactions get synced across devices. These transactions were performed on this device (the example is from an iPhone). Depending on the card provider only the last few transactions may be saved. Those that have the Apple Card (not shown) will have far more transaction and card related information in this database.

© 2022 Sarah Edwards

```
SELECT
DATETIME(PAYMENT_TRANSACTION.TRANSACTION_DATE+978307200,'UNIXEPOCH') AS 'TRANSACTION DATE',
CASE PAYMENT_TRANSACTION.TRANSACTION_STATUS
WHEN -1 THEN 'TRANSIT'
WHEN 0 THEN 'NOT COMPLETED'
WHEN 1 THEN 'COMPLETED'
ELSE PAYMENT_TRANSACTION.TRANSACTION_STATUS
END 'STATUS',
CASE PAYMENT_TRANSACTION.TRANSACTION_TYPE
WHEN 0 THEN 'PURCHASE'
WHEN 2 THEN 'TRANSIT'
WHEN 3 THEN 'APPLE CASH PEER PAYMENT'
ELSE PAYMENT_TRANSACTION.TRANSACTION_TYPE
END 'TRANSACTION TYPE',
CASE WHEN PAYMENT_TRANSACTION.AMOUNT = 9223372036854775807 THEN 'NULL' ELSE PAYMENT_TRANSACTION.AMOUNT/100.00 END 'AMOUNT',
PAYMENT_TRANSACTION.CURRENCY_CODE AS 'CURRENCY CODE',PAYMENT_APPLICATION.DISPLAY_NAME AS 'CARD TYPE',
CASE PAYMENT_APPLICATION.PAYMENT_TYPE
WHEN 1 THEN 'APPLE CASH'
WHEN 2 THEN 'CREDIT CARD'
WHEN 1000 THEN 'TRANSIT'
ELSE PAYMENT_APPLICATION.PAYMENT_TYPE
END 'PAYMENT TYPE',
PLANTAINS.C AS 'NAME',PAYMENT_TRANSACTION.MERCHANT_NAME AS 'MERCHANT NAME',PAYMENT_TRANSACTION.MERCHANT_INDUSTRY_CATEGORY AS 'MERCHANT CATEGORY',PASS.UNIQUE_ID AS 'PASS ID'
FROM PAYMENT_TRANSACTION
JOIN PAYMENT_APPLICATION ON PAYMENT_TRANSACTION.SOURCE_PID == PAYMENT_APPLICATION.TRANSACTION_SOURCE_PID
JOIN PASS ON PAYMENT_APPLICATION.PASS_PID == PASS.PID
LEFT JOIN PLANTAINS ON PAYMENT_TRANSACTION.U == PLANTAINS.PID
ORDER BY TRANSACTION_DATE
```

© 2022 Sarah Edwards

	TRANSACTION DATE	STATUS	TRANSACTION TYPE	AMOUNT	CURRENCY CODE	CARD TYPE	PAYMENT TYPE	NAME	MERCHANT NAME	MERCHANT CATEGORY	PASS ID
1	2021-05-13 14:43:14	COMPLETED	APPLE CASH PEER PAYMENT	10.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
2	2021-05-13 16:14:08	COMPLETED	APPLE CASH PEER PAYMENT	15.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
3	2021-05-13 16:45:45	COMPLETED	PURCHASE	12.0	USD	Discover	APPLE CASH	NULL	SMARTRIP WASHINGTON DC	Commuter Transport, Ferries	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
4	2021-05-13 17:12:40	COMPLETED	PURCHASE	5.0	USD	Discover	APPLE CASH	Starbucks	STARBUCKS CARD EGIFT	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
5	2021-05-13 17:21:43	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=
6	2021-05-13 17:46:56	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=
7	2021-05-13 22:40:45	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=
8	2021-05-13 22:57:12	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=
9	2021-05-17 11:52:55	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=
10	2021-05-17 11:53:18	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=
11	2021-05-17 11:53:32	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=
12	2021-05-18 17:54:49	NOT COMPLETED	PURCHASE	NULL	NULL	American Express	CREDIT CARD	NULL	NULL	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=
13	2021-05-18 17:55:24	NOT COMPLETED	PURCHASE	NULL	NULL	American Express	CREDIT CARD	NULL	NULL	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=
14	2021-05-18 17:56:06	COMPLETED	PURCHASE	6.27	USD	Discover	APPLE CASH	Whole Foods Market	WHOLEFDS ARL 10042	Grocery Stores, Supermarkets	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
15	2021-05-18 17:59:43	COMPLETED	APPLE CASH PEER PAYMENT	20.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
16	2021-05-18 18:06:33	COMPLETED	PURCHASE	12.86	USD	Discover	APPLE CASH	NULL	SQ *BAKESHOP	Miscellaneous General Merchandise	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
17	2021-05-18 18:13:47	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
18	2021-05-18 18:14:03	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
19	2021-05-18 18:14:25	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
20	2021-05-18 18:15:16	COMPLETED	APPLE CASH PEER PAYMENT	15.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
21	2021-05-18 18:15:34	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
22	2021-05-18 18:37:32	COMPLETED	APPLE CASH PEER PAYMENT	15.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
23	2021-05-18 18:43:44	TRANSIT	TRANSIT	NULL	NULL	CubicMiFareDESFire	TRANSIT	NULL	NULL	NULL	GvYCdnt6ZtW8WmyIjVGxuW5su28=
24	2021-05-18 18:48:43	COMPLETED	PURCHASE	5.57	USD	Discover	APPLE CASH	Philz Coffee	SQ *PHILZ COFFEE	Eating Places, Restaurants	ja1bSpqfUKWnzR1s2ntMOnjIGhg=
25	2021-05-23 17:04:01	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hhgRM1y6i4QGHuyd9MRut2RHdCY=

© 2022 Sarah Edwards

passes23.sqlite – Apple Pay Peer Payments

```
1   SELECT
2   DATETIME(PAYMENT_TRANSACTION.TRANSACTION_DATE+978307200,'UNIXEPOCH') AS 'TRANSACTION DATE',
3   CASE WHEN PAYMENT_TRANSACTION.AMOUNT = 9223372036854775807 THEN 'NULL' ELSE PAYMENT_TRANSACTION.AMOUNT/100.00 END 'AMOUNT',
4   PAYMENT_TRANSACTION.CURRENCY_CODE AS 'CURRENCY CODE',
5   CASE PAYMENT_TRANSACTION.PEER_PAYMENT_TYPE
6   WHEN 0 THEN 'NON-PEER PAYMENT'
7   WHEN 1 THEN 'SENT'
8   WHEN 2 THEN 'RECEIVED'
9   ELSE PAYMENT_TRANSACTION.PEER_PAYMENT_TYPE
10  END 'PEER PAYMENT TYPE',
11  PAYMENT_TRANSACTION.PEER_PAYMENT_COUNTERPART_HANDLE AS 'PEER PAYMENT HANDLE',
12  PAYMENT_TRANSACTION.PEER_PAYMENT_MEMO AS 'PEER PAYMENT MEMO'
13  FROM PAYMENT_TRANSACTION
14  JOIN PAYMENT_APPLICATION ON PAYMENT_TRANSACTION.SOURCE_PID == PAYMENT_APPLICATION.TRANSACTION_SOURCE_PID
15  WHERE PAYMENT_TRANSACTION.TRANSACTION_TYPE = 3
16  ORDER BY TRANSACTION_DATE
```

	TRANSACTION DATE	AMOUNT	CURRENCY CODE	PEER PAYMENT TYPE	PEER PAYMENT HANDLE	PEER PAYMENT MEMO
1	2021-05-13 14:43:14	10.0	USD	RECEIVED	oompa@csh.rit.edu	NULL
2	2021-05-13 16:14:08	15.0	USD	RECEIVED	oompa@csh.rit.edu	For groceries
3	2021-05-18 17:59:43	20.0	USD	RECEIVED	oompa@csh.rit.edu	NULL
4	2021-05-18 18:15:16	15.0	USD	RECEIVED	oompa@csh.rit.edu	NULL
5	2021-05-18 18:37:32	15.0	USD	SENT	oompa@csh.rit.edu	Back at you

Transactions that are specific to Apple Cash are called 'peer payments' and have a few more pieces of related information.

Each peer payment will show if it was sent or received, to/from whom, and if a message is attached.

© 2022 Sarah Edwards

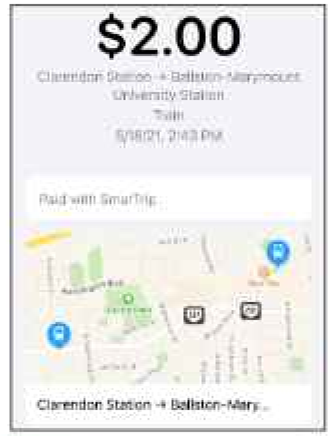
```
1   SELECT
2   DATETIME(PAYMENT_TRANSACTION.TRANSACTION_DATE+978307200,'UNIXEPOCH') AS 'TRANSACTION DATE',
3   CASE WHEN PAYMENT_TRANSACTION.AMOUNT = 9223372036854775807 THEN 'NULL' ELSE PAYMENT_TRANSACTION.AMOUNT/100.00 END 'AMOUNT',
4   PAYMENT_TRANSACTION.CURRENCY_CODE AS 'CURRENCY CODE',
5   CASE PAYMENT_TRANSACTION.PEER_PAYMENT_TYPE
6   WHEN 0 THEN 'NON-PEER PAYMENT'
7   WHEN 1 THEN 'SENT'
8   WHEN 2 THEN 'RECEIVED'
9   ELSE PAYMENT_TRANSACTION.PEER_PAYMENT_TYPE
10  END 'PEER PAYMENT TYPE',
11  PAYMENT_TRANSACTION.PEER_PAYMENT_COUNTERPART_HANDLE AS 'PEER PAYMENT HANDLE',
12  PAYMENT_TRANSACTION.PEER_PAYMENT_MEMO AS 'PEER PAYMENT MEMO'
13  FROM PAYMENT_TRANSACTION
14  JOIN PAYMENT_APPLICATION ON PAYMENT_TRANSACTION.SOURCE_PID == PAYMENT_APPLICATION.TRANSACTION_SOURCE_PID
15  WHERE PAYMENT_TRANSACTION.TRANSACTION_TYPE = 3
16  ORDER BY TRANSACTION_DATE
```

	TRANSACTION DATE	AMOUNT	CURRENCY CODE	PEER PAYMENT TYPE	PEER PAYMENT HANDLE	PEER PAYMENT MEMO
1	2021-05-13 14:43:14	10.0	USD	RECEIVED	oompa@csh.rit.edu	*NULL*
2	2021-05-13 16:14:08	15.0	USD	RECEIVED	oompa@csh.rit.edu	For groceries
3	2021-05-18 17:59:43	20.0	USD	RECEIVED	oompa@csh.rit.edu	*NULL*
4	2021-05-18 18:15:16	15.0	USD	RECEIVED	oompa@csh.rit.edu	*NULL*
5	2021-05-18 18:37:32	15.0	USD	SENT	oompa@csh.rit.edu	Back at you

© 2022 Sarah Edwards

passes23.sqlite - Transit

- Supported Transit Systems
- Start & End Station Information




```
SELECT
DATETIME(PAYMENT_TRANSACTION.TRANSACTION_DATE+978307200,'UNIXEPOCH') AS 'TRANSACTION DATE',DATETIME(PASS.INGESTED_DATE+978307200,'UNIXEPOCH') AS 'INGESTED DATE',
PASS.ORGANIZATION_NAME AS 'ORGANIZATION NAME',PAYMENT_APPLICATION.DISPLAY_NAME AS 'CARD TYPE',PASS.UNIQUE_ID AS 'PASS ID',
PAYMENT_TRANSACTION.START_STATION AS 'START STATION',PAYMENT_TRANSACTION.START_STATION_LATITUDE AS 'START STATION LAT',PAYMENT_TRANSACTION.START_STATION_LONGITUDE AS 'START STATION LONG',
PAYMENT_TRANSACTION.END_STATION AS 'END STATION',PAYMENT_TRANSACTION.END_STATION_LATITUDE AS 'END STATION LAT',PAYMENT_TRANSACTION.END_STATION_LONGITUDE AS 'END STATION LONG'
FROM PAYMENT_TRANSACTION
JOIN PAYMENT_APPLICATION ON PAYMENT_TRANSACTION.SOURCE_PID == PAYMENT_APPLICATION.TRANSACTION_SOURCE_PID
JOIN PASS ON PAYMENT_APPLICATION.PASS_PID == PASS.PID
```

	TRANSACTION DATE	INGESTED DATE	ORGANIZATION NAME	CARD TYPE	PASS ID	START STATION	START STATION LAT	START STATION LONG	END STATION	END STATION LAT	END STATION LONG
1	[illegible]	[illegible]	Washington Metropolitan Area Transit Authority	[illegible]	[illegible]	Clarendon Station	[illegible]	[illegible]	Ballston-Marymount University Station	[illegible]	[illegible]

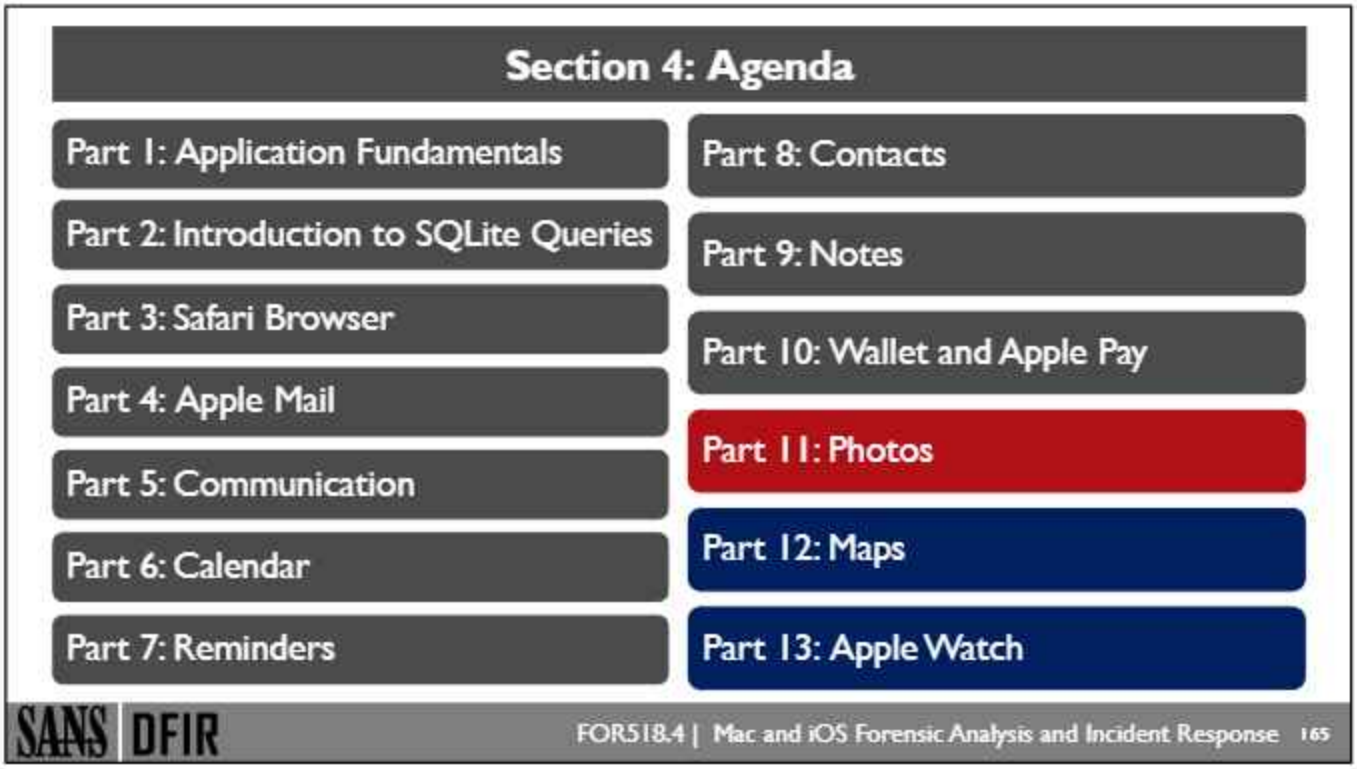
SANS DFIR FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 163

If the user has a transit card in their Wallet, these entries will also be stored in the database. In the example above, a ride was taken on the DC Metro (WAMATA) transit system from the Clarendon Station to the Ballston-Marymount University Station.

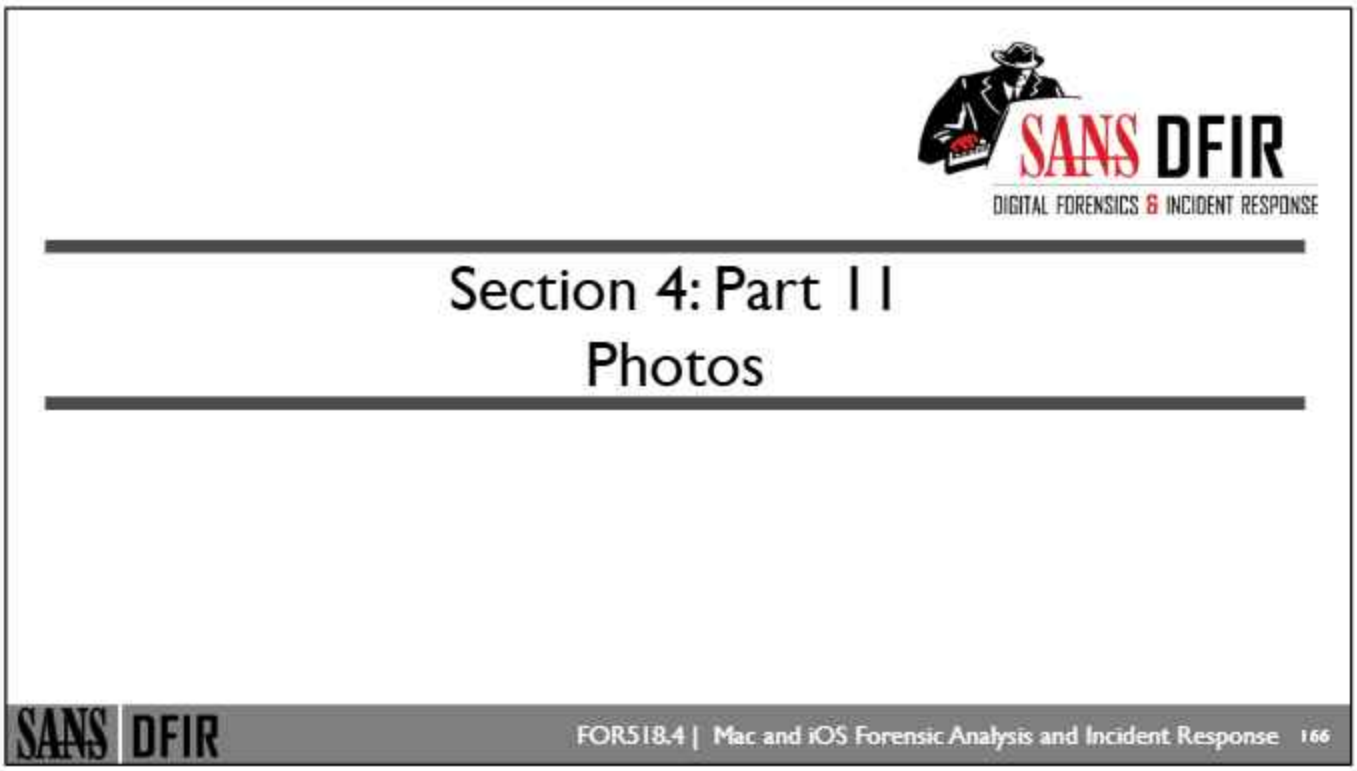
```sql
SELECT
DATETIME(PAYMENT_TRANSACTION.TRANSACTION_DATE+978307200,'UNIXEPOCH') AS 'TRANSACTION DATE',DATETIME(PASS.INGESTED_DATE+978307200,'UNIXEPOCH') AS 'INGESTED DATE',
PASS.ORGANIZATION_NAME AS 'ORGANIZATION NAME',PAYMENT_APPLICATION.DISPLAY_NAME AS 'CARD TYPE',PASS.UNIQUE_ID AS 'PASS ID',
PAYMENT_TRANSACTION.START_STATION AS 'START STATION',PAYMENT_TRANSACTION.START_STATION_LATITUDE AS 'START STATION LAT',PAYMENT_TRANSACTION.START_STATION_LONGITUDE AS 'START STATION LONG',
PAYMENT_TRANSACTION.END_STATION AS 'END STATION',PAYMENT_TRANSACTION.END_STATION_LATITUDE AS 'END STATION LAT',PAYMENT_TRANSACTION.END_STATION_LONGITUDE AS 'END STATION LONG'
FROM PAYMENT_TRANSACTION
JOIN PAYMENT_APPLICATION ON PAYMENT_TRANSACTION.SOURCE_PID == PAYMENT_APPLICATION.TRANSACTION_SOURCE_PID
JOIN PASS ON PAYMENT_APPLICATION.PASS_PID == PASS.PID
```

	TRANSACTION DATE	INGESTED DATE	ORGANIZATION NAME	CARD TYPE	PASS ID	START STATION	START STATION LAT	START STATION LONG	END STATION	END STATION LAT	END STATION LONG
1	2021-05-18 18:43:44	2021-05-13 16:45:55	Washington Metropolitan Area Transit Authority	CubicMIFareDESFire	GvYGdmt6ZtW8WmyIjVGxuW5su28=	Clarendon Station	38.8869662	-77.0953342	Ballston-Marymount University Station	38.8822259	-77.1115426

© 2022 Sarah Edwards



This page intentionally left blank.

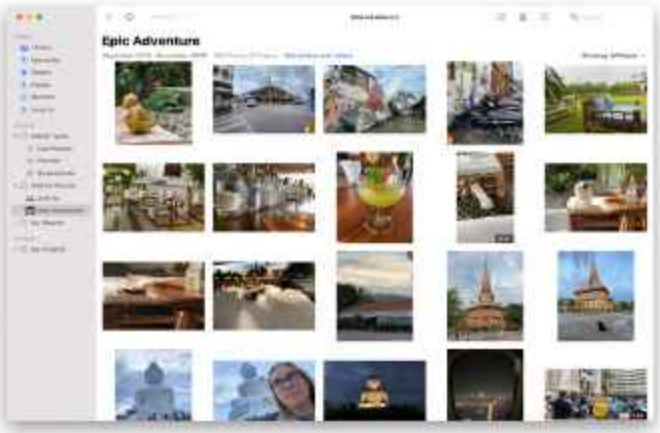


This page intentionally left blank.

© 2022 Sarah Edwards

Photos - com.apple.Photos

macOS: ~/Pictures/Photos Library.photoslibrary
iOS: [/private/var/]mobile/Media/PhotoData

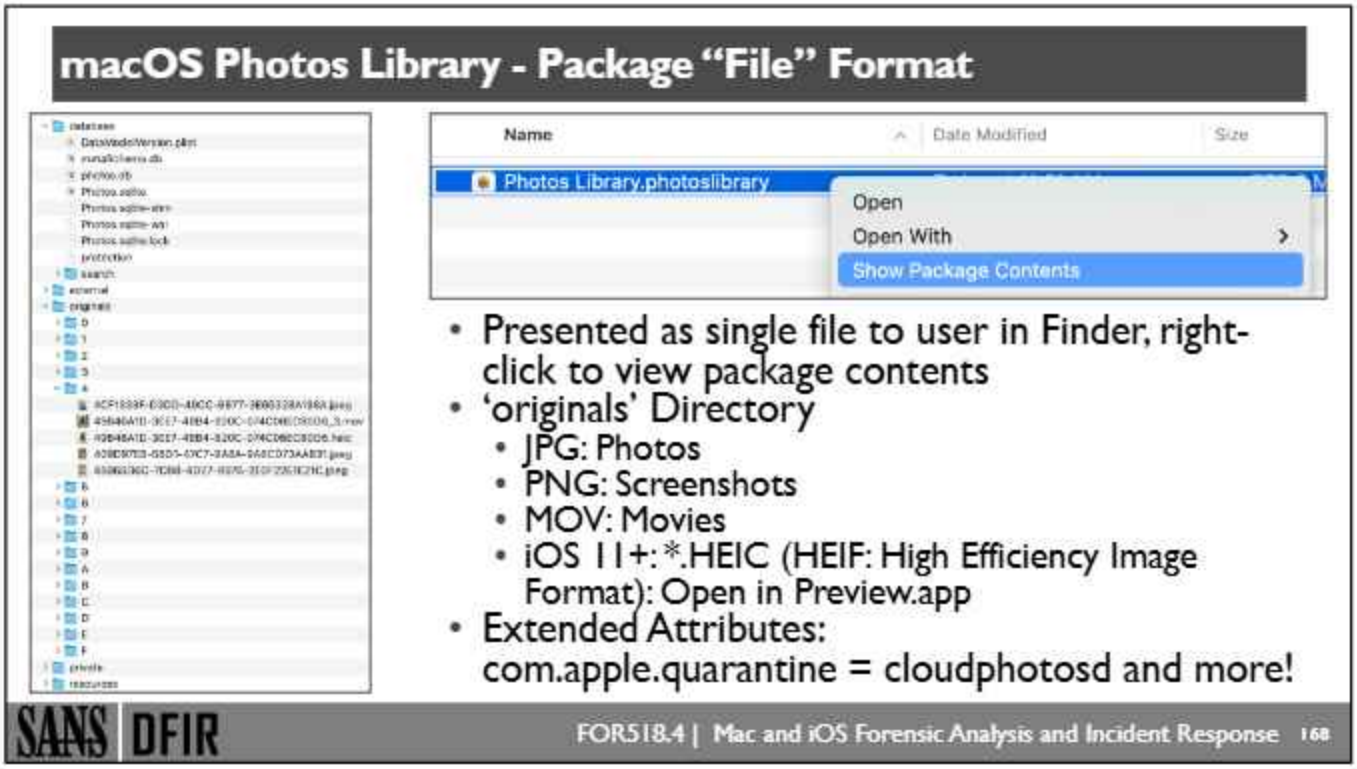


SANS DFIR FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 167

macOS: `~/Pictures/Photos Library.photoslibrary`
iOS: `[/private/var/]mobile/Media/PhotoData`

The native photo/media application is Photos. This app can store more than just photos but also screenshots and videos. Users can import media into this app and/or automatically sync their mobile device photos using iCloud syncing services.

© 2022 Sarah Edwards



macOS: `~/Pictures/Photos Library.photoslibrary`
iOS: `[/private/var/]mobile/Media/PhotoData`

The User's macOS Photo library can be found in a package file: `~/Pictures/Photos Library.photoslibrary/.`

Packages are directories that are presented to the user as a single file in Finder. To view the contents of a package, right-click and choose "`Show Package Contents`" or view it in a terminal window. This directory may contain may sub-directories, files, databases, thumbnails, etc.

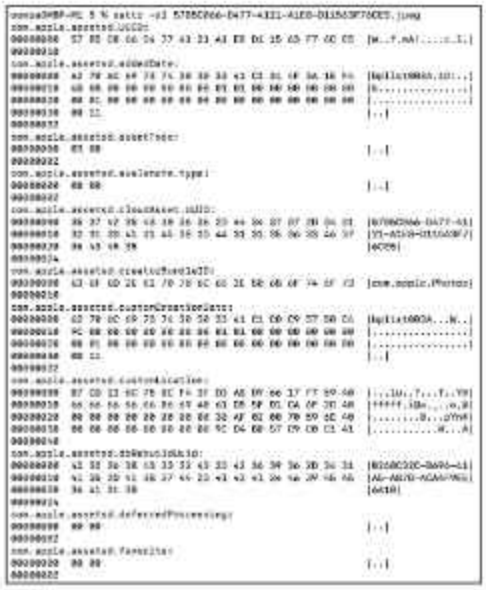
The original photos can be found in the originals directory. Since local and iCloud photos are now stored together, it might be necessary to determine which are iCloud pictures. This can be shown in the extended attributes for each file. If the `com.apple.quarantine` attribute contains `cloudphotosd`, then it came from iCloud.

Pictures taken with devices running iOS 11 will use a new format for storage called High Efficiency Image Format (HEIC extension).

References:
Apple Developer Website: About Bundles
https://developer.apple.com/library/archive/documentation/CoreFoundation/Conceptual/CFBundles/AboutBundles/AboutBundles.html#//apple_ref/doc/uid/10000123i-CH100-SW1
https://en.wikipedia.org/wiki/High_Efficiency_Image_File_Format

© 2022 Sarah Edwards

macOS Photos – Extended Attributes – com.apple.assetsd.*



Each piece of media may have a set of extended attributes. The Photos app likes to add the com.apple.assetsd.* attributes.

The example above is the set of xattrs for a single photo. Notable attributes contain the original filename, location, favorite flag, hidden flag, time zone, and quarantine information.

© 2022 Sarah Edwards

Photo Databases – photos.db or Photos.sqlite

macOS - ~/Pictures/Photos Library.photoslibrary/database/photos.db or photos.sqlite

iOS - [/private/var/]mobile/Media/PhotoData/Photos.sqlite

Features:

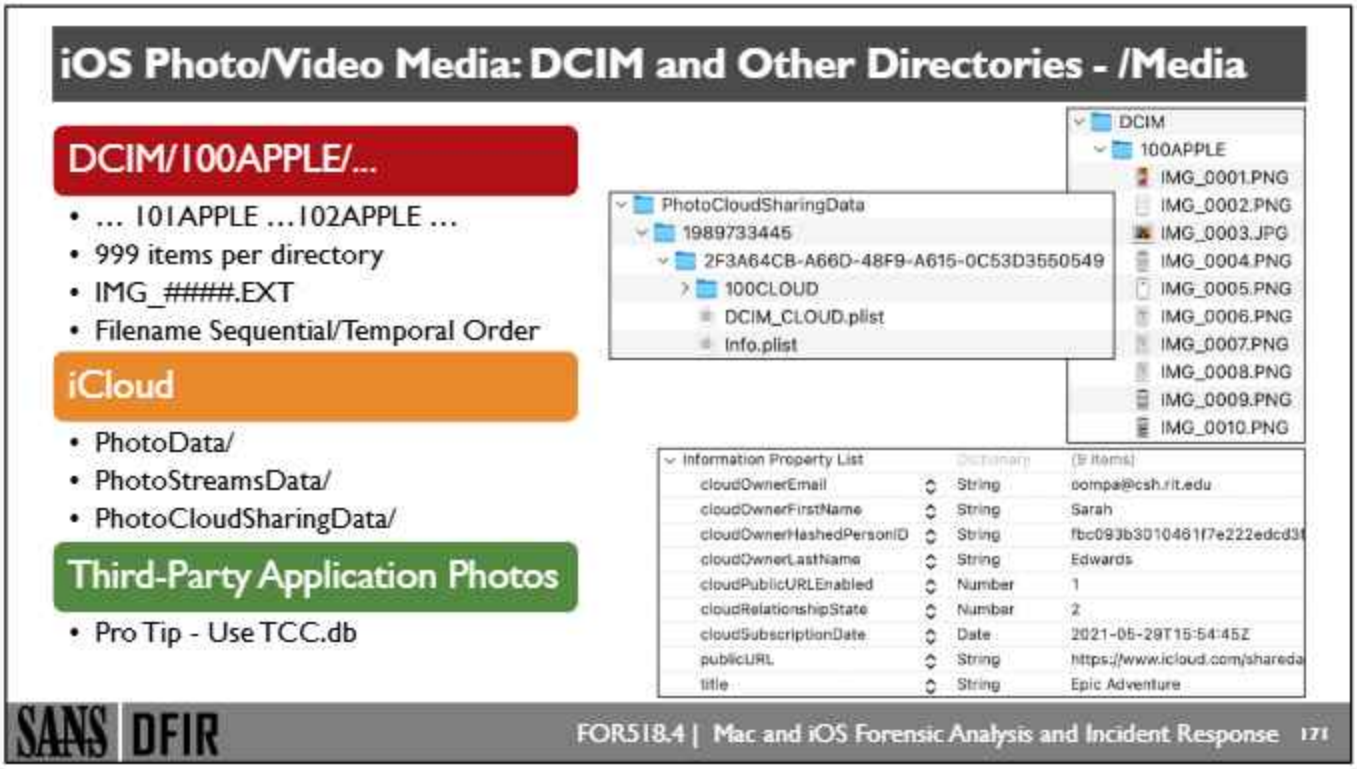
- Filename, Timestamps, Time Zones, Height, Width, Rotation
- Associated Notes
- Location Latitude/Longitude, Reversed Location Protobuf Data (Similar to Reverse IP Location)
- Albums/Moments/Faces/Places
- Usage

macOS: `~/Pictures/Photos Library.photoslibrary/database/photos.db` or `photos.sqlite`
iOS: `[/private/var/]mobile/Media/PhotoData/Photos.sqlite`

The metadata for each photo (or other media) is found in the Photos.db/Photos.sqlite databases. These databases contain nearly the same type of data; however, the table and column names may differ.

Notable items in the database are extracted EXIF metadata, location information, annotated notes, viewing usage, detected faces and objects, and organization (Albums, Moments, etc.).

© 2022 Sarah Edwards



iOS: `/[private/var/]mobile/Media/`

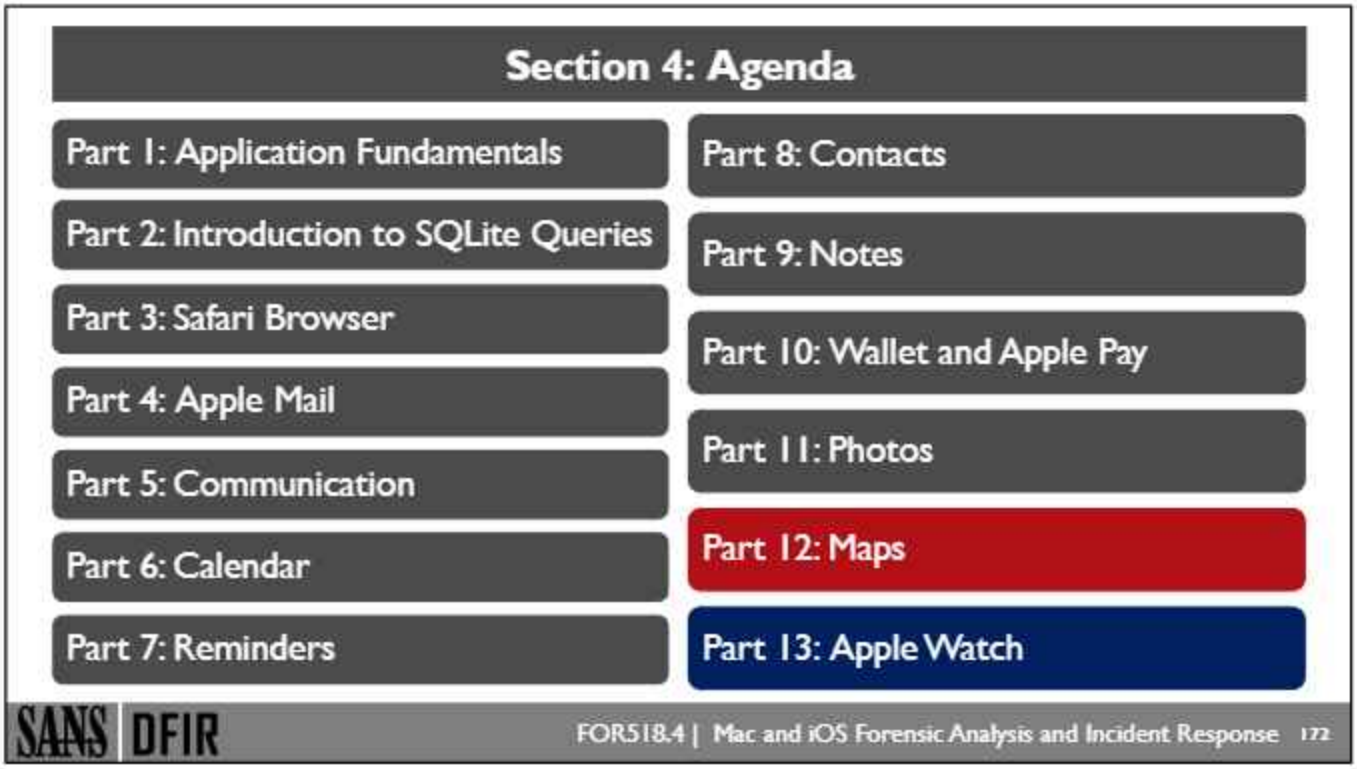
Photos, videos, and screenshots that are taken with the device will be stored in the DCIM directory. This directory works just like it does on other camera devices. This directory may contain multiple other directories named `100APPLE`, `101APPLE`, and so on. Each of these may contain up to 999 photos, videos, and/or screenshots.

Each image is named sequentially starting with `IMG_0001.EXT`, where EXT can be JPG, HEIC for a photo, MOV for a movie, or PNG for a screenshot.

Photos, videos, and screenshots can be strewn about a device in many directories, depending on how the user uses their device, especially if iCloud is used. These directories include the DCIM, PhotoData, and PhotoStreamsData directories.

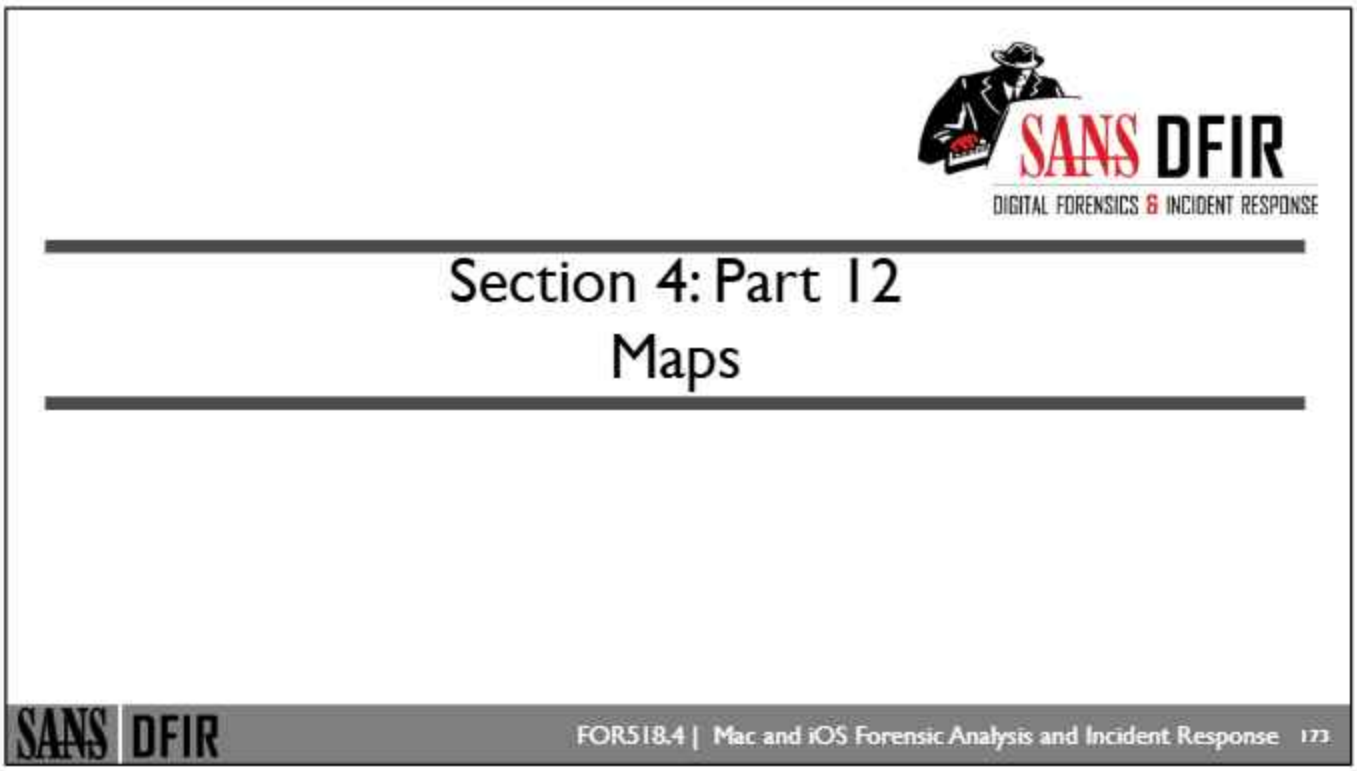
Do not forget to look at third-party applications, as they may also contain media of interest. An easy way to tell which applications might be of investigative value is to look at the TCC.db database for applications using the Camera, Microphone, Photo Album permissions.

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.

Maps - com.apple.Maps

macOS 11+, iOS 14+

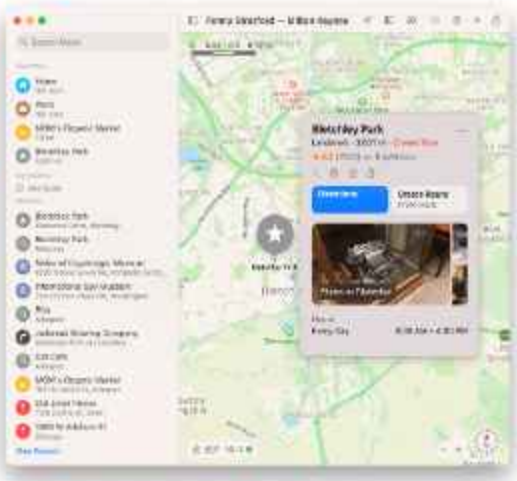
- Database

macOS 10.15-, iOS 13-

- Plist Files

Data Types

- Searches
- History
- Directions
- Bookmarks/Favorites
- Dropped Pins
- Collections



macOS:

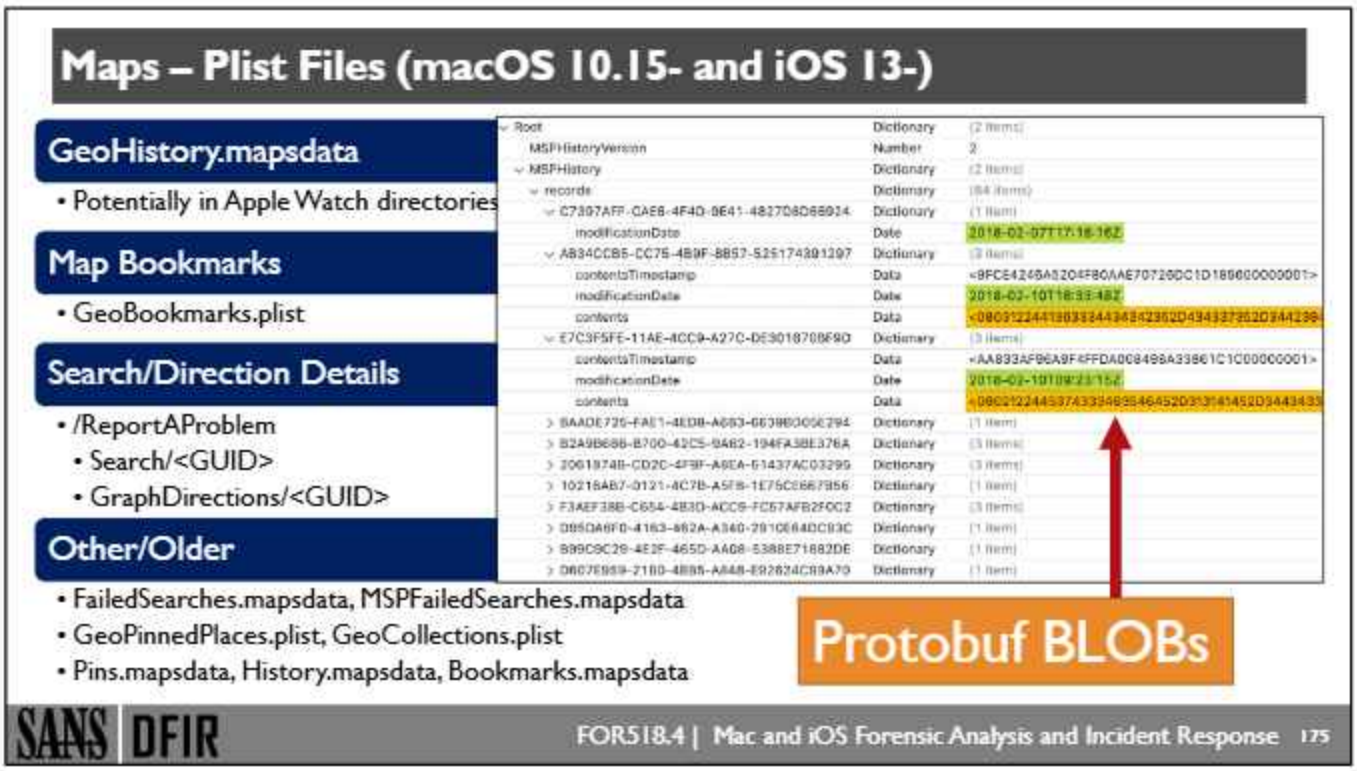
- `~/Library/Containers/com.apple.Maps/`

iOS:

- `FFS: /private/var/mobile/Containers/Data/Application/<GUID>/Library/Maps/`
- `Backup: /mobile/Applications/com.apple.Maps/ (May not always contain all files)`

The native mapping application is Apple Maps. The Maps application is available for both iOS and macOS, and the files are very similar. Maps data can be synced to all devices using iCloud.

© 2022 Sarah Edwards



The data files for Maps has changed in iOS 14 and macOS 11 to a database. Before this update, they were stored in many plist files scattered throughout the file system.

The `*.mapsdata` files are stored in a plist file that contains BLOB protobuf location data that is hard to parse. The protobuf BLOB data can be extracted and viewed in a hex editor to determine some of the location data.

The `GeoHistory.mapsdata` plist file contains individual history items, but in the protobuf BLOB format. Each item may contain a start and finish route, possible review information, and/or reverse geolocation data.

Each item contains a GUID as a key in the plist, or as the first few bytes of the BLOB that can be used to potentially correlate to an on-disk file with a matching GUID in the ReportAProblem directory.

The ReportAProblem GUID files are a single protobuf file that contains route data in the `/Directions` and `/GraphDirections` directories, and possibly review sites and other search data in the `/Search` directory.

The GUID in each `GeoHistory.mapsdata` item can be matched to files in the `/Search`, `/GraphDirections`, or `/Direction` directories to find additional related information.

These files do not appear to get backed up in iTunes backups.

© 2022 Sarah Edwards

Maps on iOS 14 and macOS 11 - MapsSync_0.0.1 Database

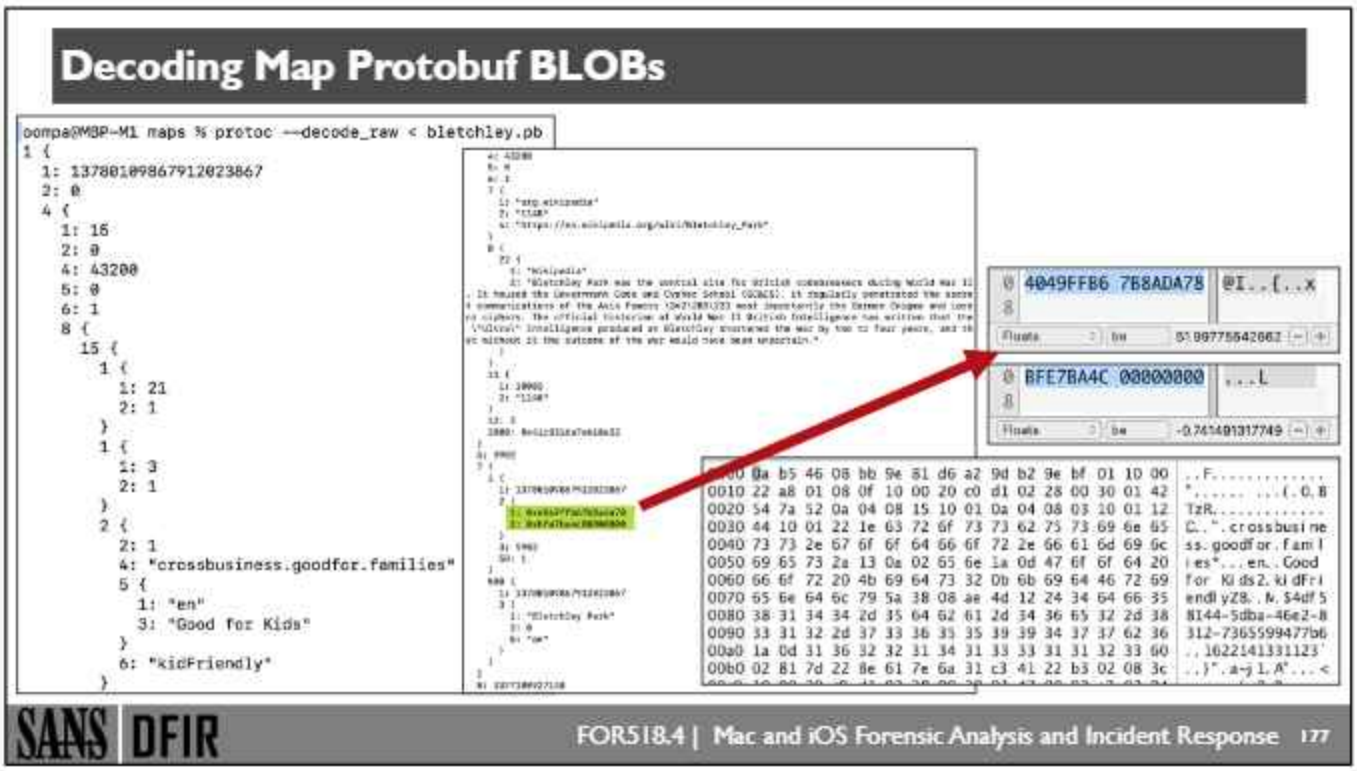
```
1   SELECT
2   ZMIXINMAPITEM.ZLATITUDE,
3   ZMIXINMAPITEM.ZLONGITUDE,
4   ZMIXINMAPITEM.ZMAPITEMSTORAGE,
5   ZFAVORITEITEM.ZCREATETIME,
6   ZFAVORITEITEM.ZORIGINATINGADDRESSSTRING,
7   ZHISTORYITEM.ZCREATETIME,
8   ZHISTORYITEM.ZMODIFICATIONTIME
9   FROM ZMIXINMAPITEM
10  LEFT JOIN ZFAVORITEITEM ON ZMIXINMAPITEM.ZFAVORITEITEM = ZFAVORITEITEM.Z_PK
11  LEFT JOIN ZHISTORYITEM ON ZMIXINMAPITEM.ZHISTORYPLACEITEM = ZHISTORYITEM.Z_PK
```

	ZLATITUDE	ZLONGITUDE	ZMAPITEMSTORAGE	ZCREATETIME	ZORIGINATINGADDRESSSTRING	ZCREATETIME	ZMODIFICATIONTIME
1	41.9475159	-87.6562758	BLOB	NULL	NULL	642621434.79447	642621434.794477
2	41.5456452449917	-88.0741658806801	BLOB	NULL	NULL	642621516.360475	642621516.360504
3	38.89634367153	-77.0861399173737	BLOB	NULL	NULL	642621578.439475	642621578.439485
4	38.89634367153	-77.0861399173737	BLOB	642621583.622226	NULL	NULL	NULL
5	38.883934378972	-77.025511264801	BLOB	NULL	NULL	644011012.845177	644011012.845219
6	39.1148200530054	-76.7746871709824	BLOB	NULL	NULL	644011023.729719	644011023.729728
7	41.9475159	-87.6562758	BLOB	644011136.511095	1060 W Addison Chicago IL 60613 United States	NULL	NULL
8	41.5463347	-88.0733369	BLOB	644011136.511632	1125 Collins St Joliet IL 60432 United States	NULL	NULL
9	51.99775642662	-0.741491317749023	BLOB	NULL	NULL	644011262.225946	644011262.22595
10	51.99775642662	-0.741491317749023	BLOB	644011269.891376	NULL	NULL	NULL

Starting with iOS14 and macOS 11, everything Maps-related is now stored in a SQLite database. However, the older files may still exist on the device, so it's still worth checking for those if looking for historical information.

This database contains everything that was previously stored in separate plist files. Favorites, collections, history items, as well as the location protobuf files stored in ZMAPITEMSTORAGE.

© 2022 Sarah Edwards



The protobuf BLOBs may be embedded in plist files, databases, or just files themselves. They look like the example above; some may have more data; some may have less.

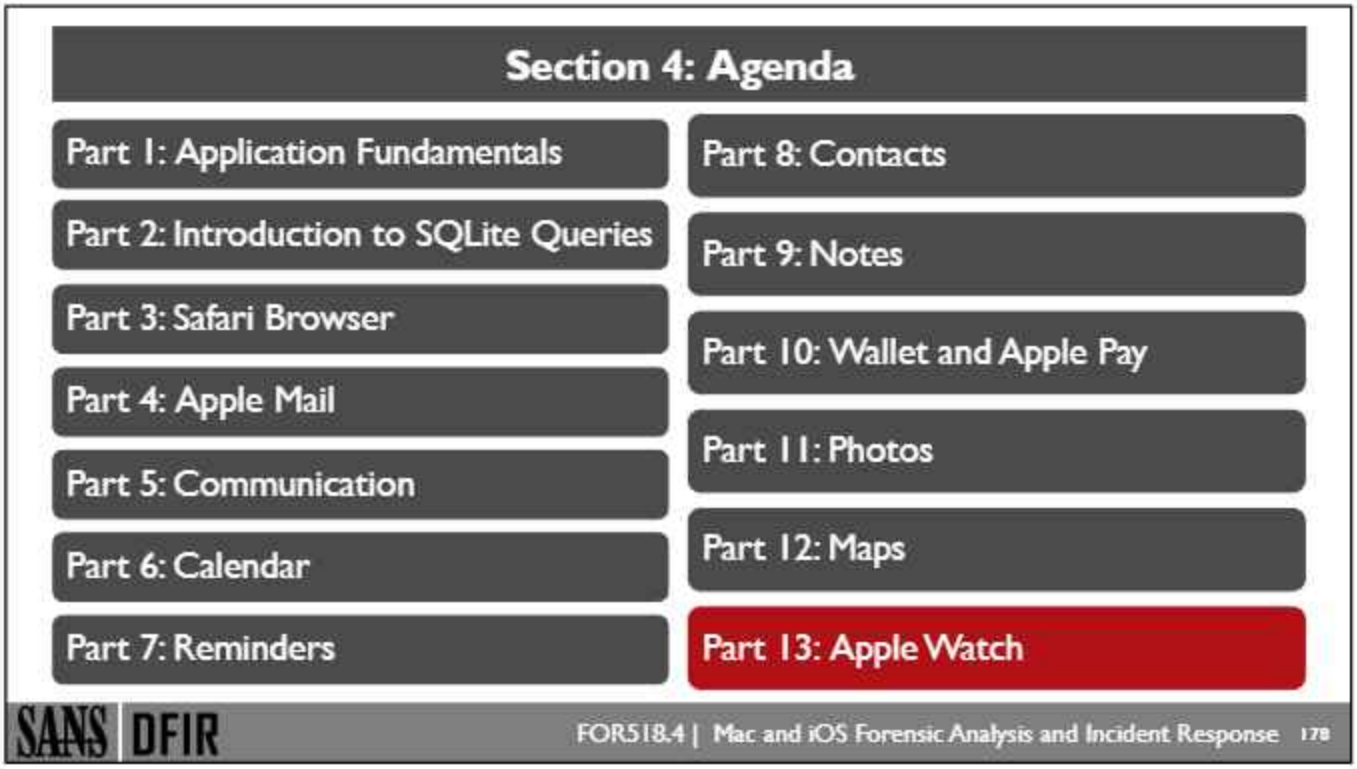
The first step is to extract the BLOB (if it needs to be extracted). Save this data to a file. The protobuf data above is being decoded by the protoc utility from Google's Protobuf utilities. Another option to decode is by using CyberChef (https://gchq.github.io/CyberChef/).

```
brew install protobuf
protoc --decode_raw < [BLOB]
```

Highlighted above in the protoc output, are a set of coordinates that can be converted to decimal latitude/longitude pairs using a converter or hex editor. There may be many pairs of coordinates strewn about the protobuf contents, so do not assume you know the significance of them without knowing the structure of the protobuf.

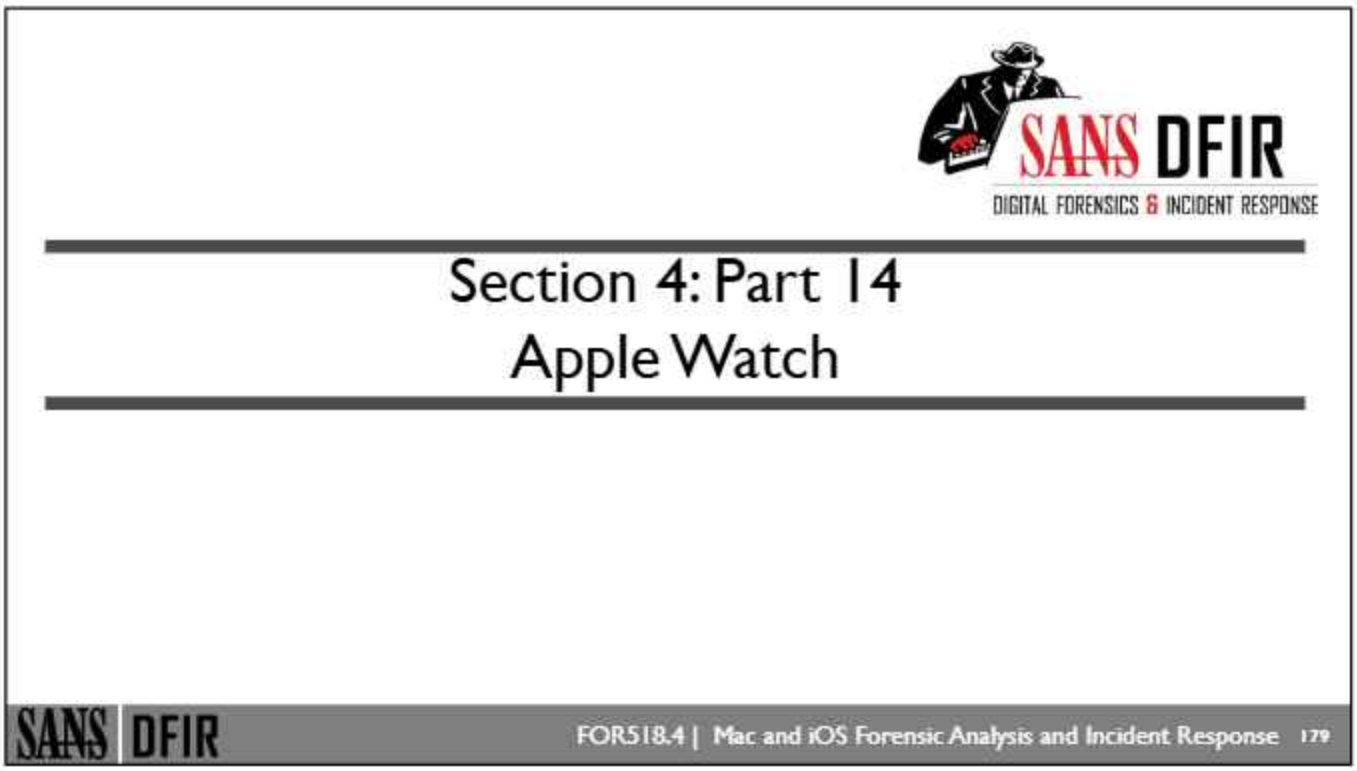
Other items contained in these files may be UUIDs, text, reviews, links to photos, address information, etc.

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards



This page intentionally left blank.

Apple Watch

Usually paired with iPhone, proxy device

Wi-Fi, GPS, NFC, Bluetooth, Cellular

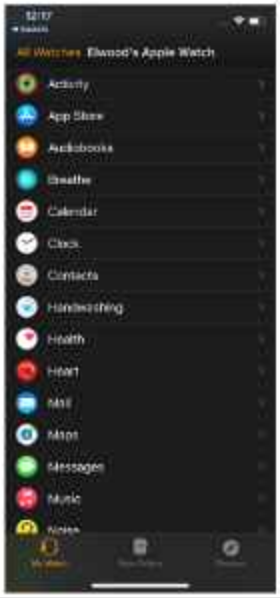
"System in Package" Chip

WatchOS (iOS-based)

Carousel Interface

WatchKit Apps

Watch App on iPhone

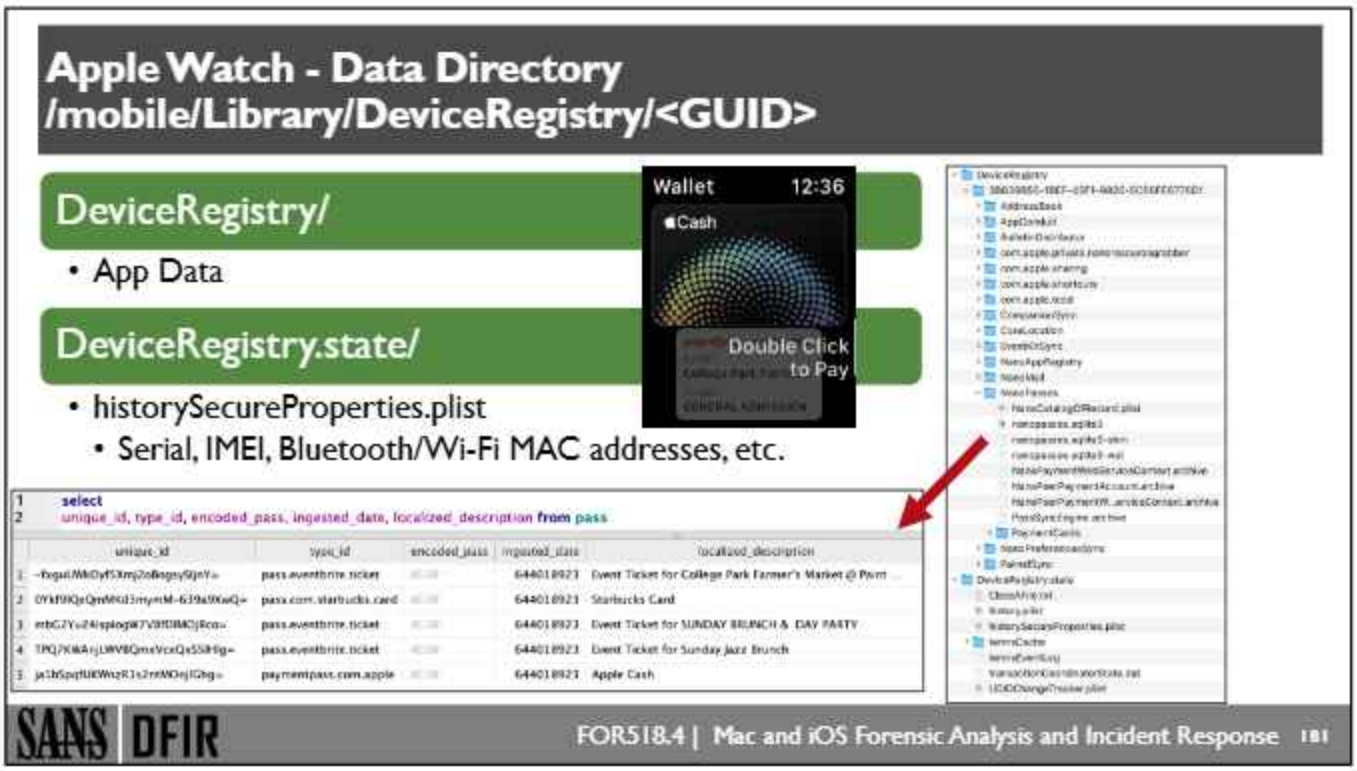


The Apple Watch is Apple's wearable technology, or "Smart Watch". It can be paired with an iPhone (not an iPod or iPad) or not paired at all.

The Apple Watch runs WatchOS. Research shows this may be a lite modified version of iOS.

Applications on the watch include many of those on the iPhone. The Watch can be thought of as an extension of the iPhone.

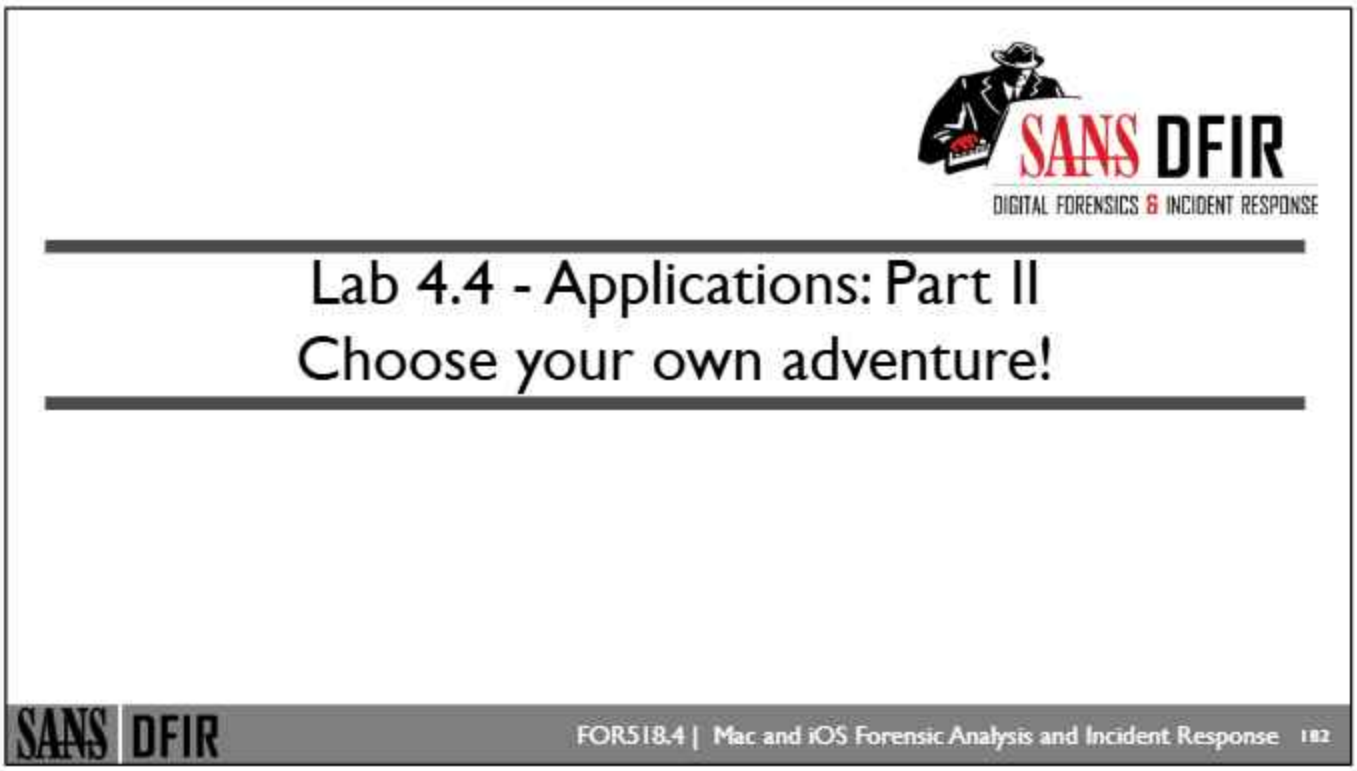
© 2022 Sarah Edwards



Most of the data associated with the Apple Watch is stored in the `/mobile/Library/DeviceRegistry/` directory. This data is mostly redundant data found on the device itself. Shown above is an example of the Watch Wallet database.

Identifying information can be found in the `DeviceRegistry.state/` directory in plist files.

© 2022 Sarah Edwards



This page intentionally left blank.

© 2022 Sarah Edwards