

518.4

Application Data Analysis

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.



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>

Course Agenda

Section : Mac and iOS Essentials

Section 2: System Triage and File Systems

Section 3: Log Analysis, User Data & System Configuration

Section 4: Application Data Analysis

Section 5: Advanced Analysis Topics

Section 6: Mac Forensic Challenge

This page intentionally left blank.

Application Data Analysis

This page intentionally left blank.

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 13: Apple Watch

Part 7: Reminders

This page intentionally left blank.

Section 4: Part I

Application Fundamentals

This page intentionally left blank.

App Fundamentals

File System Locations

iOS Snapshots

Permissions

Most Recently Used (MRUs)

Configured Accounts

App Testing and Analysis

This page intentionally left blank.

App Preference Files

Application Configurations

- Bundle ID/Reverse DNS Filename – net.whatsapp.WhatsApp.plist, com.apple.Maps.plist

macOS

- ~/Library/Preferences/
- ~/Library/Containers/.../<bundle_id>/.../Preferences/

iOS Backup

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

iOS Physical

- /private/var/mobile/Containers/.../<bundle_id>/Preferences/
- /private/var/mobile/Preferences

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.

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_STOREDEDEFAULTS_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/App
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



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!

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.

Application Caches

Cache.db SQLite Database and/or Other Cached Files

macOS

- ~/Library/Caches/
- ~/Library/Containers/.../<bundle_id>/.../Cache/

iOS Backup

- Probably little to none; cached items rarely get backed up

iOS Physical

- /private/var/mobile/Containers/.../<bundle_id>/Library/Caches/
- /private/var/mobile/Library/Cache/<bundle_id>

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.

Example Cache.db File: Uber Application

```

1 select cfurl_cache_response.entry_ID,request_key,time_stamp, receiver_data
2 from cfurl_cache_response
3 left outer join cfurl_cache_receiver_data on cfurl_cache_response.entry_ID = cfurl_cache_receiver_data.entry_ID
4 where cfurl_cache_response.Entry_ID = 613

```

entry_ID	request_key	time_stamp	receiver_data
613	https://cn-sjc1.uber.com/rt/geocoding/reverse?latitude=47.44236897086387&longitude=-122.300106501819&lang...	2016-03-12 22:54:02	{"latitude":4...

Mode: Text

Import Export as NULL

```

{"latitude":47.443301,"longitude":-122.3016229,"shortAddress":"12 Departures Dr","longAddress":"12 Departures Dr, SeaTac, WA 98158, USA","addressComponents":[{"long_name":"12","short_name":"12","types":["street_number"]}, {"long_name":"Departures Drive","short_name":"Departures Dr","types":["route"]}, {"long_name":"SeaTac","short_name":"SeaTac","types":["locality","political"]}, {"long_name":"King County","short_name":"King County","types":["administrative_area_level_2","political"]}, {"long_name":"Washington","short_name":"WA","types":["administrative_area_level_1","political"]}, {"long_name":"United States","short_name":"US","types":["country","political"]}, {"long_name":"98158","short_name":"98158","types":["postal_code"]}]}

```

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.

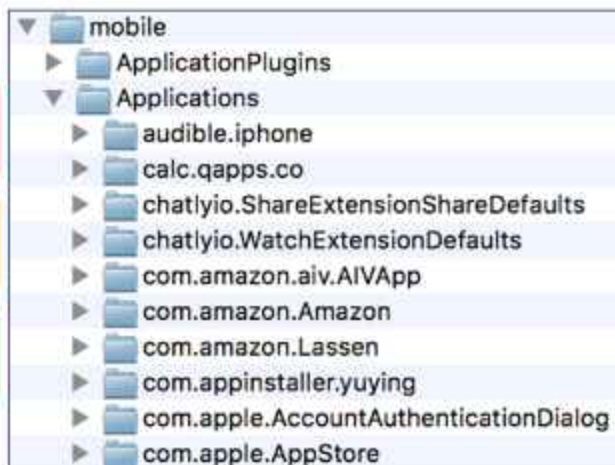
The example above shows the `Cache.db` (`com.ubercab.UberClient/Cache.db`) file for the Uber application. Any applications that use location services may be good candidates for location data. The Uber `cache.db` database contains HTTP URL queries containing location data for where the user was requesting 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.

iOS Containers: Disk Structure App Data

/private/var/mobile/Containers/

/Bundle/Application/<GUID>/

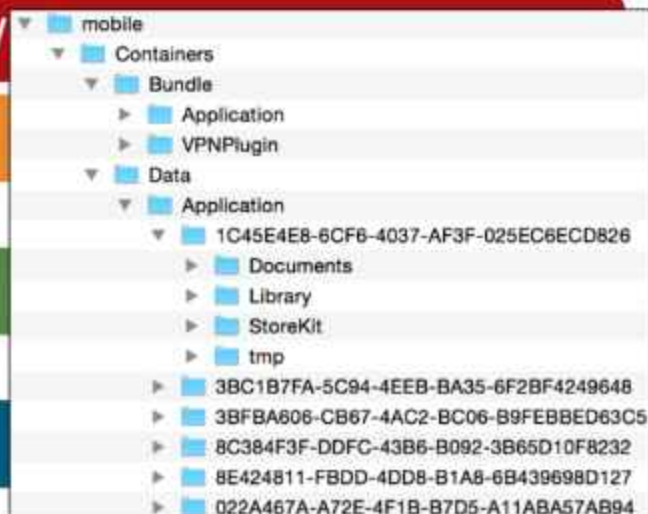
- Contains the Application Binary

/Data/Application/<GUID>/

- App Data

/Shared/AppGroup/<GUID>/

- Shared App Data



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>

iOS Third-Party App Directory Structure



App Binary Package

Documents - Databases

Library

- Cookies, Caches/WebKit, Preferences, App Analytics

Key	Type	Value
Root	Dictionary	(4 items)
MCMMetadataContentClass	Number	2
MCMMetadataIdentifier	String	net.whatsapp.WhatsApp
MCMMetadataInfo	Dictionary	(2 items)
com.apple.MobileInstallation.ContentProtectionClass	Number	0
com.apple.MobileInstallation.GroupContainerIDs	Array	(2 items)
Item 0	String	group.com.facebook.family
Item 1	String	group.net.whatsapp.WhatsApp.shared
MCMMetadataUUID	String	3166F790-D188-4A09-895C-1F84D8F886FF

SANS DFIR

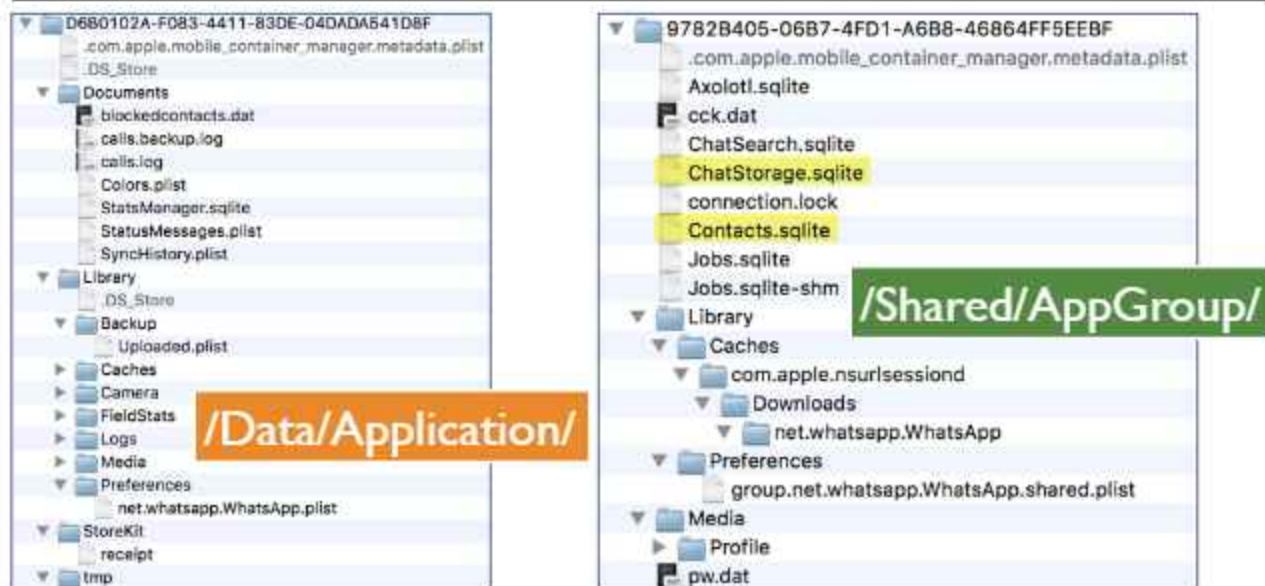
FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 14

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.

App Data Directory and Shared Data Directory



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 15

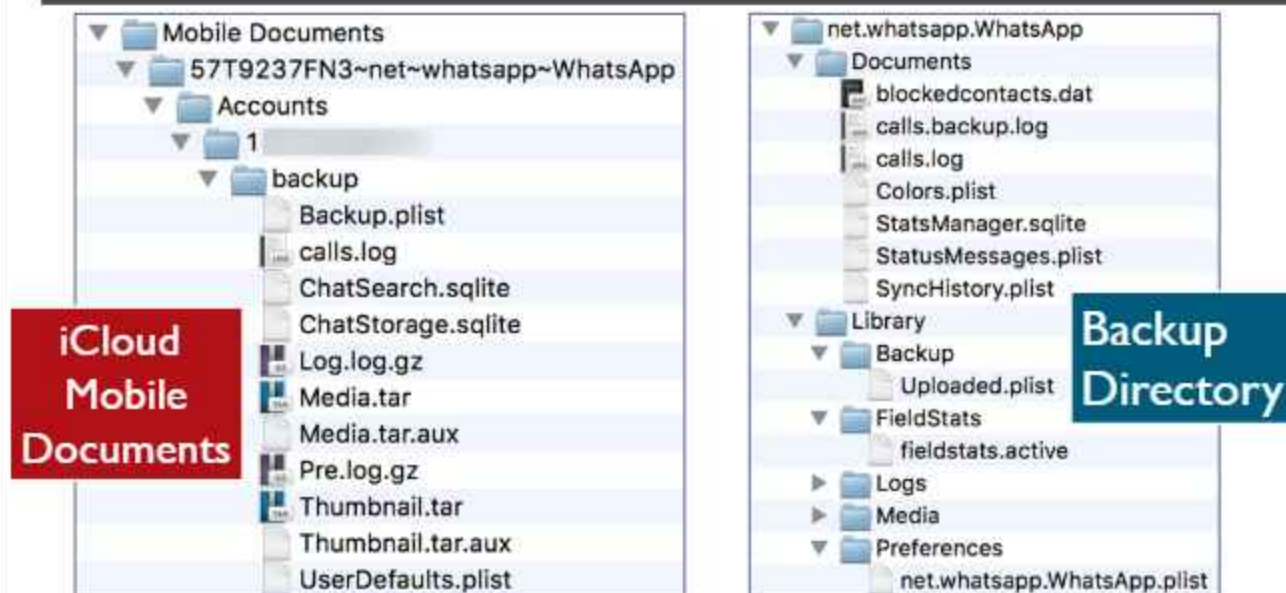
The differences between the `/Data/Application/` and the `/Shared/AppGroup/` directories in the app containers can be quite different and it's always worth looking at both to find the information you are seeking.

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.

iCloud and Backup Data Directories



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 16

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.

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.

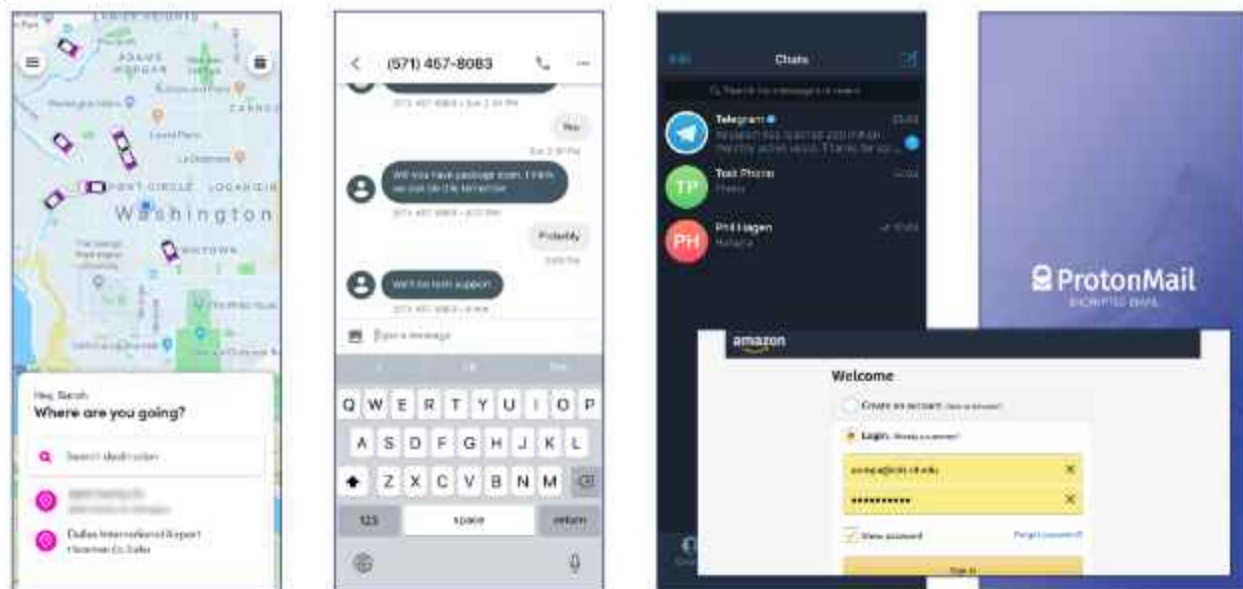
These screen captures are only available on physical acquisitions.

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>

iOS Application Snapshot Examples



SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 18

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)

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



Spectacle requires that the Accessibility API be enabled

Would you like to open the Universal Access preferences so that you can turn on "Enable access for assistive devices"?

Stop Spectacle

Open Universal Access Preferences

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.

iOS Application Privacy Settings: TCC.db [1]



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 20

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 Glypse, 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/>

iOS Application Privacy Settings: TCC.db [2]

```
1 select service, client, auth_value,
2 datetime(last_modified,'unixepoch') as last_modified
3 from access
```

	service	client	auth_value	last_modified
1	kTCCServiceUbiquity	com.microsoft.onenote	2	2018-09-18 01:39:08
2	kTCCServiceUbiquity	com.microsoft.skydrive	2	2018-09-18 01:39:08
3	kTCCServiceCamera	com.amazon.Amazon	2	2018-09-18 01:39:08
4	kTCCServiceMicrophone	com.dropcam.Dropcam	0	2018-09-18 01:39:08
5	kTCCServiceUbiquity	com.tinyspeck.chatylo	2	2018-09-18 01:39:08
6	kTCCServiceUbiquity	com.apple.MailCompositionService	2	2018-09-18 01:39:08
7	kTCCServiceUbiquity	com.apple.mobilemail	2	2018-09-18 01:39:08
8	kTCCServiceLiverpool	com.apple.mobilenotes	2	2018-09-18 01:39:08
9	kTCCServiceUbiquity	com.apple.mobilenotes	2	2018-09-18 01:39:08
10	kTCCServiceUbiquity	com.getdropbox.Dropbox	2	2018-09-18 01:39:08
11	kTCCServiceLiverpool	com.apple.mobilesafari	2	2018-09-18 01:39:08
12	kTCCServiceAddressBook	com.apple.mobilesafari	0	2018-09-18 01:39:08
13	kTCCServiceUbiquity	com.apple.mobilesafari	2	2018-09-18 01:39:08
14	kTCCServiceLiverpool	com.apple.news	2	2018-09-18 01:39:08
15	kTCCServiceUbiquity	com.apple.news	2	2018-09-18 01:39:08
16	kTCCServiceUbiquity	com.apple.Preferences	2	2018-09-18 01:39:08
17	kTCCServiceUbiquity	net.whatsapp.WhatsApp	2	2018-09-18 01:39:08

```
1 select distinct service from access
```

	service
1	kTCCServiceAddressBook
2	kTCCServiceBluetoothAlways
3	kTCCServiceCalendar
4	kTCCServiceCamera
5	kTCCServiceExposureNotification
6	kTCCServiceFaceID
7	kTCCServiceFacebook
8	kTCCServiceKeyboardNetwork
9	kTCCServiceLiverpool
10	kTCCServiceMicrophone
11	kTCCServiceMotion
12	kTCCServicePhotos
13	kTCCServiceSiri
14	kTCCServiceTwitter
15	kTCCServiceUbiquity
16	kTCCServiceWebKitIntelligentTrackingPrevention

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/>

macOS Application Privacy Settings: User's TCC.db

macOS 11+:
auth_value replaces
allowed column
0=Unallowed
2=Allowed

id	service	client	allowed	indirectObjectIdentifier	lastModified
13	kTCCServiceLiverpool	com.apple.systempreferences	1	UNUSED	2018-11-30 00:29:41
14	kTCCServiceLiverpool	com.apple.Safari	1	UNUSED	2018-11-30 00:36:18
15	kTCCServiceUtility	com.getdropbox.dropbox	1	UNUSED	2018-11-30 00:47:55
16	kTCCServiceAppleEvents	com.xmoya.desktopsyncmac...	1	com.apple.systemevents	2018-11-30 01:00:12
17	kTCCServiceAppleEvents	com.xmoya.desktopsyncmac...	1	com.apple.finder	2018-11-30 01:06:27
18	kTCCServiceLiverpool	com.apple.shocks	1	UNUSED	2018-11-30 02:09:32
19	kTCCServiceCamera	com.objective-see.iiid	1	UNUSED	2018-11-30 02:11:54
20	kTCCServiceAppleEvents	com.apple.systemevents	1	com.apple.systemevents	2018-11-30 02:38:29
21	kTCCServiceUtility	com.apple.reminders	1	UNUSED	2018-12-01 16:16:54
22	kTCCServiceLiverpool	com.apple.Note	1	UNUSED	2018-12-01 17:17:51
23	kTCCServiceUtility	com.apple.Safari	1	UNUSED	2018-12-01 18:35:45
24	kTCCServiceUtility	com.apple.Photos	1	UNUSED	2018-12-03 01:09:40
25	kTCCServiceAppleEvents	com.TechSmith.Snagit2019	1	com.apple.Safari	2018-12-03 01:39:38
26	kTCCServiceAddressBook	com.apple.EventKit	1	UNUSED	2018-12-09 15:23:39
27	kTCCServiceMicrophone	com.logmein.GoToMeeting	1	UNUSED	2018-12-17 22:56:41
28	kTCCServiceCamera	com.logmein.GoToMeeting	1	UNUSED	2018-12-17 22:57:18
29	kTCCServiceUtility	com.apple.Chat	1	UNUSED	2018-12-24 16:35:16
30	kTCCServiceLiverpool	com.apple.Wave	1	UNUSED	2018-12-24 18:06:48

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.

macOS Application Privacy Settings: System TCC.db



macOS 11+:
auth_value replaces
allowed column
0=Unallowed
2=Allowed

```

1 select
2 service, client, allowed, indirect_object_identifier,
3 datetime(last_modified, 'unixepoch') as last_modified
4 from tccdb;
    
```

	service	client	allowed	indirect_object_identifier	last_modified
1	KTCCServicePostEvent	com.apple.system.Spotlight	1	UNUSED	2018-11-30 01:19:59
2	KTCCServiceAccessibility	com.apple.system.Spotlight	1	UNUSED	2018-11-30 01:19:59
3	KTCCServicePostEvent	com.getdropbox.dropbox	1	UNUSED	2018-11-30 01:28:55
4	KTCCServiceAccessibility	com.getdropbox.dropbox	1	UNUSED	2018-11-30 03:11:10
5	KTCCServiceAccessibility	com.vimeo.vuim	1	UNUSED	2018-11-30 02:36:48
6	KTCCServiceAccessibility	com.techsmith.screenshoter2019	1	UNUSED	2018-12-01 21:26:41
7	KTCCServiceAccessibility	com.techsmith.screenshoter	1	UNUSED	2018-12-01 21:26:41
8	KTCCServicePostEvent	com.TechSmith.Sight2019	1	UNUSED	2018-12-01 21:27:00
9	KTCCServicePostEvent	com.techsmith.screenshoter2019	1	UNUSED	2018-12-01 01:37:13
10	KTCCServiceAccessibility	net.sourceforge.sitedeveloper	1	UNUSED	2018-12-01 03:03:11
11	KTCCServicePostEvent	net.sourceforge.sitedeveloper	1	UNUSED	2018-12-01 03:01:08
12	KTCCServiceSystemPolicyAllFiles	com.apple.dt.Xcode	1	UNUSED	2018-12-09 15:17:13
13	KTCCServiceSystemPolicyAllFiles	com.macropoint.iExplorer	1	UNUSED	2018-12-09 15:17:21
14	KTCCServicePostEvent	com.logmein.goToMeeting	1	UNUSED	2018-12-17 22:57:11
15	KTCCServiceAccessibility	com.logmein.goToMeeting	1	UNUSED	2018-12-17 22:57:11
16	KTCCServiceSystemPolicyAllFiles	com.apple.Terminal	1	UNUSED	2018-12-30 03:06:01

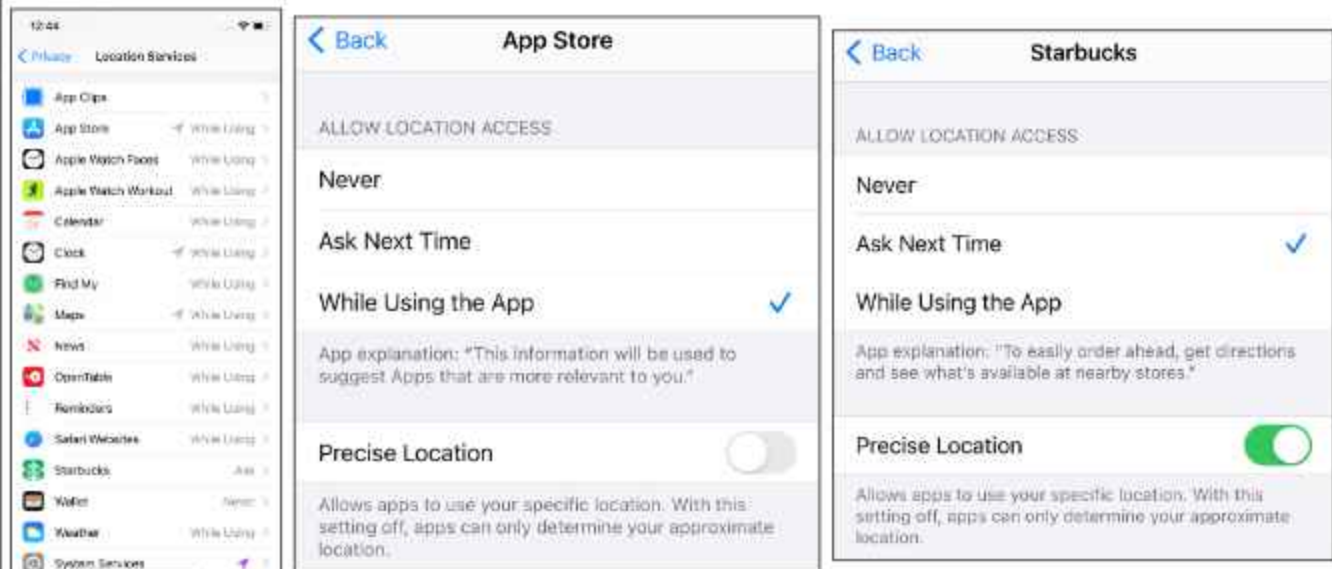
The macOS privacy settings found in TCC.db are taken from System Preferences | Security & Privacy | Privacy.

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".

iOS Applications Using Location Services: clients.plist



SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 24

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.

iOS App Location Service Authorizations: clients.plist

com.apple.AppStore	Dictionary	(12 items)
Bundled	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)
com.starbucks.mystarbucks	Dictionary	(7 items)
Whitelisted	Boolean	0
Bundled	String	com.starbucks.mystarbucks
SupportedAuthorizationMask	Number	3
RemoteUsage	Dictionary	(0 items)
Registered	String	
CorrectiveCompensationEnabled	Number	1
Executable	String	

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

CorrectiveCompensationEnabled:
'Precise Location' (iOS 14):
1: Enabled (or no key listed)
2: Disabled

SANS DFIR

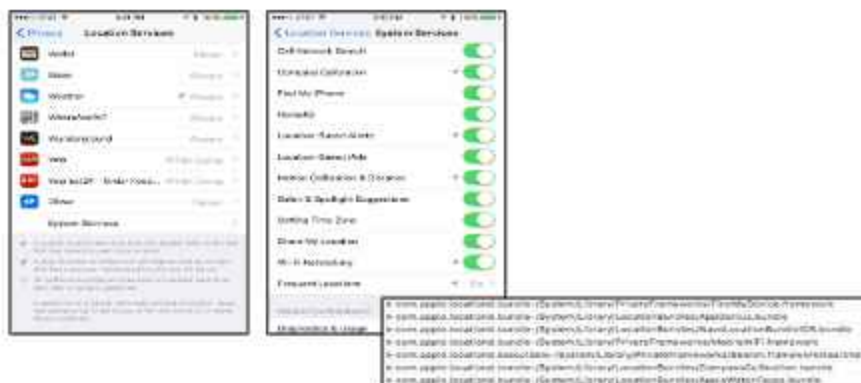
FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 25

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.



macOS App Location Service Authorizations: clients.plist

The screenshot shows the macOS Security & Privacy window with the 'Privacy' tab selected. Under 'Location Services', 'Weather' and 'Calendar.app' are listed as authorized apps. To the right, a plist file viewer displays the contents of the 'clients.plist' file, showing two dictionaries for 'com.apple.weather' and 'com.apple.iCal'.

Key	Type	Value
com.apple.weather	Dictionary	(9 items)
Hide	Number	0
Whitelisted	Boolean	NO
LocationTimeStopped	Number	508,209,366.563485
BundleID	String	com.apple.weather
BundlePath	String	/System/Library/CoreServices/Weather.app/Contents/Resources/Weather.app
Registered	String	/System/Library/CoreServices/Weather.app/Contents/Resources/Weather.app
Executable	String	/System/Library/CoreServices/Weather.app/Contents/Resources/Weather.app
Requirement	String	identifier "com.apple.weather" and anchor app
Authorized	Boolean	YES
com.apple.locationd.bundleID	Dictionary	(6 items)
com.apple.iCal	Dictionary	(6 items)
Hide	Number	0
Whitelisted	Boolean	NO
BundleID	String	com.apple.iCal
Registered	String	
BatchEnabled	Boolean	NO
Authorized	Boolean	YES
Requirement	String	identifier "com.apple.iCal" and anchor app
LocationTimeStarted	Number	508,295,306.279874

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.

Most Recently Used (MRUs)

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

- Each Office application has its own plist with its MRU's

macOS: `~/Library/Preferences/com.apple.finder.plist`

- Recent folders

macOS: `~/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.ApplicationRecentDocuments/`

- `<bundle_id>.sfl` (i.e., `com.apple.textedit.sfl2`)
- Recent documents per application

macOS: `~/Library/Application Support/com.apple.sharedfilelist/`

- `com.apple.LSSharedFileList.RecentApplications.sfl2`
- `com.apple.LSSharedFileList.RecentDocuments.sfl2`
- `com.apple.LSSharedFileList.RecentHosts.sfl2`
- `com.apple.LSSharedFileList.RecentServers.sfl2`

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

√ Root	Dictionary	(16 items)
> file:///Users/oempa/Dropbox/FOR518_F03_01_POWERPOINTS/F04_01_EDITS/FOR518_4_F03_01.pptx	Dictionary	(3 items)
> file:///Users/oempa/Dropbox/FOR518_F03_01_POWERPOINTS/FOR518_3_F03_01.pptx	Dictionary	(3 items)
> file:///Users/oempa/Dropbox/FOR518_F03_01_POWERPOINTS/FOR518_1_F03_01.pptx	Dictionary	(3 items)
> file:///Users/oempa/Dropbox/FOR518_F03_01_POWERPOINTS/FOR518_4_F03_01.pptx	Dictionary	(3 items)
√ file:///Users/oempa/Dropbox/GettingSpookyWithAPOLLO.pptx	Dictionary	(3 items)
kBookmarkDataKey	Data	<626F6F6BC802000000000410300000006CA7B73E>
kUUIDKey	String	8DC511C6-360E-4901-A280-1851F8978784
kLastUsedDateKey	Date	2020-12-10T14:23:45Z
> file:///Users/oempa/Dropbox/FOR518_F03_01_POWERPOINTS/F04_01_EDITS/FOR518_2_F03_01.pptx	Dictionary	(3 items)
> file:///Users/oempa/Dropbox/FOR518_F03_01_POWERPOINTS/F04_01_EDITS/FOR518_5_F03_01.pptx	Dictionary	(3 items)

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!

Bookmark Data

0	626F6F6B	C8020000	00000410	30000000	5CA7873E	276DF1FC	85A43F14	043EA320	book»	0	\Bd>'m0.05? >E
32	D540F7C0	72A099CB	F805FFD1	CE364F48	C4010000	04000000	03030000	00080028	'@~zrjôÄ~'-@60Hf		(
64	05000000	01010000	55736572	73000000	05000000	01010000	6F6F6D70	61000000	Users	oampa	
96	07000000	01010000	44726F70	626F7800	1C000000	01010000	47657474	696E6753	Dropbox	GettingS	
128	706F6F6B	79576974	6841504F	4C4C4F2E	70707478	10000000	01060000	10000000	pookyWithAPOLLO.pptx		
160	20000000	30000000	40000000	08000000	04030000	9E580000	00000000	08000000	0 @	ûX	
192	04030000	DD9B0300	00000000	08000000	04030000	67201000	00000000	08000000	>ô	g	
224	04030000	C4A61000	00000000	10000000	01060000	7C000000	8C000000	9C000000	f¶	l à û	
256	AC000000	08000000	00040000	41C2A659	68000000	18000000	01020000	01000000	-	A-¶Yh	
288	00000000	0F000000	00000000	00000000	00000000	08000000	04030000	02000000			
320	00000000	04000000	03030000	F5010000	08000000	01090000	66696C65	3A2F2F2F	1	file:///	
352	0C000000	01010000	4D616369	6E746F73	68204844	08000000	04030000	0020458C	Macintosh HD	Eâ	
384	D0010000	08000000	00040000	41C1DE44	80000000	24000000	01010000	45323245	-	AjfidA \$	E22E
416	46393636	20334334	41203442	45332D39	4342432D	38313635	44304435	41433746	F966-3C4A-4BE3-9CBC-8165D0D5AC7F		
448	18000000	01020000	81000000	01000000	EF130000	01000000	00000000	00000000	A	ô	
480	01000000	01010000	2F000000	00000000	01050000	CC000000	FEFFFFFF	01000000	/	Ã	---
512	00000000	10000000	04100000	64000000	00000000	05100000	BC000000	00000000	d	o	
544	10100000	E4000000	00000000	40100000	D4000000	00000000	02200000	00010000	€	@ ' w	
576	00000000	05200000	20010000	00000000	10200000	30010000	00000000	11200000		0	
608	64010000	00000000	12200000	44010000	00000000	13200000	54010000	00000000	d	D T	
640	20200000	90010000	00000000	30200000	8C010000	00000000	01C00000	04010000	ê	0 °	¿
672	00000000	11C00000	20000000	00000000	12C00000	14010000	00000000	10000000	¿	¿	-
704	04000000	00000000									

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:

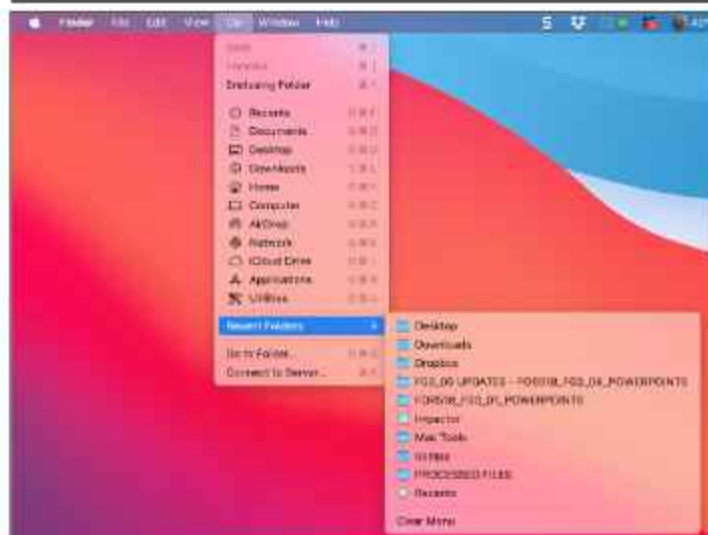
- Strings making up the file path (/Users/oampa/Dropbox/GettingSpookyWithAPOLLO.pptx)
- Volume Name (Macintosh HD)
- Volume GUID (E22EF966-3C4A-4BE3-9CBC-8165D0D5AC7F)

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	<826F6F6B8020000000004103>
name	String	Dropbox
Item 1	Dictionary	(2 items)
file-bookmark	Data	<826F6F6B8020000000004103>
name	String	Mac Tools
Item 2	Dictionary	(2 items)
file-bookmark	Data	<826F6F6B8020000000004103>
name	String	Downloads
Item 3	Dictionary	(2 items)
file-bookmark	Data	<826F6F6B8020000000004103>
name	String	Impactor
Item 4	Dictionary	(2 items)
file-bookmark	Data	<826F6F6B8020000000004103>
name	String	Desktop
Item 5	Dictionary	(2 items)
file-bookmark	Data	<826F6F6B8020000000004103>
name	String	oompa
Item 6	Dictionary	(2 items)
file-bookmark	Data	<826F6F6B8020000000004103>
name	String	Reports
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.

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
```

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 31

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.

NSKeyedArchiver Formatted Binary Plist Files [I]

The image shows two Xcode windows. The top window, titled 'com.apple.LSSharedFileList.RecentApplications.sfl', shows a blank document with a green callout box that says 'Default Xcode View'. The bottom window, titled 'com.apple.LSSharedFileList.RecentApplications.plist', shows the same file with a red arrow pointing to it from the 'Default Xcode View' box. A green callout box on the left says 'Change ".sfl" to ".plist"'. The bottom window displays a table of the plist's contents.

Key	Type	Value
▼ Root	Dictionary	(4 items)
\$version	Number	100,000
▶ \$objects	Array	(76 items)
\$sarchiver	String	NSKeyedArchiver
▶ \$top	Dictionary	(0 items)

SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 32

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
- \$sarchiver with the value “NSKeyedArchiver”
- \$top

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	NSURL
Item 18	Data	<626f666b 00000410 30000000 0>
Item 19	Dictionary	(2 items)
NS.keys	Array	(2 items)
NS.objects	Array	(2 items)
Item 20	Dictionary	(2 items)
\$classname	String	SFURLItem
\$classes	Array	(2 items)
Item 0	String	SFURLItem
Item 1	String	NSURL
Item 21	Dictionary	(1 item)
order	Number	-319
Item 22	Dictionary	(1 item)
NS.uuidbytes	Data	<c7f6a33 8b48478e b83a3384 e4c48242>
Item 23	String	Terminal
Item 24	Dictionary	(2 items)
Item 25	String	file:///Applications/Utilities/Terminal.app/
Item 26	Data	<626f666b 00000410 30000000 0>
Item 27	Dictionary	(1 item)
order	Number	-320
Item 28	Dictionary	(1 item)
NS.uuidbytes	Data	<6229995c 879435a b6012f00 a7c1852>
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.

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 00
▼ Item 19	Dictionary	(2 items)
▼ NS.keys	Array	(0 items)
▼ NS.objects	Array	(0 items)
▼ Item 20	Dictionary	(2 items)
\$classname	String	SFListItem
▼ \$classes	Array	(2 items)
Item 0	String	SFListItem
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 00
▼ 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

NSKeyedArchiver Formatted Binary Plist Files [3]

```
{
  "version" => 100000
  "objects" => [
    0 => "null"
    1 => {
      "NS.keys" => [
        0 => <CFKeyedArchiverUID 0x7ff370401b00 [0x7fff760dd390]>{value = 2}
        1 => <CFKeyedArchiverUID 0x7ff370401b00 [0x7fff760dd390]>{value = 3}
        2 => <CFKeyedArchiverUID 0x7ff370401b00 [0x7fff760dd390]>{value = 4}
      ]
      "NS.objects" => [
        0 => <CFKeyedArchiverUID 0x7ff370401c60 [0x7fff760dd390]>{value = 5}
        1 => <CFKeyedArchiverUID 0x7ff370401c60 [0x7fff760dd390]>{value = 6}
        2 => <CFKeyedArchiverUID 0x7ff370401c60 [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"
      ]
    }
  ]
}
```

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.

```

{
  "$version" => 100000
  "$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"
      ]
    }
  ]
}

```


NSKeyedArchiver Formatted Binary Plist Files [4]

Start at \$stop / "root"

```
"$stop" => {  
  "root" => <CFKeyedArchiverUID 0x7ff370406bb0 [0x7fff760dd390]>{value = 1}  
}
```

Find Object "1"

```
{  
  "$version" => 100000  
  "$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  
  }  
}
```

To start parsing these plist files, we need to start at the `$stop` 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.

```

{
    "$version" => 100000
    "$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
}

```

NSKeyedArchiver Formatted Binary Plist Files [5]

\$classname = Data Type
(May be NS data type or custom)

```
9 => {  
  "$classname" => "NSDictionary"  
  "$classes" => [  
    0 => "NSDictionary"  
    1 => "NSObject"  
  ]  
}
```

```
20 => {  
  "$classname" => "SFLListItem"  
  "$classes" => [  
    0 => "SFLListItem"  
    1 => "NSObject"  
  ]  
}
```

```
17 => {  
  "$classname" => "NSURL"  
  "$classes" => [  
    0 => "NSURL"  
    1 => "NSObject"  
  ]  
}
```

In the previous example, the `$classname` 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 0x7ff370401db8 [0x7fff760dd390]>{value = 7}
  ]
  "NS.objects" => [
    0 => <CFKeyedArchiverUID 0x7ff370401e18 [0x7fff760dd390]>{value = 8}
  ]
  "sclass" => <CFKeyedArchiverUID 0x7ff370401d18 [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 = 31}
    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 0x7ff370402118 [0x7fff760dd390]>{value = 69}
  ]
  "sclass" => <CFKeyedArchiverUID 0x7ff3704021b8 [0x7fff760dd390]>{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” ...

```

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}
}

```


MacMRU Python Script: github.com/mac4n6/macMRU-Parser

Older Plists, SFL* Files, Microsoft Office Files, and Spotlight Shortcuts

Run on live system or mounted image

Parses Bookmark and Alias BLOBs

```
bit:macMRU-Parser-master oompa$ python macMRU.py ~/Library/
##### MacMRU Parser v1.1 #####
=====
Parsing: /Users/oompa/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.iCloudItems.sfl
Cannot open file: /Users/oompa/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.iCloudItems.sfl
=====
Parsing: /Users/oompa/Library/Application Support/com.apple.sharedfilelist/com.apple.LSSharedFileList.RecentApplications.sfl
Max number of recent items in this plist: 10
[Item Number: 0 | Order: -134.0] Name: 'Microsoft Word.app' (URL: 'file:///Applications/Microsoft%20Word.app/')
[Item Number: 1 | Order: -136.0] Name: 'Microsoft Error Reporting.app' (URL: 'file:///Applications/Microsoft%20PowerPoint.app/')
[Item Number: 2 | Order: -140.0] Name: 'Console.app' (URL: 'file:///Applications/Utilities/Console.app/')
[Item Number: 3 | Order: -130.0] Name: 'osxpmem.app' (URL: 'file:///private/var/folders/n7/vnfzcl55443_qg8zp2cwz1880000gn/T/AppleInternal/AppleInternal/Microsoft%20PowerPoint.app/')
[Item Number: 4 | Order: -137.0] Name: 'Microsoft PowerPoint.app' (URL: 'file:///Applications/Microsoft%20PowerPoint.app/')
[Item Number: 5 | Order: -131.0] Name: 'Microsoft Excel.app' (URL: 'file:///Applications/Microsoft%20Excel.app/')
[Item Number: 6 | Order: -138.0] Name: 'Preview.app' (URL: 'file:///Applications/Preview.app/')
[Item Number: 7 | Order: -129.0] Name: 'osxpmem.app' (URL: 'file:///Users/oompa/Downloads/osxpmem.app/')
[Item Number: 8 | Order: -133.0] Name: 'Epoch Converter.app' (URL: 'file:///Applications/Epoch%20Converter.app/')
[Item Number: 9 | Order: -132.0] Name: 'System Preferences.app' (URL: 'file:///Applications/System%20Preferences.app/')
=====
```

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 44

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>.

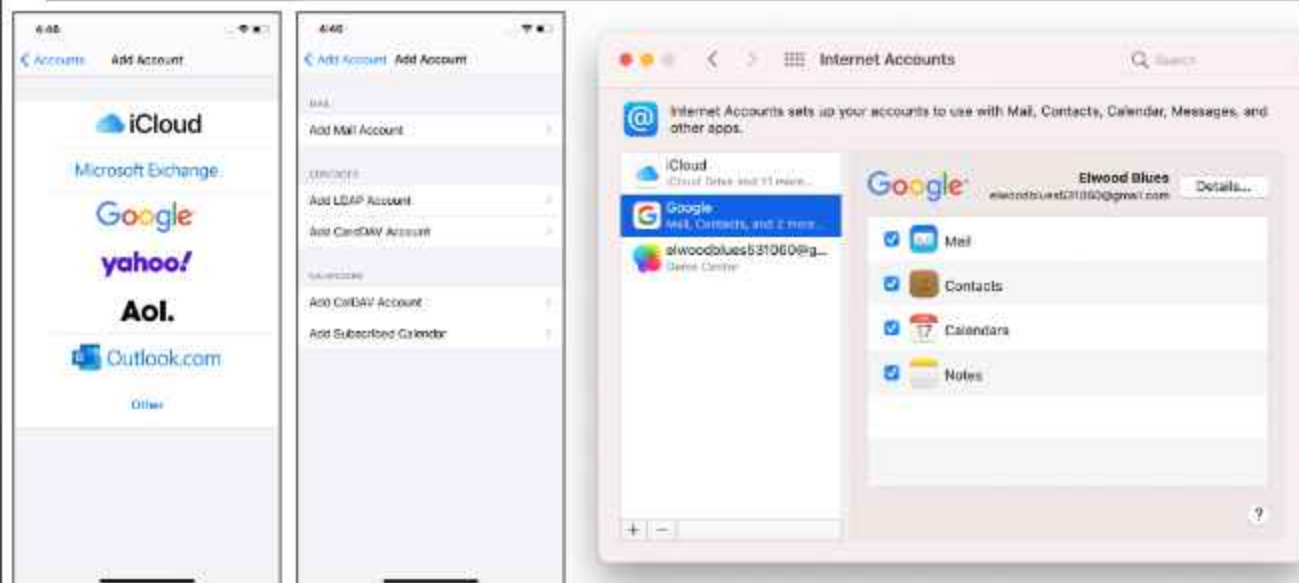
References:

<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>

Accounts – com.apple.accountsd



SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 45

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.

1	SELECT						
2	ZACCOUNT.Z_PK,ZACCOUNTTYPE,ZACCOUNTTYPEDESCRIPTION,ZACCOUNT.ZIDENTIFIER,ZACCOUNT.ZUSERNAME,						
3	ZACCOUNT.ZACCOUNTDESCRIPTION,ZACCOUNTPROPERTY.ZKEY,ZACCOUNTPROPERTY.ZVALUE						
4	FROM ZACCOUNTPROPERTY						
5	LEFT JOIN ZACCOUNT ON ZACCOUNT.Z_PK == ZACCOUNTPROPERTY.ZOWNER						
6	LEFT JOIN ZACCOUNTTYPE ON ZACCOUNT.ZACCOUNTTYPE == ZACCOUNTTYPE.Z_PK						
7	order by ZACCOUNT.ZIDENTIFIER						
	Z_PK	ZACCOUNTTYPEDESCRIPTION	ZIDENTIFIER	ZUSERNAME	ZACCOUNTDESCRIPTION	ZKEY	ZVALUE
1	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	handles	BL00
2	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	account-info	BL00
3	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	apple-id	BL00
4	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	self-handle	BL00
5	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	invitation-context	BL00
6	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	auth-token-receipt-date	BL00
7	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	add-timestamp	BL00
8	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	email-address	BL00
9	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	profile-id	BL00
10	4	Messages	05E7C21C-8F6E-425B-981C-9E396C14C178	ehwoodblues331060@gmail.com	NULL	bundleRef	BL00
11	17	IMAP	38330C55-06CA-45F7-93A1-738737D5931D	NULL	NULL	EmailAliases	BL00
12	17	IMAP	38330C55-06CA-45F7-93A1-738737D5931D	NULL	NULL	SendingAccountIdentifier	BL00
13	17	IMAP	38330C55-06CA-45F7-93A1-738737D5931D	NULL	NULL	SSLEnabled	BL00
14	17	IMAP	38330C55-06CA-45F7-93A1-738737D5931D	NULL	NULL	AuthenticationScheme	BL00
15	17	IMAP	38330C55-06CA-45F7-93A1-738737D5931D	NULL	NULL	Hostname	BL00
16	17	IMAP	38330C55-06CA-45F7-93A1-738737D5931D	NULL	NULL	AllowsInsecureAuthentication	BL00
17	17	IMAP	38330C55-06CA-45F7-93A1-738737D5931D	NULL	NULL	PortNumber	BL00
18	17	IMAP	38330C55-06CA-45F7-93A1-738737D5931D	NULL	NULL	SSLIsDirect	BL00
19	6	iCloud	3FEFF8EF-3CE8-4591-A839-7C3E398F7227	ehwoodblues331060@gmail.com	iCloud	altDSID	BL00
20	6	iCloud	3FEFF8EF-3CE8-4591-A839-7C3E398F7227	ehwoodblues331060@gmail.com	iCloud	cloudDocsMigrationComplete	BL00
21	6	iCloud	3FEFF8EF-3CE8-4591-A839-7C3E398F7227	ehwoodblues331060@gmail.com	iCloud	accountClass	BL00

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.

Internet Accounts - Accounts3.sqlite | Accounts4.sqlite [2]

ZKEY	ZVALUE
EmailAliases	BLOB
SendingAccountIdentifier	BLOB
SSLEnabled	BLOB
AuthenticationScheme	BLOB
Hostname	BLOB
AllowsInsecureAuthentication	BLOB
PortNumber	BLOB
SSLUseDirect	BLOB
autoDSID	BLOB
cloudDocsMigrationComplete	BLOB
accountClass	BLOB
personID	BLOB
regionInfo	BLOB
familyEligible	BLOB
notesMigrated	BLOB

```

0000 02 70 6c 69 73 74 30 30 d4 01 02 03 04 05 06 07 bpl i st 00 .....
0010 0a 58 24 76 65 72 73 69 6f 0e 39 24 61 72 63 68 , X5ver si onY3arch
0020 69 76 65 72 54 24 74 6f 70 58 24 6f 62 6a 65 63 i ver T3topX5obj ec
0030 74 73 12 00 01 86 40 5f 10 0f 4e 53 4b 65 79 65 ts ..... ASKeye
0040 64 41 72 63 68 69 76 65 72 d1 08 09 34 72 6f 6f dArchi ver... Troo
0050 74 80 01 af 10 10 0b 0c 12 1f 20 21 22 23 24 25 t ..... !' #EN
0060 29 13 34 15 36 3c 55 24 6e 75 6c 6c d2 0d 0e 0f 3456<U5mal l...
0070 11 5a 4e 53 2e 6f 62 6a 65 63 74 73 56 24 63 6c .2NS. obj ect sV5ci
0080 01 73 73 a1 10 80 02 80 0f d3 13 0d 0e 14 19 1e ass .....
0090 57 4e 53 2e 6b 65 79 73 a4 15 16 17 18 80 03 80 W5S. keys.....
00a0 04 80 05 80 06 a4 1a 1b 1c 1b 80 07 80 08 80 09 ..... [ D xpl ayName
00b0 04 80 05 80 06 a4 1a 1b 1c 1b 80 07 80 08 80 09 Y1 sEnabledEmail
00c0 01 64 04 12 00 73 73 65 73 59 49 73 50 72 69 6d AddresseY1 sPri m
00d0 01 64 04 12 00 73 73 65 73 59 49 73 50 72 69 6d aryY1 wood Bl ues
00e0 09 d2 0d 0e 26 11 2f 00 0a 80 0f d3 13 0d 0e .....&.....0
00f0 2a 2e 1e a3 16 2c 2d 80 0e 0b 80 0c a3 1b 30 .....Enai l A
0100 09 d2 0d 0e 26 11 2f 00 0a 80 0f d3 13 0d 0e .....el woodBl ues53
0110 1b 80 08 80 0d 80 08 80 0e 5c 4e 61 69 6c 41 2060@gmail. com. 7
0120 64 64 72 65 73 73 59 49 73 44 65 68 6c 65 74 89: Z5ci assnameX5
0130 5f 10 1b 65 6c 77 6f 6f 64 62 6c 75 65 73 73 ci asses\NSCi ct i o
0140 21 30 36 30 40 67 6d 61 69 6c 2e 63 6f 6d 6e 27 nary. 9: XNSChj ect
0150 38 39 3a 5a 24 63 6c 61 73 73 6e 61 6d 65 58 24 .78->W5Array. <
0160 63 6c 61 73 73 65 73 5c 4e 53 4f 62 6a 65 63 74 .....S. ) . 2. 7. i
0170 6e 61 72 79 a2 39 3b 58 4e 53 4f 62 6a 65 63 74 .....L. C. S. f. i. q. i.
0180 d2 37 38 3d 5e 57 4e 53 41 72 72 61 79 a2 3d 3b .....
0190 00 08 00 11 00 1a 00 24 00 29 00 32 00 37 00 49 .....
01a0 00 4c 00 51 00 53 00 66 00 6c 00 71 00 7c 00 83 .....
01b0 00 85 00 87 00 89 00 90 00 98 00 9d 00 9f 00 a1 .....
01c0 00 a3 00 a5 00 aa 00 ac 00 ae 00 b0 00 b2 00 b4 .....
01d0 00 c0 00 ca 00 c9 00 e3 00 f0 00 f1 00 f6 00 f8 .....
01e0 00 fa 00 fc 01 03 01 07 01 09 01 0b 01 0d 01 11 .....
01f0 01 13 01 15 01 17 01 19 01 26 01 30 01 4e 01 53 .....&. 0. N. S
0200 01 5e 01 67 01 74 01 77 01 80 01 85 01 8d 00 00 .....A. g. t. w.....?..
0210 00 00 00 00 02 01 00 00 00 00 00 00 00 00 00
0220 00 00 00 00 00 00 00 00 00 00 00 00 00 01 90

```

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`.

	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

0000 62 70 6c 69 73 74 30 30 d4 01 02 03 04 05 06 07 0010 0a 58 24 76 65 72 73 69 6f 6e 59 24 61 72 63 68 0020 69 76 65 72 54 24 74 6f 70 58 24 6f 62 6a 65 63 0030 74 73 12 00 01 86 a0 5f 10 0f 4e 53 4b 65 79 65 0040 64 41 72 63 68 69 76 65 72 d1 08 09 54 72 6f 6f 0050 74 80 01 af 10 10 0b 0c 12 1f 20 21 22 23 24 25 0060 29 33 34 35 36 3c 55 24 6e 75 6c 6c d2 0d 0e 0f 0070 11 5a 4e 53 2e 6f 62 6a 65 63 74 73 56 24 63 6c 0080 61 73 73 a1 10 80 02 80 0f d3 13 0d 0e 14 19 1e 0090 57 4e 53 2e 6b 65 79 73 a4 15 16 17 18 80 03 80 00a0 04 80 05 80 06 a4 1a 1b 1c 1b 80 07 80 08 80 09 00b0 08 80 0e 5b 44 69 73 70 6c 61 79 4e 61 6d 65 00c0 59 45 15 6e 61 62 6c 65 64 5e 45 6d 61 69 6c 00d0 41 64 64 7e 15 73 73 65 73 59 49 73 50 72 69 6d 00e0 61 72 79 5c 45 6f 6f 6f 64 20 42 6c 75 65 73 00f0 09 d2 0d 0e 26 11 a1 70 0a 80 0f d3 13 0d 0e 0100 2a 2e 1e a3 16 2c 2d 80 0e 0b 80 0c a3 1b 30 0110 1b 80 08 80 0d 80 08 80 0e 5c 4e 15 61 69 6c 41 0120 64 64 72 65 73 73 59 49 73 44 65 66 6e 6c 74 0130 5f 10 1b 65 6c 77 6f 6f 64 62 6c 75 65 73 0140 31 30 36 30 40 67 6d 61 69 6c 2e 63 6f 6d 6e 57 0150 38 39 3a 5a 24 63 6c 61 73 73 6e 61 6d 65 58 24 0160 63 6c 61 73 73 65 73 5c 4e 53 44 69 63 74 69 6f 0170 6e 61 72 79 a2 39 3b 58 4e 53 4f 62 6a 65 63 74 0180 d2 37 38 3d 3e 57 4e 53 41 72 72 61 79 a2 3d 3b 0190 00 08 00 11 00 1a 00 24 00 29 00 32 00 37 00 49 01a0 00 4c 00 51 00 53 00 66 00 6c 00 71 00 7c 00 83 01b0 00 85 00 87 00 89 00 90 00 98 00 9d 00 9f 00 a1 01c0 00 a3 00 a5 00 aa 00 ac 00 ae 00 b0 00 b2 00 b4 01d0 00 c0 00 ca 00 d9 00 e3 00 f0 00 f1 00 f6 00 f8 01e0 00 fa 00 fc 01 03 01 07 01 09 01 0b 01 0d 01 11 01f0 01 13 01 15 01 17 01 19 01 26 01 30 01 4e 01 53 0200 01 5e 01 67 01 74 01 77 01 80 01 85 01 8d 00 00 0210 00 00 00 00 02 01 00 00 00 00 00 00 3f 00 00 0220 00 00 00 00 00 00 00 00 00 00 00 00 01 90	BLOB	bpl ist 00..... .X\$versl ony\$arch l ver T\$topX\$obj ec ts.....NSKeye d archl ver...Troo t.....i"#S\$%) 3456<U\$nu lZNS. obj ect sV\$cl ass..... WNS.keys..... [Di spl ayName Yl sEnabl edV\$enai l Addres ses Yl sPri m ary\El wood Bl ues&.....0\Enai l A ddres s Yl sDefaul t ...el woodbl ues53 1060@gmail l..con.7 89:Z\$cl assnameX\$ cl asses\NSDi ctio nary.9: XNSCbj ect ..78=>WNSArray.=-\$.).2.7.1 ..L.C.S.f.l.q.l..&.O.N.5 ..^..g.t.w.....?
---	------	---

Mac and iOS App Testing and Analysis

On Analysis Host (Mac)

- iproxy (libimobiledevice)
- sqlite3
- SQLite Viewer
- Xcode
- Virtual Machines
- etc.

On Jailbroken Device

- SSH/SCP
- sqlite3
- Jonathan Levin's Binpack
- fsmon
- cda
- etc.

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>

Lab 4.1

iOS Snapshots, App Permissions, and MRUs

This page intentionally left blank.

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 13: Apple Watch

Part 7: Reminders

This page intentionally left blank.

Section 4: Part 2

Introduction to SQLite Queries

This page intentionally left blank.

SQLite Databases

Database Files

- Main Database – *.db, *.sqlite, *.sqlitedb, *.storedata, or no file extension!
- Write Ahead Log – *.wal – May contain additional database transactions
- Shared Memory – *.shm

Tables and Columns

Analysis

- Forensic Viewers vs. Non-Forensic Viewers
- GUI Viewers vs. Command-Line Utilities
- Database Coalescing

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/>

Example Database – TCC.db

Table: access											
id	service	client	client_type	allowed	prompt_count	count	policy_id	subject_object_identifier_type	subject_object_identifier	subject_object_uuid_identity	flags
1	ITCServiceUtility	com.apple.weather	0	1	1	0/0	0/0	0/0	0/0	0	154555510
2	ITCServiceUtility	com.apple.ChessKit.MultipleGameOfFifteen	0	1	1	0/0	0/0	0/0	0/0	0	154555519
3	ITCServiceUtility	com.apple.Accounts	0	1	1	0/0	0/0	0/0	0/0	0	154555541
4	ITCServiceUtility	com.apple.ScriptEditor2	0	1	1	0/0	0/0	0/0	0/0	0	154555548
5	ITCServiceUtility	com.apple.Work.Masters	0	1	1	0/0	0/0	0/0	0/0	0	154555545
6	ITCServiceUtility	com.apple.Previews	0	1	1	0/0	0/0	0/0	0/0	0	154555545
7	ITCServiceUtility	com.apple.Work.Keynote	0	1	1	0/0	0/0	0/0	0/0	0	154555548
8	ITCServiceUtility	com.apple.Bleut	0	1	1	0/0	0/0	0/0	0/0	0	154555545
9	ITCServiceUtility	com.apple.Work.Pages	0	1	1	0/0	0/0	0/0	0/0	0	154555545
10	ITCServiceUtility	com.apple.mail	0	1	1	0/0	0/0	0/0	0/0	0	154555555
11	ITCServiceUtility	com.apple.QuartzComposer	0	1	1	0/0	0/0	0/0	0/0	0	154555555
12	ITCServiceUtility	com.apple.TextEdit	0	1	1	0/0	0/0	0/0	0/0	0	154555558
13	ITCServiceUtility	com.apple.systempreferences	0	1	1	0/0	0/0	0/0	0/0	0	154555578
14	ITCServiceUtility	com.apple.Safari	0	1	1	0/0	0/0	0/0	0/0	0	154555573
15	ITCServiceUtility	com.apple.systempreferences	0	1	1	0/0	0/0	0/0	0/0	0	154555575
16	ITCServiceUtility	com.apple.systempreferences	0	1	1	0/0	0/0	0/0	0/0	0	154555580
17	ITCServiceUtility	com.apple.systempreferences	0	1	1	0/0	0/0	0/0	0/0	0	154555580
18	ITCServiceUtility	com.apple.systempreferences	0	1	1	0/0	0/0	0/0	0/0	0	154555572
19	ITCServiceUtility	com.apple.systempreferences	0	1	1	0/0	0/0	0/0	0/0	0	154555584
20	ITCServiceUtility	com.apple.systempreferences	0	1	1	0/0	0/0	0/0	0/0	0	154555580
21	ITCServiceUtility	com.apple.systempreferences	0	1	1	0/0	0/0	0/0	0/0	0	154555584
22	ITCServiceUtility	com.apple.systempreferences	0	1	1	0/0	0/0	0/0	0/0	0	154555584
23	ITCServiceUtility	com.apple.Safari	0	1	1	0/0	0/0	0/0	0/0	0	154555585

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.

SELECT Statement – Using DB Browser for SQLite

1	select * from access											
	service	client	client_type	allowed	prompt_count	csreq	policy_id	object_idenf	indirect_object_idenf	object_code	flags	last_modified
1	KTCCServiceUbiquity	com.apple.weather	0	1	1	BLD0	NULL	NULL	UNUSED	NULL	0	1543635532
2	KTCCServiceUbiquity	com.apple.CloudDocs.Mobi...	0	1	1	BLD0	NULL	NULL	UNUSED	NULL	0	1543538919
3	KTCCServiceUbiquity	com.apple.Automator	0	1	1	NULL	NULL	NULL	UNUSED	NULL	0	1543535941
4	KTCCServiceUbiquity	com.apple.ScriptEditor2	0	1	1	NULL	NULL	NULL	UNUSED	NULL	0	1543535945

1	select				
2	service, client, allowed, indirect_object_identifier, last_modified				
3	from access				
	service	client	allowed	indirect_object_identifier	last_modified
24	KTCCServiceUbiquity	com.apple.Photos	1	UNUSED	1543799326
25	KTCCServiceAppleEvents	com.TechSmith.Sagit2019	1	com.apple.Safari	1543801178
26	KTCCServiceAddressBook	com.evernote.Evernote	1	UNUSED	1544369016
27	KTCCServiceMicrophone	com.logmein.GoToMeeting	1	UNUSED	1545087401
28	KTCCServiceCamera	com.logmein.GoToMeeting	1	UNUSED	1545087438
29	KTCCServiceUbiquity	com.apple.iChat	1	UNUSED	1545665718

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>

SELECT Statement – Using sqlite3

```

Sarahs-Air:com.apple.TCC compas$ sudo tcpdump -i en0
SQLite version 3.26.0 2018-12-01 12:34:55
Enter ".help" for usage hints.
sqlite> select * from access;
WTCCServiceUbiquity|com.apple.weather|0|1|1??
||||UNUSED||0|1543535532
WTCCServiceUbiquity|com.apple.CloudDocs.MobileDocumentsFileProvider|0|1|1??
||||UNUSED||0|1543535919
WTCCServiceUbiquity|com.apple.Automation|0|1|1|1|UNUSED||0|1543535941
WTCCServiceUbiquity|com.apple.ScriptEditor2|0|1|1|1|UNUSED||0|1543535945
WTCCServiceUbiquity|com.apple.iWork.Numbers|0|1|1|1|UNUSED||0|1543535945
WTCCServiceUbiquity|com.apple.Preview|0|1|1|1|UNUSED||0|1543535945
WTCCServiceUbiquity|com.apple.iWork.Keynote|0|1|1|1|UNUSED||0|1543535945
WTCCServiceUbiquity|com.apple.iBooksX|0|1|1|1|UNUSED||0|1543535945
WTCCServiceUbiquity|com.apple.iWork.Pages|0|1|1|1|UNUSED||0|1543535965
WTCCServiceUbiquity|com.apple.ncui|0|1|1|1|UNUSED||0|1543535999
WTCCServiceUbiquity|com.apple.QuickTimePlayerX|0|1|1|1|UNUSED||0|1543535999
WTCCServiceUbiquity|com.apple.TextEdit|0|1|1|1|UNUSED||0|1543535999
WTCCServiceLiverpool|com.apple.systempreferences|0|1|1|1??
||||
aTCCServiceLiverpool|com.apple.Safari|0|1|1|1??
||||UNUSED||0|15
aTCCServiceUbiquity|com.getdropbox.dropbox|0|1|1|1|UNUSED||0|15
||||service||client

```

service	client	client_type	allowed	prompt_count	csreq
kTCCServiceUbiquity	com.apple.weather	B	1	1	??
kTCCServiceUbiquity	com.apple.CloudDo	B	1	1	??
kTCCServiceUbiquity	com.apple.Autonat	B	1	1	
kTCCServiceUbiquity	com.apple.ScriptE	B	1	1	
kTCCServiceUbiquity	com.apple.iWork.N	B	1	1	
kTCCServiceUbiquity	com.apple.Preview	B	1	1	
kTCCServiceUbiquity	com.apple.iWork.K	B	1	1	
kTCCServiceUbiquity	com.apple.iBooksA	B	1	1	

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>

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|1|UNUSED||0|1543535941  
kTCCServiceUbiquity|com.apple.ScriptEditor2|0|1|1|1|UNUSED||0|1543535945  
kTCCServiceUbiquity|com.apple.iWork.Numbers|0|1|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             1  
kTCCServiceUbiquity  com.apple.ScriptE  0             1          1             1  
kTCCServiceUbiquity  com.apple.iWork.N  0             1          1             1  
kTCCServiceUbiquity  com.apple.Preview  0             1          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>

Timestamp Conversion and Column Renaming

```
1 select
2 service, client, allowed, indirect_object_identifier,
3 datetime(last_modified,'unixepoch','localtime')
4 from access
```

	service	client	allowed	indirect_object_identifier	datetime(last_modified,'unixepoch','localtime')
24	kTCCServiceUbiquity	com.apple.Photos	1	UNUSED	2018-12-02 20:08:46
25	kTCCServiceAppleEvents	com.TechSmith.Snagit2019	1	com.apple.Safari	2018-12-02 20:39:38
26	kTCCServiceAddressBook	com.evernote.Evernote	1	UNUSED	2018-12-08 10:23:36
27	kTCCServiceMicrophone	com.logmein.GoToMeeting	1	UNUSED	2018-12-17 17:56:41
28	kTCCServiceCamera	com.logmein.GoToMeeting	1	UNUSED	2018-12-17 17:57:18

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

	service	client	allowed	indirect_object_identifier	Last Modified
24	kTCCServiceUbiquity	com.apple.Photos	1	UNUSED	2018-12-02 20:08:46
25	kTCCServiceAppleEvents	com.TechSmith.Snagit2019	1	com.apple.Safari	2018-12-02 20:39:38

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 unwieldy; 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

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>

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 19:02:39
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

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>

ORDER BY Clause

```

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
10 order by 'Last Modified' asc;

```

	service	client	Allowed	indirect_object_identifier	Last Modified
1	KTCCServiceUbiquity	com.apple.weather	Allowed	UNUSED	2018-11-29 18:52:12
2	KTCCServiceUbiquity	com.apple.CloudDocs.MobileDocumentaFileP...	Allowed	UNUSED	2018-11-29 18:58:39
3	KTCCServiceUbiquity	com.apple.Automator	Allowed	UNUSED	2018-11-29 18:59:01
4	KTCCServiceUbiquity	com.apple.ScriptEditor2	Allowed	UNUSED	2018-11-29 18:59:06

```

5   when 1 then 'Allowed'
6   end Allowed,
7   indirect_object_identifier,
8   datetime(last_modified, 'unixepoch', 'localtime') as 'Last Modified'
9 from access
10 order by 'Last Modified' desc;

```

	service	client	Allowed	indirect_object_identifier	Last Modified
1	KTCCServiceAppleEvents	com.TechSmith.SnagIt2019	Allowed	com.google.Chrome	2019-04-28 17:08:40
2	KTCCServiceLiverpool	com.apple.news	Allowed	UNUSED	2019-04-10 23:34:56
3	KTCCServiceUbiquity	com.apple.news	Allowed	UNUSED	2019-04-10 23:34:59
4	KTCCServiceAppleEvents	Amr/bin/bscript	Allowed	com.apple.Tinder	2019-03-31 09:02:14

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 63

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>

WHERE Clause and LIKE Function

```
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
10 where "Last Modified" like "%2019-%"
11 order by "Last Modified"
```

	service	client	Allowed	indirect_object_identifier	Last Modified
1	kTCCServiceAppleEvents	com.blackbagtech.BlackLight	Allowed	com.apple.finder	2019-01-12 19:02:39
2	kTCCServiceAddressBook	com.microsoft.onenote.mac	Allowed	UNUSED	2019-01-18 12:15:28
3	kTCCServiceAddressBook	com.microsoft.Word	Allowed	UNUSED	2019-01-18 12:16:09
4	kTCCServiceAddressBook	com.microsoft.Powerpoint	Allowed	UNUSED	2019-01-18 12:17:15
5	kTCCServiceAppleEvents	com.logitech.presenter	Allowed	com.apple.systempreferences	2019-01-28 10:16:50
6	kTCCServiceAppleEvents	com.logitech.presenter	Allowed	com.apple.Safari	2019-01-28 10:21:03

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>

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

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>

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 13: Apple Watch

Part 7: Reminders

This page intentionally left blank.

Section 4: Part 3

Safari Browser

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/

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



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 69

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.

Safari Browser – History.db [2]

```

1 SELECT
2 HISTORY_VISITS.ID AS 'HISTORY_ITEM_ID', DATETIME(HISTORY_VISITS.VISIT_TIME+57607220,'UNIXEPOCH') AS 'VISIT TIME',
3 HISTORY_ITEMS.URL,HISTORY_ITEMS.VISIT_COUNT,HISTORY_VISITS.TITLE,HISTORY_VISITS.ORIGIN,HISTORY_VISITS.LOAD_SUCCESSFUL,HISTORY_VISITS.REDIRECT_SOURCE,HISTORY_VISITS.REDIRECT_DESTINATION
4 FROM HISTORY_ITEMS
5 LEFT OUTER JOIN HISTORY_VISITS ON HISTORY_ITEMS.ID = HISTORY_VISITS.HISTORY_ITEM
6 ORDER BY 'VISIT TIME'

```

HISTORY_ITEM_ID	VISIT TIME	url	visit_count	title	origin	load_successful	redirect_source	redirect_destination
1	2021-05-13 14:14:11	https://www.apple.com/uk/macos/whats-new	1		0	1		
2	2021-05-13 14:15:51	https://www.apple.com/uk/macos/whats-new	2	MacOS Big Sur - What's new	0	1		
3	2021-05-13 14:16:07	https://www.apple.com/uk/macos/whats-new	3	MacOS Big Sur - What's new	0	1		
4	2021-05-13 14:17:07	https://www.apple.com/uk/macos/whats-new	4	MacOS Big Sur - What's new	0	1		
5	2021-05-13 14:17:14	https://www.apple.com/uk/macos/whats-new	5	MacOS Big Sur - What's new	0	1		
6	2021-05-13 14:17:25	https://www.apple.com/uk/macos/whats-new	6	MacOS Big Sur - What's new	0	1		
7	2021-05-13 14:18:11	https://www.apple.com/uk/macos/whats-new	7	MacOS Big Sur - What's new	0	1		
8	2021-05-13 14:18:51	https://www.apple.com/uk/macos/whats-new	8	MacOS Big Sur - What's new	0	1		
9	2021-05-13 14:19:20	https://www.apple.com/uk/macos/whats-new	9	MacOS Big Sur - What's new	0	1		
10	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	10	MacOS Big Sur - What's new	0	1		
11	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	11	MacOS Big Sur - What's new	0	1		
12	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	12	MacOS Big Sur - What's new	0	1		
13	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	13	MacOS Big Sur - What's new	0	1		
14	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	14	MacOS Big Sur - What's new	0	1		
15	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	15	MacOS Big Sur - What's new	0	1		
16	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	16	MacOS Big Sur - What's new	0	1		
17	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	17	MacOS Big Sur - What's new	0	1		
18	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	18	MacOS Big Sur - What's new	0	1		
19	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	19	MacOS Big Sur - What's new	0	1		
20	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	20	MacOS Big Sur - What's new	0	1		
21	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	21	MacOS Big Sur - What's new	0	1		
22	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	22	MacOS Big Sur - What's new	0	1		
23	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	23	MacOS Big Sur - What's new	0	1		
24	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	24	MacOS Big Sur - What's new	0	1		
25	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	25	MacOS Big Sur - What's new	0	1		
26	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	26	MacOS Big Sur - What's new	0	1		
27	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	27	MacOS Big Sur - What's new	0	1		
28	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	28	MacOS Big Sur - What's new	0	1		
29	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	29	MacOS Big Sur - What's new	0	1		
30	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	30	MacOS Big Sur - What's new	0	1		
31	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	31	MacOS Big Sur - What's new	0	1		
32	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	32	MacOS Big Sur - What's new	0	1		
33	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	33	MacOS Big Sur - What's new	0	1		
34	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	34	MacOS Big Sur - What's new	0	1		
35	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	35	MacOS Big Sur - What's new	0	1		
36	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	36	MacOS Big Sur - What's new	0	1		
37	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	37	MacOS Big Sur - What's new	0	1		
38	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	38	MacOS Big Sur - What's new	0	1		
39	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	39	MacOS Big Sur - What's new	0	1		
40	2021-05-13 14:19:32	https://www.apple.com/uk/macos/whats-new	40	MacOS Big Sur - What's new	0	1		

Origin = 0: Visited on
This Device
Origin = 1: Visited on
Another iCloud
Connected System

70

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.

Safari Browser – Sessions – macOS - LastSession.plist

[illegible]

SessionState – Binary plist containing tab history, may be encrypted or not.



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.

√ 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	<0000000262706C6973743030D2010203045E52>
> AncestorTabIdentifiers	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>
> AncestorTabIdentifiers	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

Safari Browser – Sessions – iOS – BrowserState.db

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

```

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 10:38:58	FC00D163-4360-401D-8427-68785A57A886	Sketchley Park - Wikitrends	https://en.m.wikipedia.org/wiki/Sketchley_Park	0	0	0	0
2	2021-05-13 10:38:40	E7275877-4564-4C80-8EF8-CBF30030F086	A subreddit for cute and kashly pictures	https://www.reddit.com/r/aww/	1	0	0	0
3	2021-05-13 13:18:32	651DB862-8AC9-4F9C-9D3F-55B0B001C3E	Starbucks Coffee Company	https://ssp.starbucks.com/	4	0	1	1
4	2021-05-13 13:53:40	06C0C0B5-0523-46AD-8084-8F60B991B000	How can I update my Starbucks.com email address?	https://customerservice.starbucks.com/pages/answers/...	3	0	0	0
5	2021-05-13 12:50:43	1C494D18-6780-4674-8FEE-55C205362148	Apple	https://www.apple.com/	2	0	0	0
6	2021-05-13 12:43:23	C644AC40-1081-4610-5AAS-8EF79A8DA000	zappos - Google Search	https://www.google.com/search/...	0	1	0	0
7	2021-05-13 12:43:18	98183DC8-CAAE-41F7-9288-752F2D80F548	Cubs Downloadable Schedule Chicago Cubs	https://www.mlb.com/cubs/schedule/downloadable...	1	0	0	0
8	2021-05-13 12:43:10	5A28F29F-4909-4558-A0EE-39CE9893937D	flappy - Google Search	https://www.google.com/search/...	1	1	0	0

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 73

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.

```

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	FCD0D5B3-4360-405D-B4Z7-0878A57A80D	Blechney Park - Wikipedia	https://en.m.wikipedia.org/wiki/Blechney_Park	0	0	0	0
2	2021-05-13 10:39:40	E7275877-F564-4C80-BE78-CE3015E1D98	A subreddit for cute and cuddly pictures	https://www.reddit.com/r/aww/	1	0	0	0
3	2021-05-13 13:19:32	851D862-80C9-479C-8D9F-518891E65C3E	Starbucks Coffee Company	https://app.starbucks.com/	4	0	1	0
4	2021-05-13 13:50:40	D9CC0C85-0923-46AD-8E8A-B76D80189D0	How can I update my Starbucks.com email address?	https://customerservice.starbucks.com/app/answers/...	3	0	0	0
5	2021-05-13 13:50:43	3C49AD18-8780-4674-BFE6-5FC053E2148	Apple	https://www.apple.com/	2	0	0	0
6	2021-05-15 12:43:23	C64A4C40-3D81-461D-9A4F-BE78A6DAD60	puppies - Google Search	https://www.google.com/search?...	0	1	0	0
7	2021-05-15 12:43:59	98386DC8-CAAF-41F7-9286-75E7D80F548	Cubs Downloadable Schedule Chicago Cubs	https://www.mlb.com/cubs/schedule/downloadable-...	3	0	0	0
8	2021-05-15 12:43:59	5828F29F-4909-4558-A6EE-39CE9893857D	clippy - Google Search	https://www.google.com/search?...	1	1	0	0

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



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 75

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.

Safari Browser – macOS - Tab Snapshots/Metadata.db

Table: snapshot_metadata

	uuid	date_created	filename	url
Filter	Filter	Filter	Filter	Filter
1	ESA3EF81-A21F-4616-B080-7231AF698ED0	642789689.733873	ESA3EF81-A21F-4616-B080-7231AF698ED0.png	favorites://
2	5CFFB73A-182E-45B3-A28D-686948D30FD7	642790386.868395	5CFFB73A-182E-45B3-A28D-686948D30FD7.png	https://en.wikipedia.org/wiki/Enigma_machine
3	69CFE76F-CB4B-4360-85E6-48F95231E086	642790349.974687	69CFE76F-CB4B-4360-85E6-48F95231E086.png	https://www.reddit.com/r/catsvstechnology/
4	EB4DE3DD-E959-482E-94FE-9DC5164A5063	642790347.897222	EB4DE3DD-E959-482E-94FE-9DC5164A5063.png	https://www.whatsapp.com/download/

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.

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	119551E9-C886-4763-B818-9D0F7ECD6977	MBP-M1	Cats vs. Technology	https://www.reddit.com/r/catsvstechnology/	2021-05-15 00:00:00
2	333F0A90-EA0C-4C87-93D8-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-4C87-93D8-D72DF01D6172	Elwood's iPhone	Bletchley Park - Wikipedia	https://en.m.wikipedia.org/wiki/Bletchley_Park	2021-05-15 12:44:19
4	333F0A90-EA0C-4C87-93D8-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-4C87-93D8-D72DF01D6172	Elwood's iPhone	Apple	https://www.apple.com/	2021-05-15 12:44:19
6	119551E9-C886-4763-B818-9D0F7ECD6977	MBP-M1	Download WhatsApp	https://www.whatsapp.com/download/	2021-05-15 00:00:00
7	119551E9-C886-4763-B818-9D0F7ECD6977	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.


```

1 select
2 cloud_tab_devices.device_uid, 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_uid = cloud_tab_devices.device_uid

```

	device_uid	device_name	title	url	last_modified
1	119351E9-C886-4763-8B1B-900F7ECD6977	MBP-M1	Cats vs. Technology	https://www.reddit.com/r/catsystechnology/	2021-05-15 00:00:00
2	333F0A90-EA0C-4C87-93D8-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-4C87-93D8-D72DF01D6172	Elwood's iPhone	Bletchley Park - Wikipedia	https://en.m.wikipedia.org/wiki/Bletchley_Park	2021-05-15 12:44:19
4	333F0A90-EA0C-4C87-93D8-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-4C87-93D8-D72DF01D6172	Elwood's iPhone	Apple	https://www.apple.com/	2021-05-15 12:44:19
6	119351E9-C886-4763-8B1B-900F7ECD6977	MBP-M1	Download WhatsApp	https://www.whatsapp.com/download/	2021-05-15 00:00:00
7	119351E9-C886-4763-8B1B-900F7ECD6977	MBP-M1	Enigma machine - Wikipedia	https://en.wikipedia.org/wiki/Enigma_machine	2021-05-15 00:00:00

Safari Browser Cache - Cache.db

Contains downloaded cache files (or GUID references to them)

```
1 SELECT
2 CFURL_CACHE_RESPONSE.ENTRY_ID,CFURL_CACHE_RESPONSE.REQUEST_KEY,
3 CFURL_CACHE_RESPONSE.TIME_STAMP,CFURL_CACHE_RECEIVER_DATA.RECEIVER_DATA
4 FROM CFURL_CACHE_RESPONSE
5 LEFT JOIN CFURL_CACHE_RECEIVER_DATA ON CFURL_CACHE_RESPONSE.ENTRY_ID == CFURL_CACHE_RECEIVER_DATA.ENTRY_ID
```

entry_ID	request_key	time_stamp	receiver_data
1	https://configuration.apple.com/...	2021-05-13 20:48:52	<!doctype html>...
2	https://configuration.apple.com/...	2021-05-13 20:48:52	<?xml version="1.0" encoding="UTF-8"?>...
3	https://configuration.apple.com/...	2021-05-13 20:48:52	3022A82A-97D5-4...B-8087D1DCDDC
4	https://configuration.apple.com/...	2021-05-15 15:36:32	B34F257F-6355-48FC-9EE2-A0F2445F5A2E
5	https://configuration.apple.com/...	2021-05-15 15:36:32	<!doctype html>...
6	https://configuration.apple.com/...	2021-05-15 15:36:32	<!doctype html>...
7	https://cdn2.smoot.apple.com/images/...	2021-05-15 16:41:32	EA4F5E17-684F-4AAB-8C1A-3F6CD31A0E46
8	https://cdn2.smoot.apple.com/...	2021-05-15 16:41:32	EA4F5E17-684F-4AAB-8C1A-3F6CD31A0E46
9	https://cdn2.smoot.apple.com/...	2021-05-15 16:41:33	FD1110D9-7678-4...69-8F85A584388
10	https://cdn2.smoot.apple.com/...	2021-05-15 16:41:33	FD1110D9-7678-4...69-8F85A584388
11	https://cdn2.smoot.apple.com/...	2021-05-15 16:50:14	999F8EBA-356F-42FC-AA9E-E2929261DCB8

SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 79

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.

Safari Browser Cache - WebKit Cache

Metadata and Cached Data may be stored in a variety of files

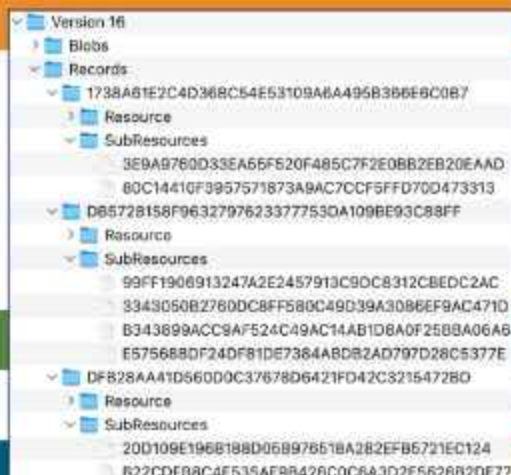
Version ###/Records

- Per Website Cached Metadata
- **/SubResources**
 - Cached Items List Per Website Visit, Embedded Resource Hashes
- **/Resources**
 - Cached Content - Metadata & Cached Data
 - <hash>-blob - Usually media items too large to store in a single file

Version ###/Blobs

- Additional Cached Data (Images, HTML, etc.)
- Matches hash embedded in Resources files

Correlate with 20-byte SHA1 hash filenames



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 80

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 cached 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/>

SubResource – Website Visit, List of Cached Items

00C14410F3957571873A9AC70CF6FD700473313									
Find: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F									
Replace:									
Resources: All Sources Sources & Find Previous Next									
2912	6469612E	6F726708	00000001	5265736F	75726365	6F000000	01687474	70733A2F	dia.org Resourceo https://
2944	2F75706C	6F61642E	77696869	6D656469	612E6F72	672F7769	68697065	6469612F	/upload.wikimedia.org/wikipedia/
2976	636F6D6D	6F6E732F	7468756D	622F612F	61312F41	6C616E5F	54757269	6E675F41	commons/thumb/a/a1/Alan_Turing_A
3008	6765645F	31362E6A	70672F34	34307078	2D416C61	6E5F5475	72696E67	5F416765	ged_16.jpg/440px-Alan_Turing_Age
3040	645F3136	2E6A7067	FFFFFFF	03226824	45E2A93D	E34ED2A3	2C2E765C	1947043D	d_16.jpg.... "h\$E...N...v\ G =
3072	1738A61E	2C4D368C	54E53109	A6A49583	66E6C087	91444871	FE27D841	91444871	8. ,MG.T.1f....DHq.'A.DHq
3104	FE27D841	00002900	00000168	74747073	3A2F2F65	6E2E7769	68697065	6469612E	.'A) https://en.wikipedia.
3136	6F72672F	77696869	2F416C61	6E5F5475	72696E67	03000000	00000000	06000000	org/wiki/Alan_Turing
3168	01416363	65707448	00000001	696D6167	652F7765	62702C69	6D616765	2F706E67	AcceptH image/webp,image/png
3200	2C696D61	67652F73	76672878	6D6C2C69	6D616765	2F2A3B71	3D302E38	2C766964	,image/svg+xml,image/*;q=0.8,vid
3232	656F2F2A	3B713D30	2E382C2A	2F2A3B71	3D302E35	07000000	01526566	65726572	eo/*;q=0.8,*/*;q=0.5 Referer
3264	19000000	01687474	70733A2F	2F656E2E	77696869	70656469	612E6F72	672F0A00	https://en.wikipedia.org/
3296	00000155	7365722D	4167656E	74750000	00014D6F	7A696C6C	612F352E	3020284D	User-Agent: Mozilla/5.0 (M
3328	6163696E	746F7368	3820496E	74656C20	4D616320	4F532058	2031305F	31355F37	acintosh; Intel Mac OS X 10_15_7
3360	29204170	706C6557	65624869	742F3630	352E312E	31352028	4B48544D	4C2C206C	) AppleWebKit/605.1.15 (KHTML, l
3392	69686520	47656368	6F292056	65727369	6F6E2F31	342E3120	53616661	72692F36	ike Gecko) Version/14.1 Safari/6
3424	30352E31	2E313501	00000000	0000000D	00000001	77696869	70656469	612E6F72	05.1.15 wikipedia.or

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 81

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 Wikipedia article.

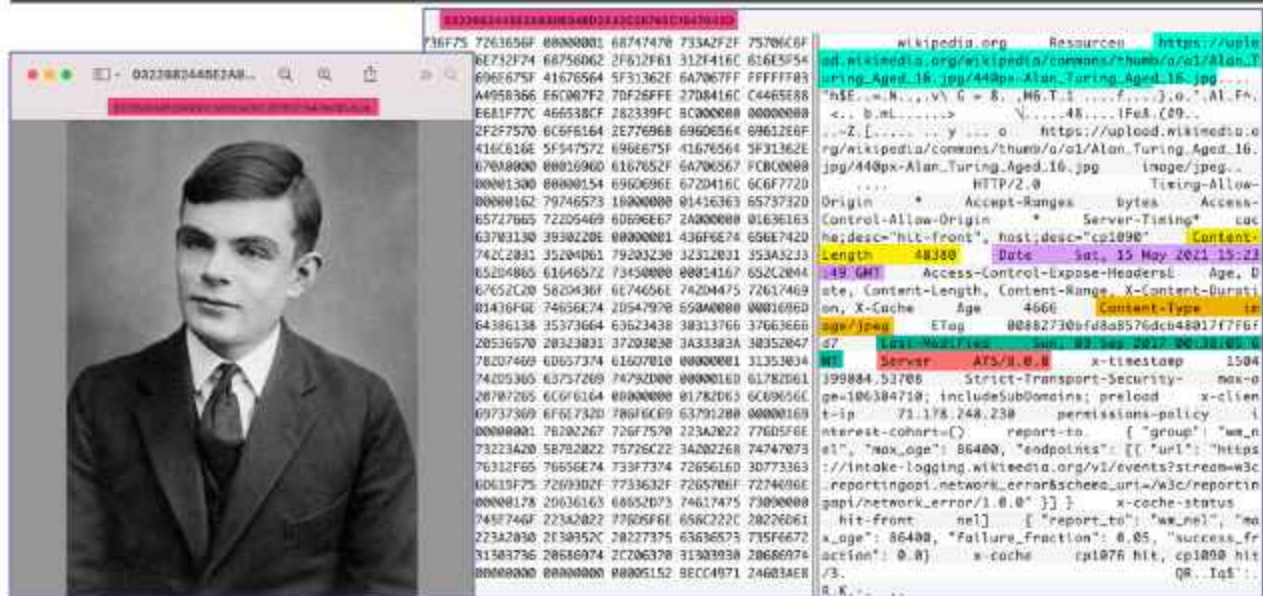
This metadata includes the original source website, the location of the photo on the webserver, and the recorded User Agent info. (It should be noted that the User Agent states 10.15.7, while the Safari browser on macOS 11.3 was actually used.)

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

80C1A410F396787873A9AC7CCBFDF0D0A73313															
Find 03 22 09 24 45 E2 AF 30 E3 4E 02 A3 2C 2E 76 5F 19 47 4E 32															
Replace															
2912	6469612E	6F726708	00000001	5265736F	75726365	6F000000	01687474	70733A2F	dia.org	Resource	https://				
2944	2F75706C	6F61642E	77696B69	6D656469	612E6F72	672F7769	68697065	6469612F	/upload.wikimedia.org/wiki/pedia/						
2976	636F6D6D	6F6E732F	7468756D	622F612F	61312F41	6C616E5F	54757269	6E675F41	commons/thumb/a/a1/Alan_Turing-A						
3008	6765645F	31362E6A	70672F34	34307078	2D416C61	6E5F5475	72696E67	5F416765	ged_16.jpg/440px-Alan_Turing-Age						
3040	645F3136	2E6A7067	FFFFFFFF	03226824	45E2A93D	E34ED2A3	2C2E765C	1947043D	d_16.jpg...	"h\$E...=..N...V\ G =					
3072	1738A61E	2C4D368C	54E53109	A6A495B3	66E6C087	91444871	FE27D841	91444871	8.,M6.T.1f....DHq.'A.DHq						
3104	FE27D841	00002900	00000168	74747073	3A2F2F65	6E2E7769	68697065	6469612E	..A) https://en.wikipedia.						
3136	6F72672F	77696B69	2F416C61	6E5F5475	72696E67	03000000	00000000	06000000	org/wiki/Alan_Turing						
3168	01416363	65707448	00000001	696D6167	652F7765	62702C69	6D616765	2F706E67	AcceptH	image/webp, image/png					
3200	2C696D61	67652F73	76672B78	6D6C2C69	6D616765	2F2A3B71	3D302E38	2C766964	, image/svg+xml, image/*;q=0.8, vid						
3232	656F2F2A	3B713D30	2E382C2A	2F2A3B71	3D302E35	07000000	01526566	65726572	eo/*;q=0.8,*/*;q=0.5	Referer					
3264	19000000	01687474	70733A2F	2F656E2E	77696B69	70656469	612E6F72	672F0A00	https://en.wikipedia.org/						
3296	00000155	7365722D	4167656E	74750000	00014D6F	7A696C6C	612F352E	3020284D	User-Agent	Mozilla/5.0 (M					
3328	6163696E	746F7368	3B20496E	74656C20	4D616320	4F532058	2031305F	31355F37	acintosh; Intel Mac OS X 10_15_7						
3360	29204170	706C6557	65624B69	742F3630	352E312E	31352028	4B48544D	4C2C206C	) AppleWebKit/605.1.15 (KHTML, 1						
3392	696B6520	47656368	6F292056	65727369	6F6E2F31	342E3120	53616661	72692F36	ike Gecko) Version/14.1 Safari/6						
3424	30352E31	2E313501	00000000	0000000D	00000001	77696B69	70656469	612E6F72	05.1.15	wikipedia.or					

20 bytes returned of 2000 bytes out of 2000 bytes

Resource – Cached Item and Metadata (or both in a single file)



The screenshot displays a forensic analysis tool interface. On the left, a thumbnail of a black and white portrait of Alan Turing is shown. To the right of the thumbnail is a list of hexadecimal data, likely representing the file's metadata or a hash. On the far right, a detailed view of the resource is shown, including the URL 'wikipedia.org', the resource type 'Resource', and the file path 'https://upload.wikimedia.org/wikipedia/commons/thumb/0/01/Alan_Turing_Aged_16.jpg'. The metadata includes fields such as 'Length', 'Date', 'Access-Control-Expose-Headers', 'Content-Type', 'Etag', and 'Last-Modified'.

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 83

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.


```

0322682445E2A93DE34ED2A32CE765C19A70A3D
736F75 7263656F 00000001 68747470 733A2F2F 75706C6F
5D6D6F 6E732F74 68756D62 2F612F61 312F416C 616E5F54
547572 696E675F 41676564 5F31362E 6A7067FF FFFFFFF03
8109A6 A495B366 E6C087F2 7DF26FFE 27D8416C C4465E88
8438B3 E681F77C 466538CF 282339FC BC000000 00000000
70733A 2F2F7570 6C6F6164 2E776968 696D6564 69612E6F
51312F 416C616E 5F547572 696E675F 41676564 5F31362E
2E6A70 670A0000 0001696D 6167652F 6A706567 FC8C0000
000000 00001300 00000154 696D696E 672D416C 6C6F772D
730500 00000162 79746573 1B000000 01416363 6573732D
000153 65727665 722D5469 6D696E67 2A000000 01636163
533D22 63703130 3930220E 00000001 436F6E74 656E742D
015361 742C2031 35204D61 79203230 32312031 353A3233
706F73 652D4865 61646572 73450000 00014167 652C2044
52616E 67652C20 582D436F 6E74656E 742D4475 72617469
000000 01436F6E 74656E74 2D547970 650A0000 0001696D
806266 64386138 35373664 63623438 30313766 37663666
203033 20536570 20323031 37203030 3A33383A 30352047
000001 782D7469 6D657374 616D7010 00000001 31353034
706F72 742D5365 63757269 74792D00 0000016D 61782D61
5E733B 20707265 6C6F6164 08000000 01782D63 6C69656E
55726D 69737369 6F6E732D 706F6C69 63791200 00000169
746FCA 00000001 78202267 726F7570 223A2022 776D5F6E
596E74 73223A20 587B2022 75726C22 3A202268 74747073
72672F 76312F65 76656E74 733F7374 7265616D 3D773363
536865 6D615F75 72693D2F 7733632F 7265706F 7274696E
7D0E00 00000178 2D636163 68652D73 74617475 73090000
706F72 745F746F 223A2022 776D5F6E 656C222C 20226D61
596F6E 223A2030 2E30352C 20227375 63636573 735F6672
016370 31303736 20686974 2C206370 31303930 20686974
000000 00000000 00000000 00005152 BECC4971 24603AE8

```

```

ad.wikimedia.
uring_Aged_16
"h$E...=N...
<.. b.ml...
...~Z.L.....
rg/wikipedia/
jpg/440px-Alc
....
Origin *
Control-Allow
he;desc="hit-
Length 48
:49 GMT A
ate, Content-
on, X-Cache
age/jpeg
d7 Last-M
MT Server
399084.53708
ge=106384710;
t-ip 71.1
nterest-cohor
el", "max_age
://intake-log
.reportingapi.network_error&schema_uri=/w3c/reportin
gapi/network_error/1.0.0" }] } x-cache-status
hit-front nel] {"report_to": "wm_nel", "ma
x-age": 86400, "failure_fraction": 0.05, "success_fr
action": 0.0} x-cache cp1076 hit, cp1090 hit
/3.
R.K.-. .

```



Safari Browser – Other Notable Files

`com.apple.Safari.plist`

- Recent Searches and Other Configurations

`Cookies.binarycookies`

- Proprietary File, plenty of scripts to parse them
- Also found in other apps!

Bookmarks - `Bookmarks.plist` (macOS), `Bookmarks.db` (iOS)

`RecentlyClosedTabs.plist` (macOS)

`Downloads.plist`

- Removed after one day from plist (by default)

Do you want to allow downloads on "www.whatsapp.com"?
You can change which websites can download files in Websites Preferences.

Cancel Allow

`PerSitePreferences.db` ("Do you want to allow downloads on...")

macOS 12 - `SafariTabs.db` - Tab Groups

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".

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 13: Apple Watch

Part 7: Reminders

This page intentionally left blank.

Section 4: Part 4

Apple Mail

This page intentionally left blank.

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

macOS

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

iOS

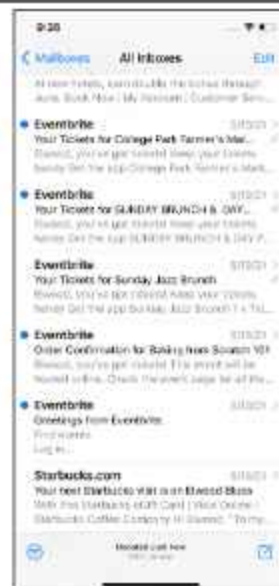
- iOS Backup: Not much.
- /private/var/mobile/Library/Mail/
- /private/var/mobile/Containers/Data/Application/<GUID>/

Data Types

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

Mailboxes

All Inboxes	888
iCloud	1
Gmail	888
VIP	
iCloud	
Inbox	1
Other	888



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/

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.

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.

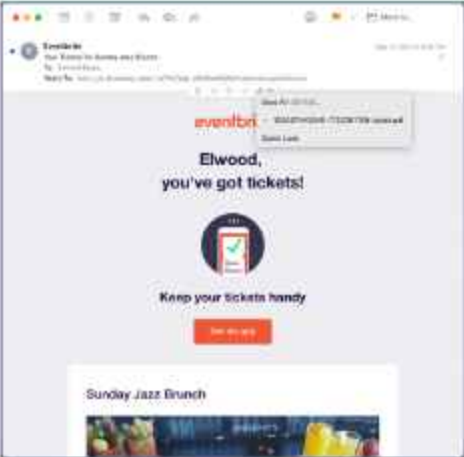
The screenshot shows mailbox (.mbox) directories. Depending on the email provider these can be named differently:

- 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.

macOS Mail - Mailbox (.mbox)

```
root@MBP-M1 [Gmail].mbox # pwd
/Users/elwoodblues/Library/Mail/VB/38338C85-86CA-45F7-93A1-73B73D5931D/[Gmail].mbox
root@MBP-M1 [Gmail].mbox # ls
All Mail.mbox  Drafts.mbox  Important.mbox  Info.plist  Sent Mail.mbox  Spam.mbox  Starred.mbox  Trash.mbox
root@MBP-M1 [Gmail].mbox # tree All\ Mail.mbox
All\ Mail.mbox
├── BADE6E98-DC4D-4C58-BAAB-5C644EFB68EE
│   └── Data
│       ├── Attachments
│       │   ├── 10
│       │   │   └── 2
│       │   │       └── 153867443845-1722381199-ticket.pdf
│       │   ├── 4
│       │   │   └── 2
│       │   │       └── 153867443845-1722381199-ticket.pdf
│       │   ├── 8
│       │   │   └── 2
│       │   │       └── 135438613539-1722384465-ticket.pdf
│       │   ├── 9
│       │   │   └── 2
│       │   │       └── 127228665363-1722381947-ticket.pdf
│       └── Messages
│           ├── 10.partial.emlx
│           ├── 100.emlx
│           ├── 101.emlx
│           ├── 102.emlx
│           ├── 103.emlx
│           ├── 104.emlx
│           ├── 105.emlx
│           ├── 106.emlx
│           └── 107.emlx
```



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.

macOS Mail – Downloaded Email Attachment via Quick Look

```
root@MBP-M1 Mail Downloads # pwd
/Users/elwoodblues/Library/Containers/com.apple.mail/Data/Library/Mail Downloads
root@MBP-M1 Mail Downloads # xattr -x1 AD1BBF58-C14A-4E84-994E-8D6C86DFB2D2/153067443845-1722381199-ticket.pdf
com.apple.metadata:kMDLabel maa3mfbt56bkw3vf5kxygvvtcq:
```

- Using 'Quick Look'
 - ~/Library/Mail Downloads/
 - ~/Library/Containers/
com.apple.mail/Data/Library/
Mail Downloads/
- Using 'Save'
 - ~/Downloads
- Extended Attributes

A2 06 01	...b...A.B..
41 BA 18	...&.s.c...v..
68 A8 B4	.d=...v...h..
6D 55 75	...z?H...nU..
A5 FA BB	}M...&..g...
54 91 A9	z...&...T...T..
07 D2 2F	-...&...B.../
07 32 45	0...&...2E
AB D5 08	...{...<...
76 7A CE	t...&...2...>vz.
F8 C3 F4	...i...&...&...
AA 57 59	...Jz.N...}WY
E2 53 98	...H...&...P.S.
28 50 A6	...X7..&A2..&P.
88 1A D1	j...W...&...
FC 3D 03	k...>...I.L...=.
9A 9C F6	...&...&...&...
FB 32 2B	...i...t.J...2+
	...z...&...&...

```
00000129 com.apple.quarantine: [0082;60a13e12;Ma  
00000000 38 30 38 32 38 36 30 61 31 33 65 31 32 38 40 61 [0082;60a13e12;Ma  
00000010 49 6C 3B [11:]  
00000013
```

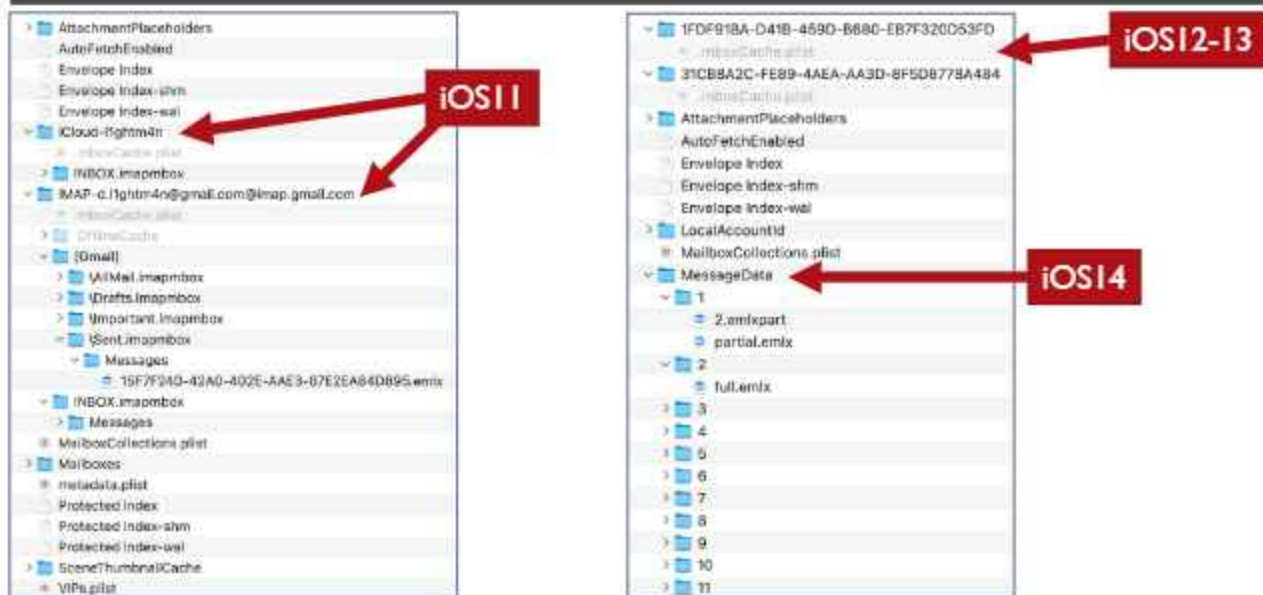


A user can view an attachment in a couple of ways:

- 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.

iOS Mail: /mobile/Library/Mail – Location of Cached Messages



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 95

The Mail directory on a full file system dump of iOS can contain 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.

Apple Mail: Envelope Index – Email Metadata

```

1 SELECT
2   MESSAGES.ROWID,
3   DATETIME(MESSAGES.DATE_SENT, (UNIXPOCH)) AS DATE_SENT, DATETIME(MESSAGES.DATE_RECEIVED, (UNIXPOCH)) AS DATE_RECEIVED, DATETIME(MESSAGES.DATE_LAST_VIEWED, (UNIXPOCH)) AS DATE_LAST_VIEWED,
4   ADDRESSES.ADDRESS, SUBJECTS.SUBJECT, MESSAGES.READ, MESSAGES.FLAGGED, MESSAGES.DELETED, MESSAGES.SIZE, MESSAGES.SHIFT, ATTACHMENTS.NAME, MAILBOXES.URL
5 FROM MESSAGES
6 LEFT JOIN ADDRESSES ON MESSAGES.SENDER = ADDRESSES.ROWID
7 LEFT JOIN SUBJECTS ON MESSAGES.SUBJECT = SUBJECTS.ROWID
8 LEFT JOIN MAILBOXES ON MESSAGES.MAILBOX = MAILBOXES.ROWID
9 LEFT JOIN ATTACHMENTS ON MESSAGES.ROWID = ATTACHMENTS.MESSAGE

```

ROWID	DATE_SENT	DATE_RECEIVED	DATE_LAST_VIEWED	address	subject	read	flagged	deleted	size	charset	name	url
1	2021-05-13 14:28:46	2021-05-13 14:28:49	2021-05-13 14:28:49	apple@apple.com	Welcome to iCloud Mail	0	0	0	11812	UTF-8	Welcome to iCloud Mail	imap://1181244-0380-004
2	2021-05-13 15:35:53	2021-05-13 15:35:54	2021-05-13 15:35:54	apple@apple.com	Security alert	0	0	0	11117	UTF-8	iCloud was granted access	imap://1181244-0380-004
3	2021-05-13 15:49:14	2021-05-13 15:49:15	2021-05-13 15:49:15	apple@apple.com	is One of the Sucker Five's	0	0	0	20100	UTF-8	Continuing from Apple	imap://1181244-0380-004
4	2021-05-14 15:04:14	2021-05-14 15:04:15	2021-05-14 15:04:15	apple@apple.com	Reminder for New Awards	0	0	0	182574	UTF-8	Find events by Tickets Get	imap://1181244-0380-004
5	2021-05-14 15:10:17	2021-05-14 15:10:18	2021-05-14 15:10:18	apple@apple.com	Get started with ...	0	0	0	47476	UTF-8	HOWARD YOURSELF WITH	imap://1181244-0380-004
6	2021-05-14 15:15:18	2021-05-14 15:15:19	2021-05-14 15:15:19	apple@apple.com	Don't miss out on the	0	0	0	37101	UTF-8	Discover Create an event	imap://1181244-0380-004
7	2021-05-15 21:15:00	2021-05-15 21:15:01	2021-05-15 21:15:01	apple@apple.com	Celebrate 1,000 Hotels	0	0	0	151842	UTF-8	At new hotels, earn double	imap://1181244-0380-004
8	2021-05-15 21:56:56	2021-05-15 21:56:57	2021-05-15 21:56:57	apple@apple.com	Your Tickets for College	0	0	0	206250	UTF-8	Good, you've got tickets!	imap://1181244-0380-004
9	2021-05-15 22:53:51	2021-05-15 22:53:52	2021-05-15 22:53:52	apple@apple.com	Your Tickets for SUNDAY	0	0	0	203833	UTF-8	Good, you've got tickets!	imap://1181244-0380-004
10	2021-05-15 22:53:56	2021-05-15 22:53:57	2021-05-15 22:53:57	apple@apple.com	Your Tickets for Sunday	0	0	0	203875	UTF-8	Good, you've got tickets!	imap://1181244-0380-004
11	2021-05-15 22:59:51	2021-05-15 22:59:52	2021-05-15 22:59:52	apple@apple.com	Order Confirmation for ...	0	0	0	81268	UTF-8	Good, you've got tickets!	imap://1181244-0380-004
12	2021-05-15 22:59:55	2021-05-15 22:59:56	2021-05-15 22:59:56	apple@apple.com	Grouping from Twitter	0	0	0	88477	UTF-8	Find events Log in to show	imap://1181244-0380-004
13	2021-05-15 18:20:02	2021-05-15 18:20:03	2021-05-15 18:20:03	apple@apple.com	Your most Starbucks visit	1	0	0	18916	UTF-8	With this Starbucks eGift	imap://1181244-0380-004
14	2021-05-15 17:14:38	2021-05-15 17:14:39	2021-05-15 17:14:39	apple@apple.com	Good, these settings	1	0	0	31851	UTF-8	Finally set up	imap://1181244-0380-004
15	2021-05-15 17:15:54	2021-05-15 17:15:55	2021-05-15 17:15:55	apple@apple.com	Security alert	0	0	0	11117	UTF-8	New device signed in to	imap://1181244-0380-004
16	2021-05-15 17:15:54	2021-05-15 17:15:55	2021-05-15 17:15:55	apple@apple.com	Security alert	0	0	0	11117	UTF-8	iOS was granted access to	imap://1181244-0380-004
17	2021-05-15 15:00:00	2021-05-15 15:00:01	2021-05-15 15:00:01	apple@apple.com	Your Apple ID was used to	1	0	0	18710	UTF-8	Dear Edward Mark, Your	imap://1181244-0380-004

SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 96

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.


```

1 SELECT
2   MESSAGES.ROWID,
3   DATETIME(MESSAGES.DATE_SENT, 'UNIQUEPOCH') AS DATE_SENT, DATETIME(MESSAGES.DATE_RECEIVED, 'UNIQUEPOCH') AS DATE_RECEIVED,
4   ADDRESSES.ADDRESS, SUBJECT, SUBJECT, MESSAGES.READ, MESSAGES.FLAGGED, MESSAGES.DELETED, MESSAGES.SIZE, MESSAGES.SNIPPET, ATTACHMENTS.NAME, MAILBOXES.URL
5 FROM MESSAGES
6 LEFT JOIN ADDRESSES ON MESSAGES.SENDER = ADDRESSES.ROWID
7 LEFT JOIN SUBJECTS ON MESSAGES.SUBJECT = SUBJECTS.ROWID
8 LEFT JOIN MAILBOXES ON MESSAGES.MAILBOX = MAILBOXES.ROWID
9 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	2021-05-13 14:28:46	2021-05-13 14:28:47	NO	noreply@gmail.appl	Welcome to iCloud Mail.	0	0	0	31812	Welcome to iCloud Mail. Yo...	nmap://53903244-0C88-459
2	2021-05-15 15:36:53	2021-05-15 15:36:54	NO	no...	Security alert	0	0	0	11337	macOS was granted access ...	nmap://38330C55-06CA-458
3	2021-05-15 13:49:14	2021-05-15 13:49:15	NO	newsdigest@nidea...	Is One of the Soviet Era's ...	0	0	0	101690	Good morning from Apple ...	nmap://38330C55-06CA-458
4	2021-05-14 15:04:14	2021-05-14 15:05:18	NO	noreply_at_event_av...	Reminder for New Branch...	0	0	0	162074	Find events by Tickets: Get...	nmap://38330C55-06CA-458
5	2021-05-14 09:16:57	2021-05-14 09:19:18	NO	Starbucks@cs.starbu...	Gas started with ...	0	0	0	47478	REWARD YOURSELF WITH ...	nmap://38330C55-06CA-458
6	2021-05-14 03:55:38	2021-05-14 05:25:21	NO	noreply_at_order_av...	Don't miss out on An ...	0	0	0	37061	Discover Create an event Hi...	nmap://38330C55-06CA-458
7	2021-05-13 23:13:00	2021-05-13 23:22:33	NO	hyatt@am.hyatt.com	Celebrate 1,000 Hotels w/...	0	0	0	151842	At new hotels, earn double ...	nmap://38330C55-06CA-458
8	2021-05-13 22:56:56	2021-05-13 22:57:02	NO	noreply_at_order_av...	Your Tickets for College ...	0	0	0	208250	Elwood, you've got tickets!	nmap://38330C55-06CA-458
9	2021-05-13 22:53:41	2021-05-13 22:53:51	NO	noreply_at_order_av...	Your Tickets for SUNDAY ...	0	0	0	203083	Elwood, you've got tickets!	nmap://38330C55-06CA-458
10	2021-05-13 22:52:48	2021-05-13 22:52:57	NO	noreply_at_order_av...	Your Tickets for Sunday ...	0	0	0	201871	Elwood, you've got tickets!	nmap://38330C55-06CA-458
11	2021-05-13 22:40:41	2021-05-13 22:49:42	NO	noreply@order.av...	Order Confirmation for ...	0	0	0	81568	Elwood, you've got tickets!	nmap://38330C55-06CA-458
12	2021-05-13 22:48:45	2021-05-13 22:48:46	NO	noreply@order.av...	Greetings from EarthWile	0	0	0	384277	Find events log in Hi Elwood	nmap://38330C55-06CA-458
13	2021-05-13 18:20:02	2021-05-13 18:20:06	NO	starbucks@giftcard...	Your next Starbucks visit...	1	0	0	198516	With this Starbucks eGift ...	nmap://38330C55-06CA-458
14	2021-05-13 17:14:58	2021-05-13 17:14:58	NO	no...	Elwood, finish setting ...	1	0	0	31661	Finish set-up - 100%	nmap://38330C55-06CA-458
15	2021-05-13 17:13:54	2021-05-13 17:13:54	NO	no...	Security alert	0	0	0	113337	New device signed in to ...	nmap://38330C55-06CA-458
16	2021-05-13 17:13:54	2021-05-13 17:13:55	NO	no...	Security alert	0	0	0	113312	iOS was granted access to ...	nmap://38330C55-06CA-458
17	2021-05-13 15:46:03	2021-05-13 15:46:04	NO	noreply@gmail.appl	Your Apple ID was used t...	1	0	0	107210	Dear Elwood Blues, Your ...	nmap://38330C55-06CA-458

Lab 4.2

Safari and Mail

This page intentionally left blank.

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 13: Apple Watch

Part 7: Reminders

This page intentionally left blank.

Section 4: Part 5

Communication

This page intentionally left blank.

Communication

Messages – SMS and iMessage

- Deprecated: Jabber (XMPP), Yahoo!, ICQ, AOL AIM, etc.

FaceTime

Call History

Voicemail

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.

macOS Messages - Archived Chats (10.15-)

~/Library/Messages/Archive/YYYY-MM-DD/*.ichat Files

<buddy's_username> on YYYY-MM-DD at HH:MM:SS

NSKeyedArchiver Plist Files

Archive	2018-09-19	on 2018-09-05 at 14.08.12
Attachments	2018-09-20	on 2018-09-21 at 11.10.02
chat.db	2018-09-21	on 2018-08-12 at 23.37.15
chat.db-shm	2018-09-22	on 2018-09-21 at 09.32.18
chat.db-wal	2018-09-23	on 2018-09-04 at 13.05.26
CloudKitMetaData	2018-09-24	on 2018-09-21 at 13.41.49
StickerCache	2018-09-25	Chat with on 2018-08-11 at 17.59.27
	2018-09-26	Chat with on 2018-09-07 at 19.44.51
	2018-09-27	Chat with on 2018-05-12 at 19.08.07
	2018-09-28	on 2018-09-21 at 13.21.11
	2018-09-29	on 2018-09-21 at 18.14.45

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.

Messages Database - iOS sms.db | macOS chat.db

macOS:

- ~/Library/Messages/chat.db

iOS FFS:

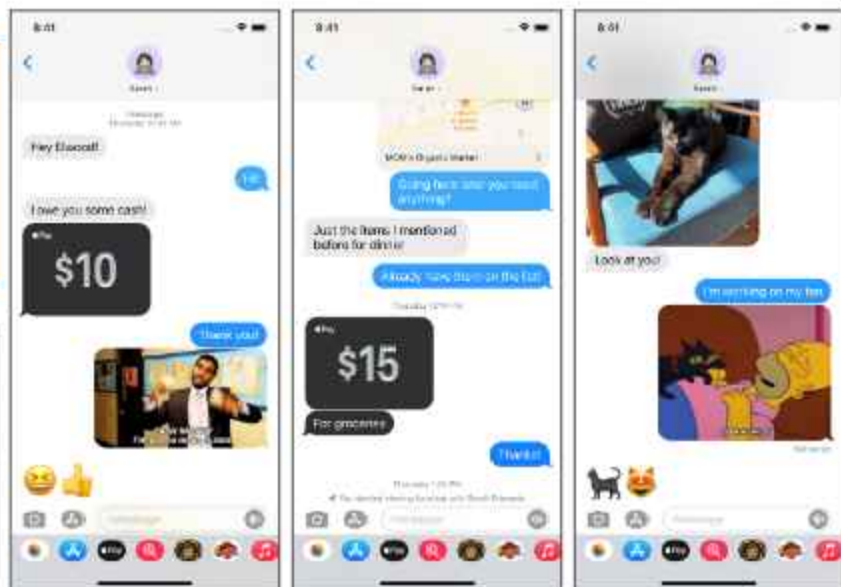
- /private/var/mobile/Library/SMS/sms.db

iOS Backup:

- /mobile/Library/SMS/sms.db

File Explorer View:

- ✓ SMS
 - > Attachments
 - > CloudKitMetadata
 - > com.apple.messages.geometrycache_v5.plist
 - > Drafts
 - > PluginMetadataCache
 - > prewarm.db
 - sms.db
 - sms.db-shm
 - sms.db-wal



SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 103

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.

Chat Database (sms.db/chat.db): Message Metadata

```

1 SELECT
2 CASE
3 WHEN LENGTH(message_date) = 18 THEN DATETIME(message_date/1000000000+87657256, 'unixepoch', 'localtime')
4 WHEN LENGTH(message_date) = 9 THEN DATETIME(message_date + 876572700, '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+876572700, 'unixepoch', 'localtime')
9 WHEN LENGTH(message_date_delivered) = 9 THEN DATETIME(message_date_delivered + 876572700, '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+876572700, 'unixepoch', 'localtime')
14 WHEN LENGTH(message_date_read) = 9 THEN DATETIME(message_date_read + 876572700, 'unixepoch', 'localtime')
15 ELSE 'N/A'
16 END AS 'DATE READ',
17 MESSAGE_TEXT, HANDLE_ID AS 'CONTACT ID', MESSAGE_SERVICE, MESSAGE_ACCOUNT, MESSAGE_ID_DELIVERED, MESSAGE_ID_FROM_ME, MESSAGE_ATTRIBUTEDBODY,
18 ATTACHMENT_FILENAME, ATTACHMENT_MIME_TYPE, ATTACHMENT_TRANSFER_NAME, ATTACHMENT_TOTAL_BYTE
19 FROM MESSAGE
20 LEFT OUTER JOIN MESSAGE_ATTACHMENT ON MESSAGE_ROWID = MESSAGE_ATTACHMENT_MESSAGE_ID
21 LEFT OUTER JOIN ATTACHMENT ON MESSAGE_ATTACHMENT_ID = ATTACHMENT_ID
22 LEFT OUTER JOIN HANDLE ON MESSAGE_HANDLE_ID = HANDLE_ROWID

```

	MESSAGE DATE	DATE DELIVERED	DATE READ	TEXT	CONTACT ID	SERVICE	ACCOUNT	MESSAGE ID_DELIVERED	MESSAGE ID_FROM_ME	ATTACHED BODY
1	2021-05-11 20:01:37	N/A	2021-05-11 20:01:37	Hey there!	contactid@icloud.com	Message	com.apple.iCloud	1	0	10/10
2	2021-05-11 20:02:33	2021-05-11 18:42:12	N/A	Hi	contactid@icloud.com	Message	com.apple.iCloud	1	1	10/10
3	2021-05-11 20:02:33	N/A	2021-05-11 20:02:33	I see you were out!	contactid@icloud.com	Message	com.apple.iCloud	1	0	10/10
4	2021-05-11 20:02:33	N/A	2021-05-11 20:02:33	👋	contactid@icloud.com	Message	com.apple.iCloud	1	0	10/10
5	2021-05-11 20:02:33	2021-05-11 18:42:12	N/A	Thank you!	contactid@icloud.com	Message	com.apple.iCloud	1	1	10/10
6	2021-05-11 20:02:33	2021-05-11 18:42:12	N/A		contactid@icloud.com	Message	com.apple.iCloud	1	1	10/10
7	2021-05-11 20:02:33	N/A	2021-05-11 20:02:33	👋	contactid@icloud.com	Message	com.apple.iCloud	1	0	10/10
8	2021-05-11 20:02:33	N/A	2021-05-11 20:02:33	Long time no see! (or not?)	contactid@icloud.com	Message	com.apple.iCloud	1	1	10/10
9	2021-05-11 20:02:33	N/A	2021-05-11 20:02:33	Just the best! (or not?)	contactid@icloud.com	Message	com.apple.iCloud	1	0	10/10
10	2021-05-11 20:02:33	2021-05-11 18:42:12	N/A	Always have them on the list!	contactid@icloud.com	Message	com.apple.iCloud	1	1	10/10
11	2021-05-11 20:02:33	N/A	2021-05-11 20:02:33	👋	contactid@icloud.com	Message	com.apple.iCloud	1	0	10/10
12	2021-05-11 20:02:33	N/A	2021-05-11 20:02:33	For goodness	contactid@icloud.com	Message	com.apple.iCloud	1	0	10/10
13	2021-05-11 20:02:33	2021-05-11 18:42:12	N/A	Thank!	contactid@icloud.com	Message	com.apple.iCloud	1	1	10/10
14	2021-05-11 20:02:33	N/A	N/A					0	0	10/10
15	2021-05-11 20:02:33	N/A	N/A					0	0	10/10
16	2021-05-11 20:02:33	N/A	2021-05-11 20:02:33	Look at you!	contactid@icloud.com	Message	com.apple.iCloud	1	0	10/10
17	2021-05-11 20:02:33	2021-05-11 18:42:12	N/A	I'm missing my app!	contactid@icloud.com	Message	com.apple.iCloud	1	1	10/10
18	2021-05-11 20:02:33	2021-05-11 18:42:12	N/A		contactid@icloud.com	Message	com.apple.iCloud	1	1	10/10
19	2021-05-11 20:02:33	N/A	N/A	👋	contactid@icloud.com	Message	com.apple.iCloud	1	0	10/10

104

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, lines 14 & 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/>


```

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!	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	0	N/A
2	2021-05-13 10:42:11	2021-05-13 10:42:13	N/A	Hi!	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	1	N/A
3	2021-05-13 10:42:39	N/A	2021-05-13 10:42:39	I owe you some cash!	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	0	N/A
4	2021-05-13 10:43:18	N/A	2021-05-13 10:43:20	💎	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	0	N/A
5	2021-05-13 10:43:28	2021-05-13 10:43:29	N/A	Thank you!	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	1	N/A
6	2021-05-13 10:43:58	2021-05-13 10:43:59	N/A		oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	1	N/A
7	2021-05-13 10:44:11	N/A	2021-05-13 10:44:12	👉👉	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	0	N/A
8	2021-05-13 10:49:13	2021-05-13 10:49:14	N/A	Going here later you need anything?	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	1	N/A
9	2021-05-13 10:50:08	N/A	2021-05-13 10:50:21	Just the items I mentioned before for dinner	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	0	N/A
10	2021-05-13 10:50:49	2021-05-13 10:50:50	N/A	Already have them on the list!	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	1	N/A
11	2021-05-13 12:14:11	N/A	2021-05-13 12:14:22	💎	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	0	N/A
12	2021-05-13 12:14:11	N/A	2021-05-13 12:14:22	For groceries	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	0	N/A
13	2021-05-13 12:14:28	2021-05-13 12:14:29	N/A	Thanks!	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	1	N/A
14	2021-05-13 13:35:34	N/A	N/A	N/A	N/A			0	0	N/A
15	2021-05-13 13:35:45	N/A	N/A	N/A	N/A			0	0	N/A
16	2021-05-16 18:45:08	N/A	2021-05-16 18:45:51	Look at you!	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	0	N/A
17	2021-05-16 18:47:39	2021-05-16 18:47:39	N/A	I'm working on my tan	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	1	N/A
18	2021-05-16 18:48:36	2021-05-16 18:48:38	N/A		oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	1	N/A
19	2021-05-16 18:49:22	N/A	N/A	👉👉	oompag@ch.rit.edu	iMessage	Eelwoodblues531060@gmail.com	1	0	N/A

Messages Database - Apple Pay - attributedBody

Also look at payload_data column in messages!



is_from_me	attributedBody	
0	BLOB	0000 84 0b 73 74 72 65 01 6d 74 79 70 65 64 81 e8 03 ...streamtyped...
1	BLOB	0010 84 01 40 84 84 12 4e 53 41 74 74 72 69 62 75 ...6...NSAttribu
0	BLOB	0020 74 65 64 53 74 72 69 6e 67 00 84 84 08 4e 53 4f ...tedString...NSO
0	BLOB	0030 62 6a 65 63 74 00 85 92 84 84 08 4e 53 53 74 ...bject...NSStr
0	BLOB	0040 72 69 6e 67 01 94 84 01 2b 06 ef bf bd ef bf bc ...ring...4...
0	BLOB	0050 86 84 02 69 49 01 01 92 84 84 0c 4e 53 44 69 ...i...NSD
0	BLOB	0060 63 74 69 6f 6e 61 72 79 00 94 84 01 69 03 92 84 ...ctionary...i
0	BLOB	0070 96 96 26 5f 5f 6b 49 4d 42 61 73 65 57 72 69 74 ..._id_MBaseWrit
0	BLOB	0080 69 6e 67 44 69 72 65 63 74 69 6f 6e 41 74 74 72 ...ingDrectionActi
0	BLOB	0090 69 62 75 7a 65 4e 61 6d 65 86 92 84 84 08 4e ...i but eName...N
0	BLOB	00a0 53 4e 75 6d 62 65 72 00 84 84 07 4e 53 56 61 6c ...SNumber...NSVal
0	BLOB	00b0 75 65 00 94 84 01 2a 84 84 01 71 9d ff 86 92 84 ...ue...7...q...
0	BLOB	00c0 96 96 26 5f 5f 6b 49 4d 42 72 65 61 64 63 72 75 ..._id_MBaseWrit
0	BLOB	00d0 6d 62 54 65 78 74 4d 61 72 6b 65 72 41 74 74 72 ...nbTextMarkerAttr
0	BLOB	00e0 65 6e 74 20 24 31 30 20 77 69 74 68 20 44 5f 5f ...i but eName...S
1	BLOB	00f0 65 6e 74 20 24 31 30 20 77 69 74 68 20 44 5f 5f ...ent: \$10 with App
1	BLOB	0100 6c 65 c2 a0 50 61 79 2e 86 92 84 96 96 1e 5f 5f ...le...Pay...
1	BLOB	0110 6b 49 4d 42 72 65 61 64 63 72 75 6d 62 54 65 78 ...id_MBaseWrit
0	BLOB	0120 74 4f 70 74 69 6f 6e 46 6c 61 67 73 86 92 84 9b ...tOptionFlags...
0	BLOB	0130 9c 9d 9d 00 86 86 97 02 01 92 84 98 99 03 92 84 ..._id_MFileTran
1	BLOB	0140 96 96 22 5f 5f 6b 49 4d 46 69 6c 65 54 72 61 6e ...sferGLCAttribut
0	BLOB	0150 73 66 65 72 47 55 49 44 41 74 74 72 69 62 75 74 ...eName...SDB78A
0	BLOB	0160 65 4e 61 6d 65 86 92 84 96 96 24 44 42 37 38 41 ...F35-038C-4E8A-BC
0	BLOB	0170 46 33 35 2d 30 33 38 44 2d 34 45 42 34 2d 42 43 ...70-81026EAD0B09
1	BLOB	0180 37 30 2d 38 31 30 32 36 45 41 30 45 42 36 39 86 ..._id_Me
0	BLOB	0190 92 99 92 9a 92 84 96 96 1d 5f 5f 6b 49 4d 4d 65 ...ssagePartAttribu
0	BLOB	01a0 73 73 61 67 65 50 61 72 74 41 74 74 72 69 62 75 ...tName...
0	BLOB	01b0 74 65 4e 61 6d 65 86 92 a1 86 86

SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 106

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'.

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

[illegible]

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 107

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.



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	2611496
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

Call History – com.apple.facetime and com.apple.mobilephone



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 109

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.

Call History - CallHistory.storedata (iOS)

```

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	MISS	us
2	2021-05-17 07:39:38	com.apple.FaceTime	+1	FACETIME VIDEO	0	INCOMING	0.0	MISS	us
3	2021-05-17 07:38:27	com.apple.FaceTime	+1	FACETIME AUDIO	0	INCOMING	0.0	MISS	us
4	2021-05-15 11:35:23	com.apple.FaceTime	(S)	FACETIME VIDEO	0	OUTGOING	0.0	53194	us
5	2021-05-15 11:34:33	com.apple.FaceTime	(S)	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

Other Service Providers: com.apple.Telephony, com.whatsapp.WhatsApp, BJ4HAAB9B3.us.zoom.videomeetings, etc.



FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 110

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)

Call History - CallHistory.storedata (iOS and macOS)

Some data may be synced across devices

	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

iOS

Encrypted BLOB

	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

macOS

SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 111

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.

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	<i>BLIND</i>	FACETIME AUDIO	0	INCOMING	0.0	0	us
2 2021-05-17 07:39:38	com.apple.FaceTime	<i>BLIND</i>	FACETIME VIDEO	0	INCOMING	0.0	0	us
3 2021-05-17 07:38:27	com.apple.FaceTime	<i>BLIND</i>	FACETIME AUDIO	0	INCOMING	0.0	0	us
4 2021-05-15 11:35:23	com.apple.FaceTime	<i>BLIND</i>	FACETIME VIDEO	0	OUTGOING	0.0	53194	us
5 2021-05-15 11:34:33	com.apple.FaceTime	<i>BLIND</i>	FACETIME AUDIO	0	OUTGOING	64.7367010116577	2649840	us
6 2021-05-13 12:16:54	com.apple.FaceTime	<i>BLIND</i>	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

iOS Voicemail - voicemail.db, AMR, and *.transcript Files

```

1 SELECT
2 ROWID, DATETIME(DATE, 'UNIXEPOCH', LOCALTIME) AS 'TIMESTAMP',
3 SENDER, DURATION, TRASHED, DATE, FLAG, REMOTE_ID
4 FROM VOICEMAIL
5 ORDER BY TIMESTAMP DESC
        
```

ROWID	TIMESTAMP	sender	duration	trashed_date	flag	remote_id
9	2021-03-18 11:15:08	+1	18	0	4355	378
10	2021-03-09 11:12:27	+1	28	0	4355	378
11	2021-03-08 11:44:42	+1	37	0	4355	377
12	2021-03-04 11:15:00	+1	18	0	4355	376
13	2021-02-11 16:10:29	+1	21	0	4355	375
14	2021-02-02 11:15:55	+1	28	0	4355	374
15	2021-01-11 09:08:18	+1	33	0	4354	373
16	2021-01-09 10:11:01	+1	27	0	4354	372
17	2020-12-28 10:41:10	+1	31	0	4355	371
18	2020-12-16 14:53:28	+1	27	0	4354	370
19	2020-10-30 10:48:38	873	14	0	4355	369
20	2020-10-02 11:01:31	+1	9	0	4355	368
21	2020-10-02 10:11:00	+1	0	0	4355	367
22	2020-09-23 14:44:37	+1	7	0	4355	366
23	2020-08-28 19:07:19	+1	27	0	4419	0
24	2020-08-28 18:33:43	+1	28	0	4419	0
25	2020-08-28 11:12:41	+1	28	643126745	4427	0
26	2020-08-20 12:17:58	+1	23	0	4419	0
27	2020-08-18 08:05:43	+1	49	0	4418	0
28	2020-08-15 11:15:08	+1	2	643126727	4171	0

```

154 => "Agent press one and you will be c
connected to the concern department if we don'
t hear from you then we will be forced to tak
e legal action against you press one and you'
ll be connected to the concern department"
155 => {
  "classes" => [
    0 => "VMVoicemailTranscript"
    1 => "NSObject"
  ]
  "classname" => "VMVoicemailTranscript"
}
        
```

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 113

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.


```
SELECT
  records.ROWID,date/time/recent,last_date/2000,timestamp as 'last Date',
  records.display_name,records.burde,identity/records.original_source,records.sending_address,records.dates,
  contacts.kind,contacts.address,contactid/key,contactdata/value
FROM records
left join metadata on records.ROWID = metadata.record_id
left contacts on records.ROWID = contacts.contact_id
```

[illegible]

```
~/Library/Containers/com.apple.corerecents.recentssd/Data/Library/Recentssd/
iOS Physical: /private/var/mobile/Library/Recentssd
iOS Backup: /mobile/Library/Recentssd
```

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.

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 6: Calendar

Part 12: Maps

Part 7: Reminders

Part 13: Apple Watch

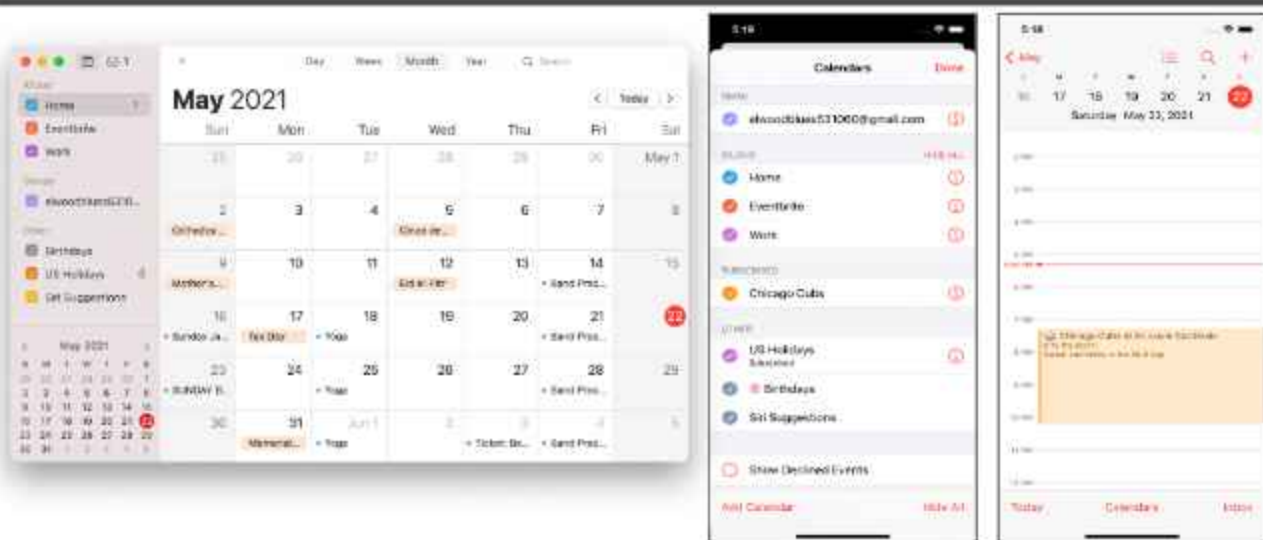
This page intentionally left blank.

Section 4: Part 6

Calendar

This page intentionally left blank.

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



SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 118

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.

Calendars – com.apple.iCal and com.apple.mobilecal [2]

*/Library/Calendar

Sources

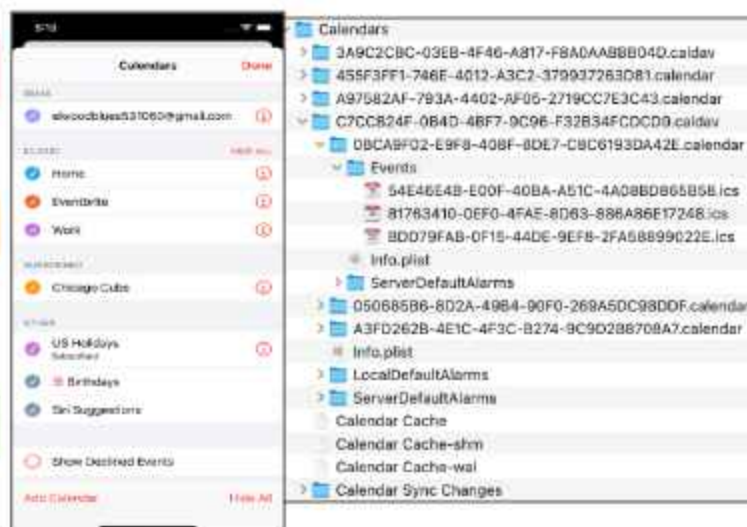
- Synced or Local
- iCloud, Apps, Email Providers, etc.

ICS Files on macOS

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

Databases

- macOS: 'Calendar Cache'
- iOS: Calendar.sqlite3db



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 119

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

macOS Calendars – ICS Files

The screenshot shows a macOS calendar interface with a weekly view from May 20 to 25. A red arrow points from a calendar event on May 23 to a detailed ICS file view on the right. The ICS file view contains the following text:

```

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:8DD79FAB-8F15-44DE-9EF8-2FA58899022E
DTSTAMP:20210513T225326Z
LOCATION:Darna'n946 North Jackson Street\, Arlington\, VA 22201
DESCRIPTION:https://www.eventbrite.com/checkout-external?eid=12722866536
3&Y&F=e30e
URL;VALUE=URI:com-eventbrite-attendee:///e/127228665363?calendar=true
X-APPLE-SCHEDULETAG:
X-APPLE-SERVERFILENAME:8DD79FAB-8F15-44DE-9EF8-2FA58899022E.ics
SEQUENCE:0
X-APPLE-NEWS-BUSINESSSTATUS:BUSY
SUMMARY:SUNDAY BRUNCH & DAY PARTY
LAST-MODIFIED:20210513T225326Z
DTSTART:TZID=America/New_York:20210523T140000
CREATED:20210513T225326Z
X-APPLE-ETAG:"konhmzsa"
END:VEVENT
END:VCALENDAR
  
```

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 120

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.

```
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=127228665363&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:"konhmsa"
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-

```

1 SELECT
2   DATETIME(ZCALENDARITEM.ZCREATIONDATE+978307200,(UNWPOUCH/LOCALTIME) AS 'CREATED', DATETIME(ZCALENDARITEM.ZDATESTAMP+978307200,(UNWPOUCH/LOCALTIME) AS 'LAST MODIFIED',
3   DATETIME(ZCALENDARITEM.ZSTARTDATE+978307200,(UNWPOUCH/LOCALTIME) AS 'EVENT START', DATETIME(ZCALENDARITEM.ZENDDATE+978307200,(UNWPOUCH/LOCALTIME) AS 'EVENT END'),
4   ZCALENDARITEM.ZTITLE, ZNOTE, ZTITLE AS 'CALENDAR',
5   ZCALENDARITEM.ZTITLE AS 'EVENT', ZCALENDARITEM.ZDURATION, ZCALENDARITEM.ZSALUDAY, ZCALENDARITEM.ZNOTES, ZCALENDARITEM.ZURLSTRING
6 FROM ZCALENDARITEM
7 LEFT JOIN ZNOTE ON ZCALENDARITEM.ZCALENDAR = ZNOTE.Z_PK
    
```

	CREATED	LAST MODIFIED	EVENT START	EVENT END	TIMEZONE	CALENDAR	EVENT	DURATION	ZSALUDAY	ZNOTES	ZURLSTRING
1	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	UTC	US Holidays	Memorial Day	1 day	1		
2	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	UTC	US Holidays	Memorial Day	1 day	1		
3	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	UTC	US Holidays	Juneteenth	1 day	1		
4	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	UTC	US Holidays	Tax Day	1 day	1		
5	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	UTC	US Holidays	Arise	1 day	1	The exact date of this holiday is...	
6	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	America/New_York	Work	Auto Practice	1 day	1		
7	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	UTC	Home	Workout	1 day	1		
8	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	America/New_York	Work	Work	1 day	1		
9	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	America/New_York	Work	Work	1 day	1		
10	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	America/New_York	Work	Work	1 day	1		
11	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	America/New_York	Work	Work	1 day	1		
12	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	2021-05-11 12:07:00	GMT-0400	Work	Work	1 day	1		

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!

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 13: Apple Watch

Part 7: Reminders

This page intentionally left blank.

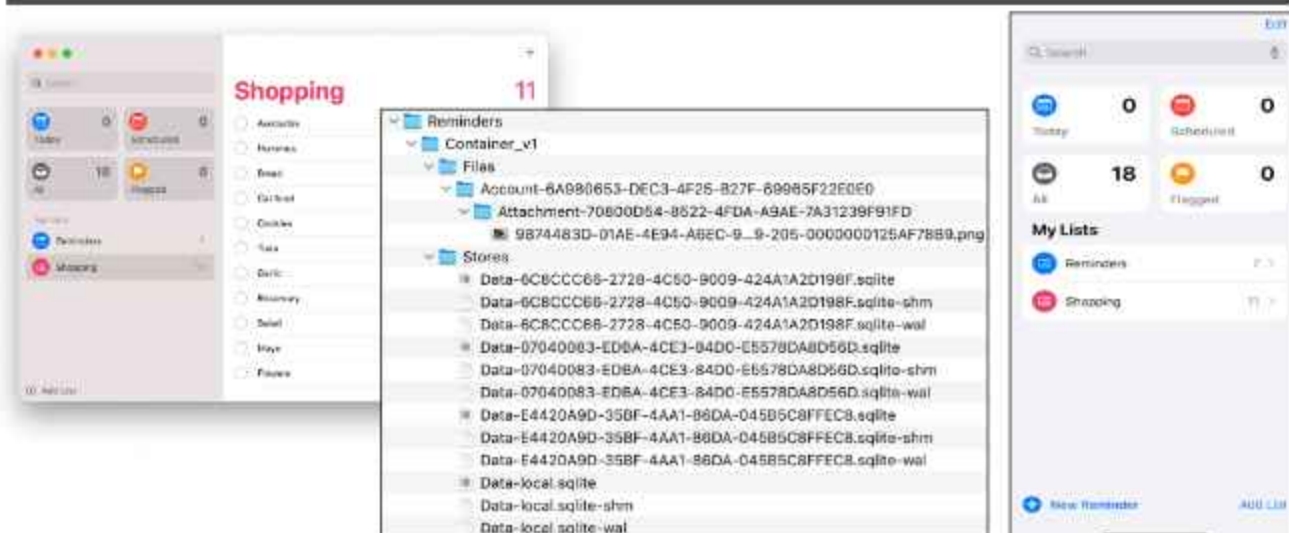
Section 4: Part 7

Reminders

This page intentionally left blank.

Reminders – com.apple.reminders (10.15+, iOS 13+)

*/Library/Reminders



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.

Reminders – Data-[UUID/Local].sqlite Database

```

1 SELECT
2 ZEMCDOBJECT.Z_ENT, ZEMCDOBJECT.Z_ACCOUNT AS 'ACCOUNT', ZEMCDOBJECT.ZNAME AS 'ACCOUNT NAME',
3 CASE ZEMCDOBJECT.Z_ENT
4 WHEN 8 THEN 'ACCOUNT'
5 WHEN 8 THEN 'TRIGGER - PROXIMITY'
6 WHEN 22 THEN 'LIST'
7 WHEN 25 THEN 'REMINDER'
8 ELSE ZEMCDOBJECT.Z_ENT
9 END 'OBJECT TYPE',
10 ZEMCDOBJECT.ZNAME AS 'REMEMINDER LIST', ZEMCDOBJECT.ZLIST AS 'REMEMINDER LIST Q.PID', ZEMCDOBJECT.ZTITLE AS 'REMEMINDER TITLE', ZEMCDOBJECT.ZDUEDATE AS 'DUE DATE',
11 ZEMCDOBJECT.ZLOCATIONDATE AS 'CREATION DATE', ZEMCDOBJECT.ZLASTMODIFIEDDATE 'MODIFIED DATE', ZEMCDOBJECT.ZMARKERORDELETION, ZEMCDOBJECT.ZADDRESS AS 'ADDRESS', ZEMCDOBJECT.ZTITLE AS 'LOCATION NAME',
12 CASE ZEMCDOBJECT.ZPROXIMITY
13 WHEN 1 THEN 'ARRIVING'
14 WHEN 2 THEN 'LEAVING'
15 END 'PROXIMITY TYPE'
16 FROM ZEMCDOBJECT

```

Z_ENT	Z_ACCOUNT	ACCOUNT NAME	OBJECT TYPE	REMEMINDER LIST	REMEMINDER LIST Q.PID	REMEMINDER TITLE	DUE DATE	CREATION DATE	MODIFIED DATE	ZMARKERORDELETION	ADDRESS	LOCATION NAME	PROXIMITY TYPE
8	4	Home	ACCOUNT							0			
22	20		LIST	Shopping						0			
18	20		LIST	Food						0			
18	20		LIST	Test						0			
29	22		LIST	Reminders						0			
25	21		REMINDER		1	Black		042622935.170945	042622935.170945	0			
25	21		REMINDER		2	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		3	Platinum		042622935.170945	042622935.170945	0			
25	21		REMINDER		4	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		5	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		6	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		7	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		8	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		9	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		10	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		11	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		12	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		13	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		14	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		15	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		16	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		17	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		18	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		19	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		20	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		21	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		22	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		23	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		24	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		25	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		26	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		27	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		28	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		29	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		30	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		31	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		32	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		33	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		34	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		35	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		36	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		37	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		38	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		39	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		40	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		41	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		42	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		43	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		44	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		45	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		46	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		47	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		48	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		49	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		50	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		51	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		52	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		53	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		54	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		55	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		56	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		57	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		58	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		59	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		60	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		61	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		62	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		63	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		64	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		65	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		66	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		67	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		68	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		69	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		70	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		71	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		72	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		73	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		74	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		75	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		76	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		77	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		78	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		79	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		80	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		81	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		82	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		83	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		84	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		85	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		86	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		87	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		88	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		89	Gold		042622935.170945	042622935.170945	0			
25	21		REMINDER		90	Gold		042622935.170945	042622935.170945	0		</	

Section 4: Agenda

Part 1: Application Fundamentals

Part 2: Introduction to SQLite Queries

Part 3: Safari Browser

Part 4: Apple Mail

Part 5: Communication

Part 6: Calendar

Part 7: Reminders

Part 8: Contacts

Part 9: Notes

Part 10: Wallet and Apple Pay

Part 11: Photos

Part 12: Maps

Part 13: Apple Watch

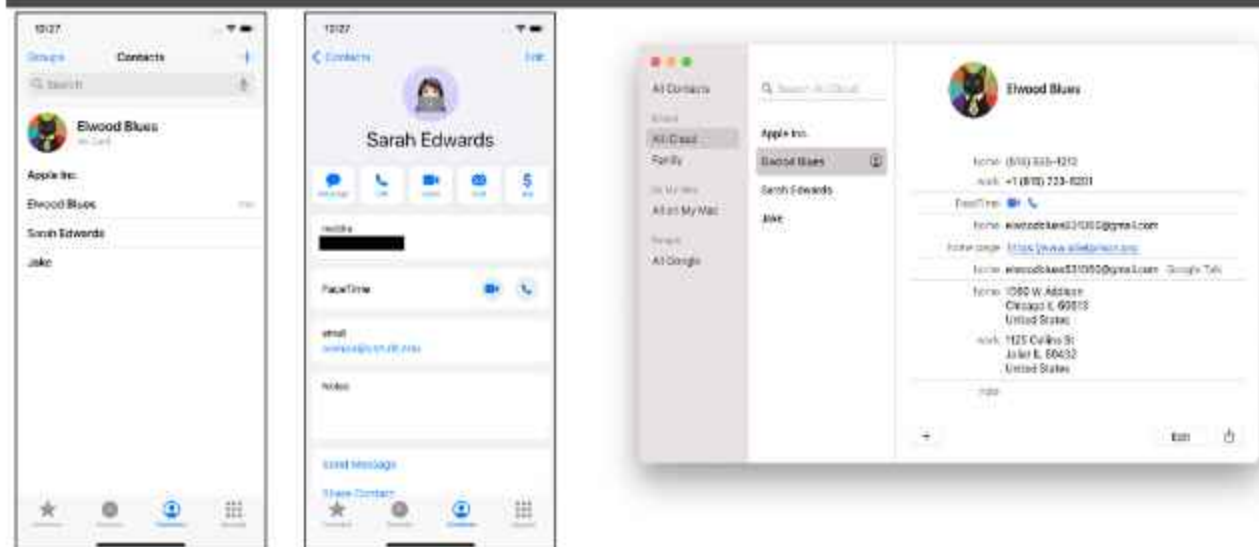
This page intentionally left blank.

Section 4: Part 8

Contacts

This page intentionally left blank.

Contacts - com.apple.AddressBook */Library/*/AddressBook



SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 131

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 Contacts - Sources and Person Records

The image shows two side-by-side screenshots. The left screenshot displays the 'Sources' directory within the macOS Address Book. It lists several sources, including '8B5A531B-F2A2-48BE-8476-B3D65F9AEEDC' and 'F5EE9E18-F812-410D-BAE5-4E6A52D14242'. A red arrow points from the 'F5EE9E18-F812-410D-BAE5-4E6A52D14242' source to the right screenshot. The right screenshot shows a detailed view of a person record in a database. The record includes fields such as 'Name', 'Email', 'Phone', 'Address', 'City', 'State', 'Country', and 'Organization'. The 'Name' field is highlighted in yellow, and the 'Email' field is highlighted in green. The 'Phone' field is highlighted in red. The 'Address' field is highlighted in blue. The 'City' field is highlighted in orange. The 'State' field is highlighted in purple. The 'Country' field is highlighted in pink. The 'Organization' field is highlighted in light blue.

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 132

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.abcdab.

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.

√ 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	+iEgE@Y~%
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	"kl92zlw"
> Email	Dictionary	(4 items)
> Phone	Dictionary	(4 items)
externalFilename	String	Yjc3YjFhMTItNzRlZi00NjAzLWE3MWQlNjU3NGUxYzRmMWNi.vcf
Creation	Date	2021-05-13T16:07:12Z
externalUUID	String	b77b1a12-74ef-4603-a71d-6574e1c4f1cb
imageType	String	PHOTO


```
SELECT  
ZABCRECORD.ZINHOKESTL,  
DATETIME(ZABCRECORD.CREATIONDATE + 978307200, UNKPGCH, LOCALTIME) AS 'CREATED', DATETIME(ZABCRECORD.MODIFICATIONDATE + 978307200, UNKPGCH, LOCALTIME) AS 'MODIFIED',  
ZABCRECORD.ZIPSTNAME, ZABCRECORD.ZLASTNAME, ZABCRECORD.ZFNAME, ZABCRECORD.ZADDRESS, ZABCRECORD.ZPHONENUMBER, ZFALNAME, ZFALPHONE, ZFALADDRESS, ZFALCITY, ZFALPOSTALADDRESS, ZSTAT, ZABCRECORD.ZCOUNTYNAM, ZABCRECORD.ZURL,  
FROM ZABCRECORD  
  
LEFT JOIN ZABCREMAILADDRESS ON ZABCRECORD.Z_PK = ZABCREMAILADDRESS.ZOWNER  
LEFT JOIN ZABCPHONENUMBER ON ZABCRECORD.Z_PK = ZABCPHONENUMBER.ZOWNER  
LEFT JOIN ZABCREADDRESS ON ZABCRECORD.Z_PK = ZABCREADDRESS.ZOWNER  
LEFT JOIN ZABCREMAILADDRESS ON ZABCRECORD.Z_OWNER = ZABCREMAILADDRESS.ZOWNER
```

	2:USERID	3:CREATED	4:MODIFIED	5:FIRSTNAME	6:LASTNAME	7:INAME	8:ADDRESS	9:PHONE	10:ZIP	11:ZIP2	12:STREET	13:CITY	14:STATE	15:COUNTRYNAME	16:URL
1	15180164-0075-1280-0010-12111000000000000000	2003-05-11 12:07:32	2003-05-11 12:07:31	John	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		
2	00100000-0000-0000-0000-00000000000000000000	2003-05-11 12:07:32	2003-05-11 12:07:31	John	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		
3	00100000-0000-0000-0000-00000000000000000000	2003-05-11 12:07:31	2003-05-12 00:23:18	John	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		
4	00100000-0000-0000-0000-00000000000000000000	2003-05-11 12:07:32	2003-05-11 14:10:30	John	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		
5	00100000-0000-0000-0000-00000000000000000000	2003-05-11 12:07:32	2003-05-11 14:15:14	John	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		
6	00100000-0000-0000-0000-00000000000000000000	2003-05-11 12:07:32	2003-05-11 11:49:18	David	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		
7	00100000-0000-0000-0000-00000000000000000000	2003-05-11 12:07:32	2003-05-11 11:49:18	David	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		
8	00100000-0000-0000-0000-00000000000000000000	2003-05-11 12:07:32	2003-05-11 11:49:18	David	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		
9	00100000-0000-0000-0000-00000000000000000000	2003-05-11 12:07:32	2003-05-11 11:49:18	David	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		
10	00100000-0000-0000-0000-00000000000000000000	2003-05-11 12:07:32	2003-05-11 11:49:18	David	Deity	Deity	7400-00-0000	1-780-00-0000	1-780-00-0000	California	CA	(United States)	http://www.apple.com		

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.

1	SELECT														
2	ZABCORD.ZUNIQUEID,														
3	DATETIME(ZABCORD.ZCREATIONDATE+978307200,'UNIXEPOCH',LOCALTIME) AS 'CREATED', DATETIME(ZABCORD.ZMODIFICATIONDATE+978307200,'UNIXEPOCH',LOCALTIME) AS 'MODIFIED',														
4	ZABCORD.ZFIRSTNAME,ZABCORD.ZLASTNAME,ZABCORD.ZNAME,ZABCORD.ZADDRESS,ZABCORD.ZPHONE,ZABCORD.ZCITY,ZABCORD.ZSTATE,ZABCORD.ZCOUNTRY,ZABCORD.ZURL														
5	ZABCORD.ZADDRESS,ZABCORD.ZCITY,ZABCORD.ZSTATE,ZABCORD.ZCOUNTRY,ZABCORD.ZURL														
6	FROM ZABCORD														
7	LEFT JOIN ZABCORD.ZADDRESS ON ZABCORD.Z_PK = ZABCORD.ZADDRESS.Z_OWNER														
8	LEFT JOIN ZABCORD.ZPHONE ON ZABCORD.Z_PK = ZABCORD.ZPHONE.Z_OWNER														
9	LEFT JOIN ZABCORD.ZCITY ON ZABCORD.Z_PK = ZABCORD.ZCITY.Z_OWNER														
10	LEFT JOIN ZABCORD.ZSTATE ON ZABCORD.Z_PK = ZABCORD.ZSTATE.Z_OWNER														
1	438E2794-467C-4796-8949-CD3A1E800337	ABPerson	2021-05-13 12:07:12	2021-05-13 12:07:14	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
2	478282A1-4023-4497-8F06-FB538F5724	ABGroup	2021-05-13 12:07:12	2021-05-15 11:28:06	Family	card	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
3	6741A0B1-844F-42C0-A740-E45CE333BC07	ABGroup	2021-05-13 12:07:11	2021-05-22 20:23:36	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL
4	CB7222EE-5B04-49C1-A0BC-702BD4E5098E	ABPerson	2021-05-13 12:07:12	2021-05-13 14:15:24	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair
5	CB7222EE-5B04-49C1-A0BC-702BD4E5098E	ABPerson	2021-05-13 12:07:12	2021-05-13 14:15:24	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair
6	DF191441-8626-48A8-ACB1-2B5A07D0286C	ABPerson	2021-05-13 12:07:13	2021-05-15 11:40:18	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair
7	DF191441-8626-48A8-ACB1-2B5A07D0286C	ABPerson	2021-05-13 12:07:13	2021-05-15 11:40:18	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair
8	DF191441-8626-48A8-ACB1-2B5A07D0286C	ABPerson	2021-05-13 12:07:13	2021-05-15 11:40:18	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair
9	DF191441-8626-48A8-ACB1-2B5A07D0286C	ABPerson	2021-05-13 12:07:12	2021-05-15 11:40:18	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair
10	DF191441-8626-48A8-ACB1-2B5A07D0286C	ABPerson	2021-05-13 12:07:12	2021-05-15 11:40:18	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair	Blair

Lab 4.3 - Applications: Part I

Choose your own adventure!

This page intentionally left blank.

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 13: Apple Watch

Part 7: Reminders

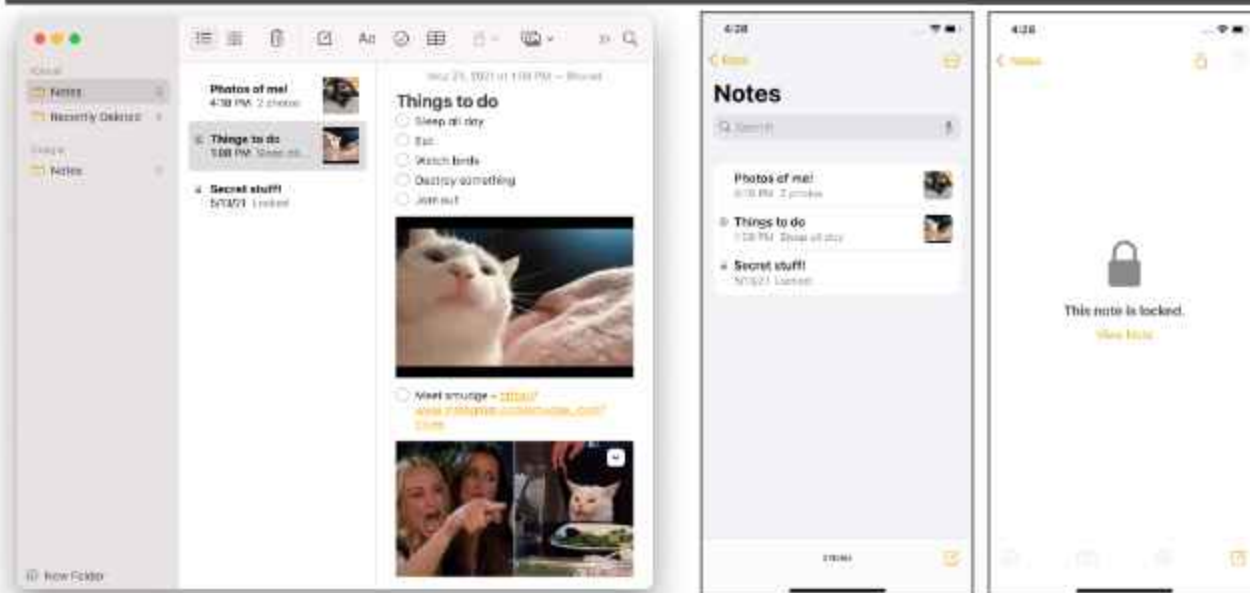
This page intentionally left blank.

Section 4: Part 9

Notes

This page intentionally left blank.

Notes – com.apple.notes and com.apple.mobilenotes



SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 139

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").

Notes - Files and Directories

Sources

Note Attachments in /Media Directory

- UUIDs referenced in note storage & database
- Secure note attachments are encrypted

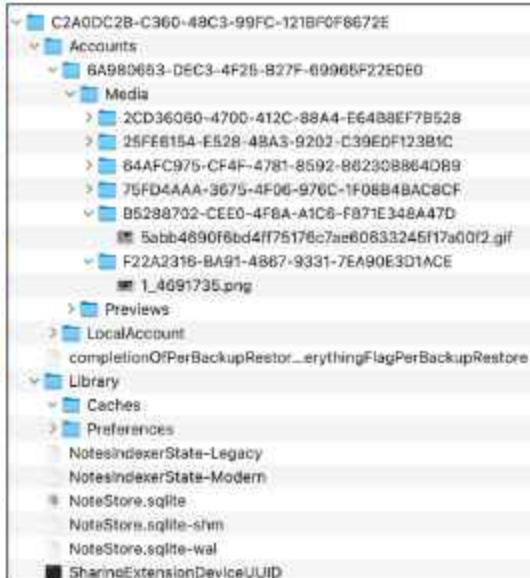
Thumbnails in /Preview Directory

Primary Database

- NoteStore.sqlite

Legacy Databases

- notes.sqlite
- NotesV#.storedata



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.

NotesStore.sqlite - Note Containers and Folders

- Folders: ZICLOUDSYNCINGOBJECT.Z_ENT = 11 and 12

```

1 SELECT
2   ZICLOUDSYNCINGOBJECT.Z_PK,ZICLOUDSYNCINGOBJECT.Z_ENT,
3   CASE ZICLOUDSYNCINGOBJECT.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 ZICLOUDSYNCINGOBJECT.Z_ENT
11  END AS "OBJECT TYPE",
12   ZICLOUDSYNCINGOBJECT.ZNAME,ZICLOUDSYNCINGOBJECT.ZTITLE2,
13   ZICLOUDSYNCINGOBJECT.ZACCOUNTNAMEFORACCOUNTLISTSORTING
14 FROM ZICLOUDSYNCINGOBJECT
15 WHERE "OBJECT TYPE" = "FOLDER"
  
```

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

Folders

Search	
iCloud	
Notes	3 >
Recently Deleted	5 >
On My iPhone	
Notes	0 >
Recently Deleted	6 >

SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 141

In the notes database, everything is mostly located in one table, ZICLOUDSYNCINGOBJECT. 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
1	ICAuthor
2	ICCloudState
3	ICCloudSyncingObject
4	ICAttachment
5	ICAttachmentPreviewImage
6	ICDeviceMigrationState
7	ICLegacyTombstone
8	ICMedia
9	ICNote
10	ICNoteContainer
11	ICAccount
12	ICFolder
13	ICDevice
14	ICGroup
15	ICLocation
16	ICAttachmentLocation
17	ICNoteChange
18	ICNoteData
19	ICPerson
20	ICSearchIndexTransaction
21	ICServerChangeToken
22	NextId


```

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.ZNAME,ZICCLOUDSYNCINGOBJECT.ZTITLE2,
13 ZICCLOUDSYNCINGOBJECT.ZACCOUNTNAMEFORACCOUNTLISTSORTING
14 FROM ZICCLOUDSYNCINGOBJECT
15 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

NotesStore.sqlite - Notes

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

```

1 SELECT
2   ZICLOUDSYNCHOBJECT_Z_PK, ZICLOUDSYNCHOBJECT_Z_ENT,
3   CASE ZICLOUDSYNCHOBJECT_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 ZICLOUDSYNCHOBJECT_Z_ENT
11  END AS 'OBJECT TYPE',
12   ZICLOUDSYNCHOBJECT_ZCREATIONDATE1, ZICLOUDSYNCHOBJECT_ZMODIFICATIONDATE1, ZICLOUDSYNCHOBJECT_ZLASTOPENEDDATE, ZICLOUDSYNCHOBJECT_ZMARKEDFORDELETION,
13   ZICLOUDSYNCHOBJECT_ZTITLE1 AS 'NOTE TITLE', ZICLOUDSYNCHOBJECT_ZSNIPPET, ZICLOUDSYNCHOBJECT_ZISPASSWORDPROTECTED AS 'PROTECTED',
14   ZICLOUDSYNCHOBJECT_ZPASSWORDHINT, ZICLOUDSYNCHOBJECT_ZFOLDER, ZICLOUDSYNCHOBJECT_ZACCOUNT3, ZICLOUDSYNCHOBJECT_ZISPINNED, ZICLOUDSYNCHOBJECT_ZDATA
15 FROM ZICLOUDSYNCHOBJECT
16 LEFT JOIN ZICNOTEDATA ON ZICLOUDSYNCHOBJECT_ZNOTEDATA == ZICNOTEDATA_Z_P
17 WHERE 'OBJECT TYPE' = 'NOTE'

```

Notes

Photos of me!

4:18 PM 2 photos

Things to do

1:08 PM Sleep all day

Secret stuff!

5/13/21 Locked

Z_PK	Z_ENT	OBJECT TYPE	ZCREATIONDATE1	ZMODIFICATIONDATE1	ZLASTOPENEDDATE	ZMARKEDFORDELETION	NOTE TITLE	ZSNIPPET	PROTECTED	ZPASSWORDHINT	ZFOLDER	ZACCOUNT3	ZISPINNED	ZDATA
38	9	NOTE	642609284.550799	642609284.550799	642609285.352613	1	New Note		0		22	23	0	
39	9	NOTE	642609175.562432	642609198.296803	642609400.629072	0	Secret stuff!		1	my normal password	20	23	0	
40	9	NOTE	642609186.948026	642609207.571788	642609206.43671	0	Photos of me!		0		20	23	0	
45	9	NOTE	642608668.768133	642608533.781124	642609407.261452	0	Things to do	Sleep all day	0		20	23	0	

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.

```

1 SELECT
2 ZICLOUDSYNCGOBJECT.Z_PK, ZICLOUDSYNCGOBJECT.Z_ENT,
3 CASE ZICLOUDSYNCGOBJECT.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 ZICLOUDSYNCGOBJECT.Z_ENT
11 END AS "OBJECT TYPE",
12 ZICLOUDSYNCGOBJECT.ZCREATIONDATE1, ZICLOUDSYNCGOBJECT.ZMODIFICATIONDATE1, ZICLOUDSYNCGOBJECT.ZLASTOPENEDATE, ZICLOUDSYNCGOBJECT.ZMARKEDFORDELETION,
13 ZICLOUDSYNCGOBJECT.ZTITLE AS "NOTE TITLE", ZICLOUDSYNCGOBJECT.ZSNIPPET, ZICLOUDSYNCGOBJECT.ZISPASSWORDPROTECTED AS "PROTECTED",
14 ZICLOUDSYNCGOBJECT.ZPASSWORDHINT, ZICLOUDSYNCGOBJECT.ZFOLDER, ZICLOUDSYNCGOBJECT.ZACCOUNT3, ZICLOUDSYNCGOBJECT.ZISPINNED, ZICNOTEDATA.ZDATA
15 FROM ZICLOUDSYNCGOBJECT
16 LEFT JOIN ZICNOTEDATA ON ZICLOUDSYNCGOBJECT.ZNOTEDATA = ZICNOTEDATA.Z_P
17 WHERE "OBJECT TYPE" = "NOTE"

```

Z_PK	Z_ENT	OBJECT TYPE	ZCREATIONDATE1	ZMODIFICATIONDATE1	ZLASTOPENEDATE	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
2	39	9	NOTE	642609175.562432	642609198.296609	643494500.629072	0	Secret stuff	NULL	1	my normal password	20	23	0
3	40	9	NOTE	642605156.948926	643493927.571789	643482604.41471	0	Photos of me!	0	0	NULL	20	23	0
4	45	9	NOTE	642608609.708133	643482533.781124	643494497.261452	0	Things to do	Sleep all day	0	NULL	20	23	0

NotesStore.sqlite – Note Contents GZIP and Secure Notes

• Cracking Secure Notes

- John the Ripper
- HashCat

• Know the password?

- https://github.com/threeplanetsoftware/apple_cloud_notes_parser

ZACCOUNT3	ZISPINNED	ZDATA
23	0	NULL
23	0	BLOB
23	0	BLOB
23	0	BLOB

```

0000 3f 8b 08 00 0
0010 18 00 e0 3d b
0020 0c e5 ec cd 5
0030 81 ac ee b4 2
0040 60 14 51 58 d
0050 14 46 d1 63 1
0060 39 7c ff ff 9
0070 5d 28 64 cf a
0080 de 98 aa a6 f
0090 90 36 38 c4 0
00a0 33 c9 b8 aa 6
00b0 3c c1 dd aa a
00c0 a9 4c bb d3 3
00d0 1d 83 c9 b8 7
00e0 6a 42 2f c7 1
00f0 a4 9e 14 72 d
0100 f0 48 30 81 2
0110 38 3d 5f 58 0
0120 f2 41 76 43 e
0130 86 d7 d1 d6 3
0140 e8 04 2c 50 d
0150 0e e4 84 d6 5
0160 87 40 b6 62 9
0170 3a 4a 02 0b 7
0180 fa 25 d0 db 6
0190 08 65 40 1f 1
01a0 2b 60 59 87 6
01b0 41 30 37 b5 2
    
```

May 23, 2021 at 1:08 PM - Shared

Things to do

- ☐ Sleep all day
- ☐ Eat
- ☐ Watch birds
- ☐ Destroy something
- ☐ Jam out



Meet smudge - <https://www.youtube.com/watch?v=9gHjIw>



```

51 .....[H.Q
5b .....gn...t.]
40 .....Gx$.v...L.@
6e .....1.1(...n
54 .....CX...a...T
96 .....F.c...Ag...C...
c0 9|.....9... (d...
e2 |(d...h"...S.N.
a1 .....P...C.8...
e5 .....68...D...
cc 3...e.x...a./...
69 <.....p8...?..l
c5 .....L...922...62Z(
1d .....s)...L...b...
41 jB/.....tA
14 .....r...A0r...#.A...
6a .....H0.1b...f...t...Lj
40 8=X.j9H.....X@
ce .....AvC...&F]....
db .....<?FH.....e...
ea .....P...3...d...
60 .....Yl...rV...67"
f9 .....@.b.&e...V...:"
60 :j...t'.5.B1.=t...
76 :%...n.cn...B...v
4b :e@...l.A(...\,CK
b0 +*Y,a_75...C^,8...
23 A07.*0.H.y...AA#
    
```

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 compressed 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/threeplanetsoftware/apple_cloud_notes_parser. This parser requires a known password, and that can be brute forced using John the Ripper or HashCat.

NotesStore.sqlite – Note Contents Protobuf

```

compa@MBP-M1 notes % file note45.bin
note45.bin: gzip compressed data, original size modulo 2^22 1566
compa@MBP-M1 notes % gcat note45.bin | xxd -c 30
00000000: 8898 1299 6c68 8818 8812 726c 1182 615a 8889 6467 7328 744f 3864 678a 6565 6665 7828 618c 8c28 .....Things to do, Sleep all
00000020: 6461 7984 4561 740a 5761 7a63 6528 6269 7264 738a 4405 7374 726f 7928 758f 6d65 7468 696a 6728 ..day, Eat, Watch birds, Destroy something
00000040: 884a 41ad 206f 7574 8eef 878c 884a 6565 7428 7360 75.....Jam out, https://www
00000060: 2a67 4673 7461 6772 616d 2a63 676d 3f73 6d78 6467 68.....instagram.com/smudge
00000080: 8a5a 188a 6468 8818 8818 801a 6468 8818 8818 651a 18....._lord/?hl=en\357\277\274\n"
000000a0: 128a 8e88 8218 6518 811a 8a88 6118 8218 8118 611a 17.....
compa@MBP-M1 notes % gcat note45.bin > note45.pb
compa@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"
    3: {
      1: {
        1: 0
        2: 0
      }
      2: 0
      3: {
        1: 0
        2: 0
      }
    }
  }
}
000000c0: 8898 1212 636f 6c28 636f 6d76 7578 6572 7a68 2a67 69.....
000000e0: 8a18 8013 9c8c f2c5 4779 93e2 8ebb 6c96 4731 1800 2a.....
00000100: f205 4779 93e2 8ebb 6c96 4731 1800 4620 6074 7a78 72.....
00000120: 636f 6d76 7578 6572 636f 6d76 636f 6d76 636f 6d76 636f 6d76 .....
00000140: 8a18 8013 9c8c f2c5 4779 93e2 8ebb 6c96 4731 1800 2a.....
00000160: 8a18 8013 9c8c f2c5 4779 93e2 8ebb 6c96 4731 1800 2a.....
00000180: 8a18 8013 9c8c f2c5 4779 93e2 8ebb 6c96 4731 1800 2a.....
000001a0: 8a18 8013 9c8c f2c5 4779 93e2 8ebb 6c96 4731 1800 2a.....
000001c0: 8a18 8013 9c8c f2c5 4779 93e2 8ebb 6c96 4731 1800 2a.....

```

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 146

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>.

Metadata & Thumbnails: Z ENT = 4 and 5

Looking for attachment metadata for Note Z PK=45, we need to now filter for attachment objects. Z ENT 4 and 5.

Z ENT 5 = Contains attachment thumbnail information

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.

148

NotesStore.sqlite – Note Files

Files: Z_ENT = 8

```

1 SELECT
2   ZICLOUDSYNCHINGOBJECT.Z_PK,ZICLOUDSYNCHINGOBJECT.Z_ENT,
3   CASE ZICLOUDSYNCHINGOBJECT.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 ZICLOUDSYNCHINGOBJECT.Z_ENT
11  END AS "OBJECT TYPE",
12   ZICLOUDSYNCHINGOBJECT.ZMARKEDFORDELETION,ZICLOUDSYNCHINGOBJECT.ZATTACHMENT1,
13   ZICLOUDSYNCHINGOBJECT.ZIDENTIFIER,ZICLOUDSYNCHINGOBJECT.ZFILENAME
14 FROM ZICLOUDSYNCHINGOBJECT
15 WHERE "OBJECT TYPE" = "FILE"

```

Z_PK	Z_ENT	OBJECT TYPE	ZMARKEDFORDELETION	ZATTACHMENT1	ZIDENTIFIER	ZFILENAME
41	8	FILE	0	42	25FE6154-E528-4BA3-9202-C39E0F123B1C	image.jpeg
47	8	FILE	1	44	2CD36060-4700-412C-88A4-E64B8EF7B528	B788D696-A747-4765-B522-95F5AAEAEC2D.jpeg
51	8	FILE	0	52	85288702-CEE0-4F8A-A1C6-F871E348A47D	5abb4690f6bd4ff75176c7ae60633245f17a00f2.gif
55	8	FILE	0	56	F22A2316-BA91-4B67-9331-7EA90E3D1ACE	1_4691735.png
66	8	FILE	0	63	75FD4AAA-3675-4F06-976C-1F08B4BAC8CF	7A789A98-A46D-4C29-ABBE-4E7EBAA7A64.jpeg



SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 150

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.

```

1 SELECT
2   ZICLOUDSYNCINGOBJECT.Z_PK,ZICLOUDSYNCINGOBJECT.Z_ENT,
3   CASE ZICLOUDSYNCINGOBJECT.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 ZICLOUDSYNCINGOBJECT.Z_ENT
11  END AS "OBJECT TYPE",
12  ZICLOUDSYNCINGOBJECT.ZMARKEDFORDELETION,ZICLOUDSYNCINGOBJECT.ZATTACHMENT1,
13  ZICLOUDSYNCINGOBJECT.ZIDENTIFIER,ZICLOUDSYNCINGOBJECT.ZFILENAME
14  FROM ZICLOUDSYNCINGOBJECT
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-C39E0F123B1C	Image.jpeg
2	47	8	FILE	1	48	2CD36060-4700-412C-88A4-E6488EF78528	B788D696-A747-4765-B522-95F5AAEAE2D.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-1F0884BAC8CF	7A789A9B-A46D-4C29-AB8E-4EFEBA7A64.jpeg

Media	
2CD36060-4700-412C-88A4-E64B8EF7B528	
B788D696-A747-4765-B522-95F5AAEAE2D.jpeg	
25FE6154-E528-4BA3-9202-C39E0F123B1C	
64AFC975-CF4F-4781-8592-B6230B864DB9	
75FD4AAA-3675-4F06-976C-1F08B4BAC8CF	
B5288702-CEE0-4F8A-A1C6-F871E348A47D	
5abb4690f6bd4ff75176c7ae60633245f17a00f2.gif	
F22A2316-BA91-4B67-9331-7EA90E3D1ACE	
1_4691735.png	

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 13: Apple Watch

Part 7: Reminders

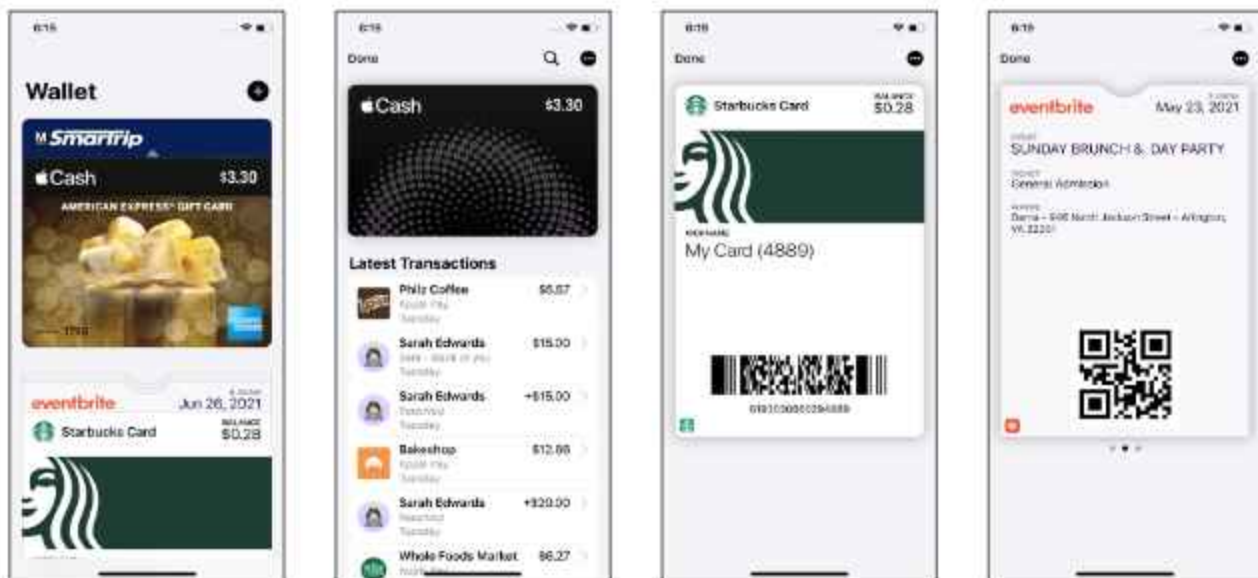
This page intentionally left blank.

Section 4: Part 10

Wallet and Apple Pay

This page intentionally left blank.

Wallet and Apple Pay [1]



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 154

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.

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 UbiquitousCards directory.

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

Pass and Card *.pkpass Package Structure

-  icon.png
-  icon@2x.png
-  logo.png
-  logo@2x.png
-  manifest.json
-  pass.json
-  signature
-  strip.png
-  strip@2x.png

```

compsMMP-M1 Cards N jq 4 ./Cards/BYK1V1Dc5wMG5epH-c39w9XuD=.pkpass/pass.json
{
  "passTypeIdentifier": "pass.com.starbucks.card",
  "formatVersion": 1,
  "serialNumber": "0002774930216090-D0B8D0-2079-4A6D-87C2-B0F0F01000A",
  "description": "Starbucks Card",
  "organizationName": "Starbucks",
  "teamIdentifier": "29HAG455A",
  "logoText": "Starbucks Card",
  "associatedStatusIdentifiers": [
    35117771A
  ],
  "foregroundColor": "rgb(0, 0, 0)",
  "backgroundColor": "rgb(255, 255, 255)",
  "staticCard": {
    "headerFields": {
      {
        "key": "balance",
        "changeMessage": "Your balance is now $0.",
        "label": "Balance",
        "currencyCode": "USD",
        "value": "0.00"
      }
    },
    "auxiliaryFields": {
      {
        "key": "nickname",
        "changeMessage": "Nickname updated to $0.",
        "label": "Nickname",
        "value": "My Card (4889)"
      }
    }
  },
  "barcode": {
    "format": "PDABarcodeFormat",
    "message": "A19303A00234860",
    "messageEncoding": "ISO-8859-1",
    "altText": "A19303A00234860"
  },
  "authenticationToken": "F010781-A640-4421-812a-01234567890",
  "webServiceURL": "https://comps.starbucks.com/v1/passable"
}

```



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.

passes23.sqlite – Passes and Cards

```

1 SELECT
2 DATETIME(PASS.INGESTED_DATE+978307200,'UNIXEPOCH') AS 'INGESTED DATE',
3 DATETIME(PASS.MODIFIED_DATE+978307200,'UNIXEPOCH') AS 'MODIFIED DATE',
4 CASE PASS.PASS_FLAVOR
5 WHEN 0 THEN 'PASS'
6 WHEN 1 THEN 'CARD'
7 END 'TYPE',
8 PASS.UNIQUE_ID AS 'UNIQUE ID', PASS.ORGANIZATION_NAME AS 'ORGANIZATION NAME',
9 PASS.GROUP_PASS_TYPE_ID AS 'PASS TYPE', PASS.GROUP_ORDER AS 'GROUP ORDER',
10 PASS.PRIMARY_ACCOUNT_SUFFIX AS 'ACCOUNT SUFFIX',
11 PASS.ISSUER_COUNTRY_CODE AS 'ISSUER COUNTRY CODE'
12 FROM PASS
13 LEFT JOIN LOCATION_SOURCE ON LOCATION_SOURCE.ID LIKE '%(PASS.UNIQUE_ID)%'
14 LEFT JOIN LOCATION ON LOCATION.LOCATION_SOURCE_ID == LOCATION_SOURCE.ID
15 LEFT JOIN PASS_GROUP ON PASS.GROUP_ID == PASS_GROUP.ID
16 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	CxYdH6Zn8BwqjVGxjW5su28=	Washington Metropolitan Area Transit Authority	paymentpass.com.apple	0	9632	US
2	2021-05-13 14:28:09	2021-05-13 14:40:38	CARD	ja3t5pqlRWNzH352mCngK2g=	Apple	paymentpass.com.apple	1	9632	US
3	2021-05-13 16:08:25	2021-05-13 16:40:38	CARD	h9gWMyBt4QCHyD9Mtu2K9dCk=	American Express	paymentpass.com.apple	2	1798	US
4	2021-05-13 22:52:42	2021-05-13 22:52:42	PASS	TPQ7Y6WAnJLWV8QmcVcxQx33Hlg=	Sunday Jazz Brunch	pass.eventbrite.ticket	3	9632	US
5	2021-05-13 22:54:03	2021-05-13 22:54:03	PASS	mbG2Yu2Aqpfqz67V3BDM0jRz=	SUNDAY BRUNCH & DAY PARTY	pass.eventbrite.ticket	3	9632	US
6	2021-05-13 22:56:21	2021-05-13 22:56:21	PASS	-f9guJWADyF5mg2ndngvYj9tY=	College Park Farmer's Market @ Penn Branch Parkway	pass.eventbrite.ticket	3	9632	US
7	2021-05-13 22:46:16	2021-05-14 19:05:38	PASS	0YMP9QzQm4Hd7mywN-639xWuG=	Starbucks	pass.com.starbucks.card	4	9632	US

SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 157

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 ID	STATUS	TRANSACTION TYPE	AMOUNT	CURRENCY CODE	CARD TYPE	PAYMENT TYPE	NAME	MERCHANT NAME	MERCHANT CATEGORY	PASS ID
1	2021-05-11 14:41:14	COMPLETED	APPLE CASH PEER PAYMENT	10.0 USD	Discover	APPLE CASH				ja25sqpfuRwHf1s2rM0yFQag=
2	2021-05-11 18:14:00	COMPLETED	APPLE CASH PEER PAYMENT	15.0 USD	Discover	APPLE CASH				ja25sqpfuRwHf1s2rM0yFQag=
3	2021-05-11 18:45:43	COMPLETED	PURCHASE	12.0 USD	Discover	APPLE CASH		SMARTER WASHINGTON DC	Greenwater Transport, Ferries	ja25sqpfuRwHf1s2rM0yFQag=
4	2021-05-11 17:12:40	COMPLETED	PURCHASE	5.0 USD	Discover	APPLE CASH	Starbucks	STARBUCKS CARD COPY		ja25sqpfuRwHf1s2rM0yFQag=
5	2021-05-11 17:21:43	NOT COMPLETED	PURCHASE	0.0 USD	American Express	CREDIT CARD	Pay			hgM1y6h4QCChuy5B8hu2B8u2Cf=
6	2021-05-11 17:40:56	NOT COMPLETED	PURCHASE	0.0 USD	American Express	CREDIT CARD	Pay			hgM1y6h4QCChuy5B8hu2B8u2Cf=
7	2021-05-11 22:40:40	NOT COMPLETED	PURCHASE	0.0 USD	American Express	CREDIT CARD	Pay			hgM1y6h4QCChuy5B8hu2B8u2Cf=
8	2021-05-11 22:57:12	NOT COMPLETED	PURCHASE	0.0 USD	American Express	CREDIT CARD	Pay			hgM1y6h4QCChuy5B8hu2B8u2Cf=
9	2021-05-11 11:52:50	NOT COMPLETED	PURCHASE	0.0 USD	American Express	CREDIT CARD	Pay			hgM1y6h4QCChuy5B8hu2B8u2Cf=
10	2021-05-11 11:53:18	NOT COMPLETED	PURCHASE	0.0 USD	American Express	CREDIT CARD	Pay			hgM1y6h4QCChuy5B8hu2B8u2Cf=
11	2021-05-11 11:53:32	NOT COMPLETED	PURCHASE	0.0 USD	American Express	CREDIT CARD	Pay			hgM1y6h4QCChuy5B8hu2B8u2Cf=
12	2021-05-11 17:54:49	NOT COMPLETED	PURCHASE	NULL	American Express	CREDIT CARD				hgM1y6h4QCChuy5B8hu2B8u2Cf=
13	2021-05-11 17:55:24	NOT COMPLETED	PURCHASE	NULL	American Express	CREDIT CARD				hgM1y6h4QCChuy5B8hu2B8u2Cf=
14	2021-05-11 17:56:00	COMPLETED	PURCHASE	8.27 USD	Discover	APPLE CASH	Whole Foods Market	WHOLEFOODS ARL 10042	Grocery Stores, Supermarkets	ja25sqpfuRwHf1s2rM0yFQag=
15	2021-05-11 17:59:45	COMPLETED	APPLE CASH PEER PAYMENT	20.0 USD	Discover	APPLE CASH				ja25sqpfuRwHf1s2rM0yFQag=
16	2021-05-11 18:09:33	COMPLETED	PURCHASE	12.89 USD	Discover	APPLE CASH		IQ MARKET	Miscellaneous General Merchandise	ja25sqpfuRwHf1s2rM0yFQag=
17	2021-05-11 18:13:47	NOT COMPLETED	PURCHASE	NULL	Discover	APPLE CASH				ja25sqpfuRwHf1s2rM0yFQag=
18	2021-05-11 18:14:00	NOT COMPLETED	PURCHASE	NULL	Discover	APPLE CASH				ja25sqpfuRwHf1s2rM0yFQag=
19	2021-05-11 18:14:20	NOT COMPLETED	PURCHASE	NULL	Discover	APPLE CASH				ja25sqpfuRwHf1s2rM0yFQag=
20	2021-05-11 18:15:10	COMPLETED	APPLE CASH PEER PAYMENT	15.0 USD	Discover	APPLE CASH				ja25sqpfuRwHf1s2rM0yFQag=
21	2021-05-11 18:15:34	NOT COMPLETED	PURCHASE	NULL	Discover	APPLE CASH				ja25sqpfuRwHf1s2rM0yFQag=
22	2021-05-11 18:17:32	COMPLETED	APPLE CASH PEER PAYMENT	15.0 USD	Discover	APPLE CASH				ja25sqpfuRwHf1s2rM0yFQag=
23	2021-05-11 18:18:44	TRANSMIT	TRANSMIT	NULL	Calix/Verizon Wireless	TRANSMIT				CaY5Gm6Z0M8u5y5B8hu2B8u2Cf=
24	2021-05-11 18:18:47	COMPLETED	PURCHASE	6.57 USD	Discover	APPLE CASH	Blue Coffee	SQ MARKET COFFEE	Baking Flavors, Restaurants	ja25sqpfuRwHf1s2rM0yFQag=
25	2021-05-11 17:04:01	NOT COMPLETED	PURCHASE	0.0 USD	American Express	CREDIT CARD	Pay			hgM1y6h4QCChuy5B8hu2B8u2Cf=

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.

```

1 SELECT
2   DATETIME(PAYMENT_TRANSACTION.TRANSACTION_DATE+978107200,'UNIXEPOCH') AS 'TRANSACTION DATE',
3   CASE PAYMENT_TRANSACTION.TRANSACTION_STATUS
4     WHEN -1 THEN 'TRANSIT'
5     WHEN 0 THEN 'NOT COMPLETED'
6     WHEN 1 THEN 'COMPLETED'
7   ELSE PAYMENT_TRANSACTION.TRANSACTION_STATUS
8   END 'STATUS',
9   CASE PAYMENT_TRANSACTION.TRANSACTION_TYPE
10    WHEN 0 THEN 'PURCHASE'
11    WHEN 2 THEN 'TRANSIT'
12    WHEN 3 THEN 'APPLE CASH PEER PAYMENT'
13  ELSE PAYMENT_TRANSACTION.TRANSACTION_TYPE
14  END 'TRANSACTION TYPE',
15   CASE WHEN PAYMENT_TRANSACTION.AMOUNT = 9223372036854775807 THEN 'NULL' ELSE PAYMENT_TRANSACTION.AMOUNT/100.00 END 'AMOUNT',
16   PAYMENT_TRANSACTION.CURRENCY_CODE AS 'CURRENCY CODE',PAYMENT_APPLICATION.DISPLAY_NAME AS 'CARD TYPE',
17   CASE PAYMENT_APPLICATION.PAYMENT_TYPE
18     WHEN 1 THEN 'APPLE CASH'
19     WHEN 2 THEN 'CREDIT CARD'
20     WHEN 1000 THEN 'TRANSIT'
21   ELSE PAYMENT_APPLICATION.PAYMENT_TYPE
22   END 'PAYMENT TYPE',
23   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'
24 FROM PAYMENT_TRANSACTION
25 JOIN PAYMENT_APPLICATION ON PAYMENT_TRANSACTION.SOURCE_PID == PAYMENT_APPLICATION.TRANSACTION_SOURCE_PID
26 JOIN PASS ON PAYMENT_APPLICATION.PASS_PID == PASS.PID
27 LEFT JOIN PLANTAINS ON PAYMENT_TRANSACTION.U == PLANTAINS.PID
28 ORDER BY TRANSACTION_DATE

```

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 PIER PAYMENT	10.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
2 2021-05-13 16:14:08	COMPLETED	APPLE CASH PIER PAYMENT	15.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
3 2021-05-13 16:45:45	COMPLETED	PURCHASE	12.0	USD	Discover	APPLE CASH	NULL	SMARTTRIP WASHINGTON, DC	Commuter Transport, Ferries	jAt15SpqfUKwxa81z2m0oqjC8hg=
4 2021-05-13 17:12:40	COMPLETED	PURCHASE	5.0	USD	Discover	APPLE CASH	Starbucks	STARBUCKS CARD EIGHT	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
5 2021-05-13 17:21:43	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hg8Kx1y6i4QChy9dMku25h6C7=
6 2021-05-13 17:46:56	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hg8Kx1y6i4QChy9dMku25h6C7=
7 2021-05-13 22:40:45	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hg8Kx1y6i4QChy9dMku25h6C7=
8 2021-05-13 22:57:12	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hg8Kx1y6i4QChy9dMku25h6C7=
9 2021-05-17 11:52:55	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hg8Kx1y6i4QChy9dMku25h6C7=
10 2021-05-17 11:53:18	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hg8Kx1y6i4QChy9dMku25h6C7=
11 2021-05-17 11:53:32	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hg8Kx1y6i4QChy9dMku25h6C7=
12 2021-05-18 17:54:49	NOT COMPLETED	PURCHASE	NULL	NULL	American Express	CREDIT CARD	NULL	NULL	NULL	hg8Kx1y6i4QChy9dMku25h6C7=
13 2021-05-18 17:55:24	NOT COMPLETED	PURCHASE	NULL	NULL	American Express	CREDIT CARD	NULL	NULL	NULL	hg8Kx1y6i4QChy9dMku25h6C7=
14 2021-05-18 17:56:06	COMPLETED	PURCHASE	6.27	USD	Discover	APPLE CASH	Whole Foods Market	WHOLEFOODS ARL 10042	Grocery Stores, Supermarkets	jAt15SpqfUKwxa81z2m0oqjC8hg=
15 2021-05-18 17:59:43	COMPLETED	APPLE CASH PIER PAYMENT	20.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
16 2021-05-18 18:06:33	COMPLETED	PURCHASE	12.86	USD	Discover	APPLE CASH	NULL	5Q "TAKESHO"	Miscellaneous General Merchandise	jAt15SpqfUKwxa81z2m0oqjC8hg=
17 2021-05-18 18:13:47	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
18 2021-05-18 18:14:03	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
19 2021-05-18 18:14:25	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
20 2021-05-18 18:15:16	COMPLETED	APPLE CASH PIER PAYMENT	15.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
21 2021-05-18 18:15:34	NOT COMPLETED	PURCHASE	NULL	NULL	Discover	APPLE CASH	NULL	NULL	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
22 2021-05-18 18:17:32	COMPLETED	APPLE CASH PIER PAYMENT	15.0	USD	Discover	APPLE CASH	NULL	NULL	NULL	jAt15SpqfUKwxa81z2m0oqjC8hg=
23 2021-05-18 18:43:44	TRANSIT	TRANSIT	NULL	NULL	CHASEBANKDISCOVER	TRANSIT	NULL	NULL	NULL	GVtC6e6z2W8hMqjC8hg=
24 2021-05-18 18:46:43	COMPLETED	PURCHASE	5.57	USD	Discover	APPLE CASH	Philz Coffee	5Q "PHILZ COFFEE	Eating Places, Restaurants	jAt15SpqfUKwxa81z2m0oqjC8hg=
25 2021-05-23 17:04:01	NOT COMPLETED	PURCHASE	0.0	USD	American Express	CREDIT CARD	NULL	Pay	NULL	hg8Kx1y6i4QChy9dMku25h6C7=

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.nt.edu	NULL
2	2021-05-13 16:14:08	15.0	USD	RECEIVED	oompa@csh.nt.edu	For groceries
3	2021-05-18 17:59:43	20.0	USD	RECEIVED	oompa@csh.nt.edu	NULL
4	2021-05-18 18:15:16	15.0	USD	RECEIVED	oompa@csh.nt.edu	NULL
5	2021-05-18 18:37:32	15.0	USD	SENT	oompa@csh.nt.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.


```

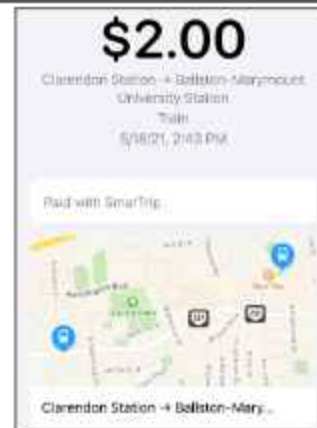
1 SELECT
2 DATETIME(PAYMENT_TRANSACTION, TRANSACTION_DATE + 978307200, 'UNIXEPOCH') AS 'TRANSACTION DATE',
3 CASE WHEN PAYMENT_TRANSACTION.AMOUNT = 9223372036854775807 THEN 'NULL' ELSE PAYMENT_TRANSACTION.AMOUNT / 100.00 END 'AMOUNT',
4 PAYMENT_TRANSACTION.CURRENCY_CODE AS 'CURRENCY CODE',
5 CASE PAYMENT_TRANSACTION.PEER_PAYMENT_TYPE
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.rlt.edu	NULL
2	2021-05-13 16:14:08	15.0	USD	RECEIVED	oompa@csh.rlt.edu	For groceries
3	2021-05-18 17:59:43	20.0	USD	RECEIVED	oompa@csh.rlt.edu	NULL
4	2021-05-18 18:15:16	15.0	USD	RECEIVED	oompa@csh.rlt.edu	NULL
5	2021-05-18 18:37:32	15.0	USD	SENT	oompa@csh.rlt.edu	Back at you

passes23.sqlite - Transit

- Supported Transit Systems
- Start & End Station Information



```

1 SELECT
2 DATETIME(PAYMENT_TRANSACTION.TRANSACTION_DATE+978307200, 'UNIXEPOCH') AS 'TRANSACTION DATE', DATETIME(PASS_RECEIVED_DATE+978307200, 'UNIXEPOCH') AS 'RECEIVED DATE',
3 PASS_ORGANIZATION_NAME AS 'ORGANIZATION NAME', PAYMENT_APPLICATION.DISPLAY_NAME AS 'CARD TYPE', PASS_UNIQUE_ID AS 'PASS ID',
4 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',
5 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',
6 FROM PAYMENT_TRANSACTION
7 JOIN PAYMENT_APPLICATION ON PAYMENT_TRANSACTION.SOURCE_ID = PAYMENT_APPLICATION.TRANSACTION_SOURCE_ID
8 JOIN PASS ON PAYMENT_APPLICATION.PASS_ID = PASS.ID

```

TRANSACTION DATE	RECEIVED DATE	ORGANIZATION NAME	CARD TYPE	PASS ID	START STATION	START STATION LAT	START STATION LONG	END STATION	END STATION LAT	END STATION LONG
2021-05-18 18:43:44	2021-05-18 18:43:55	Washington Metropolitan Area Transit Authority	Colt M Fare DEP	64C48ED7A5A96D7C6A9A254	Clarendon Station	38.836981	-77.2911347	Ballston-Marymount University Station	38.8322218	-77.1115429



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.

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 13: Apple Watch

Part 7: Reminders

This page intentionally left blank.

Section 4: Part II

Photos

This page intentionally left blank.

Photos - com.apple.Photos



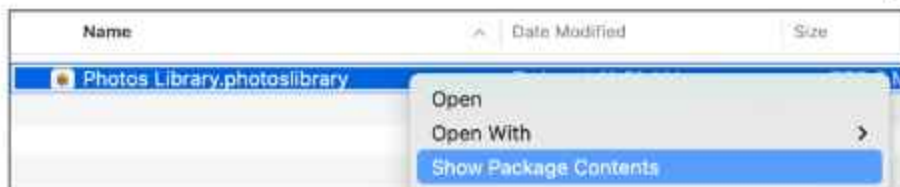
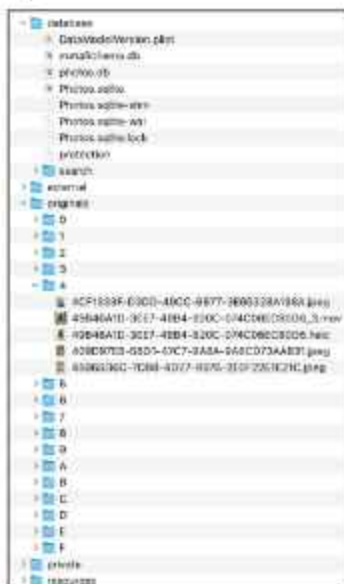
macOS: ~/Pictures/Photos Library.photoslibrary
iOS: [/private/var/]mobile/Media/PhotoData



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.

macOS Photos Library - Package "File" Format



- Presented as single file to user in Finder, right-click to view package contents
- 'originals' Directory
 - JPG: Photos
 - PNG: Screenshots
 - MOV: Movies
 - iOS 11+: *.HEIC (HEIF: High Efficiency Image Format): Open in Preview.app
- Extended Attributes:
`com.apple.quarantine` = cloudphotosd and more!

SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 168

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 many 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

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



SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 170

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.).

iOS Photo/Video Media: DCIM and Other Directories - /Media

DCIM/100APPLE/...

- ... 101APPLE ... 102APPLE ...
- 999 items per directory
- IMG_####.EXT
- Filename Sequential/Temporal Order

iCloud

- PhotoData/
- PhotoStreamsData/
- PhotoCloudSharingData/

Third-Party Application Photos

- Pro Tip - Use TCC.db

The screenshot displays the file system hierarchy of an iOS device. The 'DCIM' directory is expanded, showing a subdirectory '100APPLE' containing a list of image files (IMG_0001.PNG through IMG_0010.PNG). Another subdirectory 'PhotoCloudSharingData' is also expanded, showing a directory '1989733445' which contains a directory '2F3A64CB-A66D-48F9-A615-0C53D3550549'. This directory contains a subdirectory '100CLOUD' and two files: 'DCIM_CLOUD.plist' and 'Info.plist'.

Below the file system view, an 'Information Property List' is shown for the 'DCIM_CLOUD.plist' file. It contains the following data:

Key	Type	Value
cloudOwnerEmail	String	compa@cs.hawaii.edu
cloudOwnerFirstName	String	Sarah
cloudOwnerHashedPersonID	String	fb093b3010461f7e222edcd3f
cloudOwnerLastName	String	Edwards
cloudPublicURLEnabled	Number	1
cloudRelationshipState	Number	2
cloudSubscriptionDate	Date	2021-06-29T15:54:45Z
publicURL	String	https://www.icloud.com/shareda
title	String	Epic Adventure

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.

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 6: Calendar

Part 12: Maps

Part 7: Reminders

Part 13: Apple Watch

This page intentionally left blank.

Section 4: Part 12

Maps

This page intentionally left blank.

Maps - com.apple.Maps

macOS 11+, iOS 14+

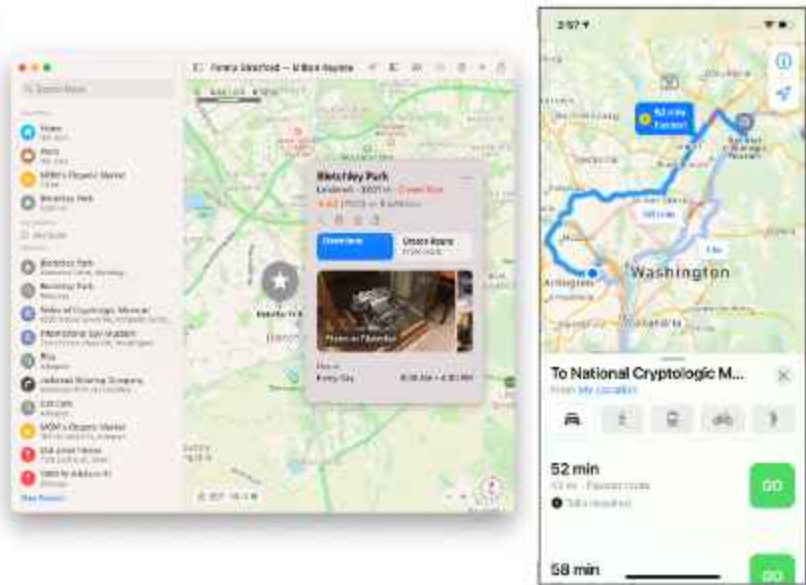
- Database

macOS 10.15-, iOS 13-

- Plist Files

Data Types

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



SANS **DFIR**

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 174

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.



Maps – Plist Files (macOS 10.15- and iOS 13-)

GeoHistory.mapsdata

- Potentially in Apple Watch directories

Map Bookmarks

- GeoBookmarks.plist

Search/Direction Details

- /ReportAProblem
- Search/<GUID>
- GraphDirections/<GUID>

Other/Older

- FailedSearches.mapsdata, MSPFailedSearches.mapsdata
- GeoPinnedPlaces.plist, GeoCollections.plist
- Pins.mapsdata, History.mapsdata, Bookmarks.mapsdata

Root	Dictionary	(2 items)
MSPHistoryVersion	Number	2
MSPHistory	Dictionary	(2 items)
records	Dictionary	(84 items)
C7397AFF-CAE6-4F4D-8E41-4817D8D88014	Dictionary	(1 item)
modificationDate	Date	2016-02-07T17:18:16Z
A834CC85-CC75-4B9F-8B97-525174391297	Dictionary	(3 items)
contentsTimestamp	Data	<8FC64245A5204F80AAE70720DC1D18600000001>
modificationDate	Date	2016-02-10T18:33:48Z
contents	Data	<080912244136332443484236204343179620944298>
E7C3F5FE-11AE-4CC9-A27C-DE3018708F9D	Dictionary	(3 items)
contentsTimestamp	Data	<AA833AF8A9FAFFDA008498A33861C1C00000001>
modificationDate	Date	2016-02-10T18:33:15Z
contents	Data	<090212244537A333A625464520313F4545203443433>
8AADD725-FAE1-4EDB-A053-0638D005E294	Dictionary	(1 item)
82A9B686-B700-42C5-B462-194FA3B376A	Dictionary	(3 items)
2061974B-CD2C-4F9F-A9EA-61437AC03295	Dictionary	(3 items)
10216AB7-0121-AC7B-A5F8-1E75C6B67956	Dictionary	(1 item)
F3AEF38B-C654-4B3D-AD09-FC57AFB2F0C2	Dictionary	(3 items)
095DA6F0-4163-862A-A340-2910E84DCB3C	Dictionary	(1 item)
899C9C29-4E2F-465D-A4D8-5388E71882DE	Dictionary	(1 item)
0607E959-21B0-48B5-AA48-E92624C93A7D	Dictionary	(1 item)

Protobuf BLOBs

SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 175

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.

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	NULL	NULL	NULL	642621434.79447	642621434.794477
2	41.5456452449917	-88.0741658806801	NULL	NULL	NULL	642621516.360473	642621516.360504
3	38.89634367153	-77.0861399173737	NULL	NULL	NULL	642621578.439475	642621578.439485
4	38.89634367153	-77.0861399173737	NULL	642621583.622226	NULL	NULL	NULL
5	38.883934378972	-77.025511264801	NULL	NULL	NULL	644011012.845177	644011012.845219
6	39.1148200530054	-76.7746871709824	NULL	NULL	NULL	644011023.729719	644011023.729728
7	41.9475159	-87.6562758	NULL	644011136.511095	1060 W Addison Chicago IL 60613 United States	NULL	NULL
8	41.5463347	-88.0733369	NULL	644011136.511632	1125 Collins St Joliet IL 60432 United States	NULL	NULL
9	51.99775642662	-0.741491317749023	NULL	NULL	NULL	644011262.225946	644011262.22595
10	51.99775642662	-0.741491317749023	NULL	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.

Decoding Map Protobuf BLOBs

```
oomp@m3p-m1 maps % protoc --decode_raw < bletchley.pb
```

```
1 {
  1: 13780109867912023867
  2: 0
  4 {
    1: 15
    2: 0
    4: 43200
    5: 0
    6: 1
    8 {
      15 {
        1 {
          1: 21
          2: 1
        }
        1 {
          1: 3
          2: 1
        }
      }
      2 {
        2: 1
        4: "crossbusiness.goodfor.families"
        5 {
          1: "en"
          3: "Good for Kids"
        }
        6: "kidFriendly"
      }
    }
  }
}
```

```

47 43288
48 0
49 1
50 1
51 1
52 "http://www.wikiipedia.org/wiki/Witchology_Fark"
53 "11528"
54 1
55 0
56 1
57 1
58 1
59 1
60 1
61 1
62 1
63 1
64 1
65 1
66 1
67 1
68 1
69 1
70 1
71 1
72 1
73 1
74 1
75 1
76 1
77 1
78 1
79 1
80 1
81 1
82 1
83 1
84 1
85 1
86 1
87 1
88 1
89 1
90 1
91 1
92 1
93 1
94 1
95 1
96 1
97 1
98 1
99 1
100 1
101 1
102 1
103 1
104 1
105 1
106 1
107 1
108 1
109 1
110 1
111 1
112 1
113 1
114 1
115 1
116 1
117 1
118 1
119 1
120 1
121 1
122 1
123 1
124 1
125 1
126 1
127 1
128 1
129 1
130 1
131 1
132 1
133 1
134 1
135 1
136 1
137 1
138 1
139 1
140 1
141 1
142 1
143 1
144 1
145 1
146 1
147 1
148 1
149 1
150 1
151 1
152 1
153 1
154 1
155 1
156 1
157 1
158 1
159 1
160 1
161 1
162 1
163 1
164 1
165 1
166 1
167 1
168 1
169 1
170 1
171 1
172 1
173 1
174 1
175 1
176 1
177 1
178 1
179 1
180 1
181 1
182 1
183 1
184 1
185 1
186 1
187 1
188 1
189 1
190 1
191 1
192 1
193 1
194 1
195 1
196 1
197 1
198 1
199 1
200 1
201 1
202 1
203 1
204 1
205 1
206 1
207 1
208 1
209 1
210 1
211 1
212 1
213 1
214 1
215 1
216 1
217 1
218 1
219 1
220 1
221 1
222 1
223 1
224 1
225 1
226 1
227 1
228 1
229 1
230 1
231 1
232 1
233 1
234 1
235 1
236 1
237 1
238 1
239 1
240 1
241 1
242 1
243 1
244 1
245 1
246 1
247 1
248 1
249 1
250 1
251 1
252 1
253 1
254 1
255 1
256 1
257 1
258 1
259 1
260 1
261 1
262 1
263 1
264 1
265 1
266 1
267 1
268 1
269 1
270 1
271 1
272 1
273 1
274 1
275 1
276 1
277 1
278 1
279 1
280 1
281 1
282 1
283 1
284 1
285 1
286 1
287 1
288 1
289 1
290 1
291 1
292 1
293 1
294 1
295 1
296 1
297 1
298 1
299 1
300 1
301 1
302 1
303 1
304 1
305 1
306 1
307 1
308 1
309 1
310 1
311 1
312 1
313 1
314 1
315 1
316 1
317 1
318 1
319 1
320 1
321 1
322 1
323 1
324 1
325 1
326 1
327 1
328 1
329 1
330 1
331 1
332 1
333 1
334 1
335 1
336 1
337 1
338 1
339 1
340 1
341 1
342 1
343 1
344 1
345 1
346 1
347 1
348 1
349 1
350 1
351 1
352 1
353 1
354 1
355 1
356 1
357 1
358 1
359 1
360 1
361 1
362 1
363 1
364 1
365 1
366 1
367 1
368 1
369 1
370 1
371 1
372 1
373 1
374 1
375 1
376 1
377 1
378 1
379 1
380 1
381 1
382 1
383 1
384 1
385 1
386 1
387 1
388 1
389 1
390 1
391 1
392 1
393 1
394 1
395 1
396 1
397 1
398 1
399 1
400 1
401 1
402 1
403 1
404 1
405 1
406 1
407 1
408 1
409 1
410 1
411 1
412 1
413 1
414 1
415 1
416 1
417 1
418 1
419 1
420 1
421 1
422 1
423 1
424 1
425 1
426 1
427 1
428 1
429 1
430 1
431 1
432 1
433 1
434 1
435 1
436 1
437 1
438 1
439 1
440 1
441 1
442 1
443 1
444 1
445 1
446 1
447 1
448 1
449 1
450 1
451 1
452 1
453 1
454 1
455 1
456 1
457 1
458 1
459 1
460 1
461 1
462 1
463 1
464 1
465 1
466 1
467 1
468 1
469 1
470 1
471 1
472 1
473 1
474 1
475 1
476 1
477 1
478 1
479 1
480 1
481 1
482 1
483 1
484 1
485 1
486 1
487 1
488 1
489 1
490 1
491 1
492 1
493 1
494 1
495 1
496 1
497 1
498 1
499 1
500 1
501 1
502 1
503 1
504 1
505 1
506 1
507 1
508 1
509 1
510 1
511 1
512 1
513 1
514 1
515 1
516 1
517 1
518 1
519 1
520 1
521 1
522 1
523 1
524 1
525 1
526 1
527 1
528 1
529 1
530 1
531 1
532 1
533 1
534 1
535 1
536 1
537 1
538 1
539 1
540 1
541 1
542 1
543 1
544 1
545 1
546 1
547 1
548 1
549 1
550 1
551 1
552 1
553 1
554 1
555 1
556 1
557 1
558 1
559 1
560 1
561 1
562 1
563 1
564 1
565 1
566 1
567 1
568 1
569 1
570 1
571 1
572 1
573 1
574 1
575 1
576 1
577 1
578 1
579 1
580 1
581 1
582 1
583 1
584 1
585 1
586 1
587 1
588 1
589 1
590 1
591 1
592 1
593 1
594 1
595 1
596 1
597 1
598 1
599 1
600 1
601 1
602 1
603 1
604 1
605 1
606 1
607 1
608 1
609 1
610 1
611 1
612 1
613 1
614 1
615 1
616 1
617 1
618 1
619 1
620 1
621 1
622 1
623 1
624 1
625 1
626 1
627 1
628 1
629 1
630 1
631 1
632 1
633 1
634 1
635 1
636 1
637 1
638 1
639 1
640 1
641 1
642 1
643 1
644 1
645 1
646 1
647 1
648 1
649 1
650 1
651 1
652 1
653 1
654 1
655 1
656 1
657 1
658 1
659 1
660 1
661 1
662 1
663 1
664 1
665 1
666 1
667 1
668 1
669 1
670 1
671 1
672 1
673 1
674 1
675 1
676 1
677 1
678 1
679 1
680 1
681 1
682 1
683 1
684 1
685 1
686 1
687 1
688 1
689 1
690 1
691 1
692 1
693 1
694 1
695 1
696 1
697 1
698 1
699 1
700 1
701 1
702 1
703 1
704 1
705 1
706 1
707 1
708 1
709 1
710 1
711 1
712 1
713 1
714 1
715 1
716 1
717 1
718 1
719 1
720 1
721 1
722 1
723 1
724 1
725 1
726 1
727 1
728 1
729 1
730 1
731 1
732 1
733 1
7
```

[illegible]

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.

Section 4: Agenda

Part 1: Application Fundamentals

Part 8: Contacts

Part 2: Introduction to SQLite Queries

Part 9: Notes

Part 3: Safari Browser

Part 10: Wallet and Apple Pay

Part 4: Apple Mail

Part 11: Photos

Part 5: Communication

Part 12: Maps

Part 6: Calendar

Part 7: Reminders

Part 13: Apple Watch

This page intentionally left blank.

Section 4: Part 14

Apple Watch

This page intentionally left blank.

Apple Watch

Usually paired with iPhone, proxy device

Wi-Fi, GPS, NFC, Bluetooth, Cellular

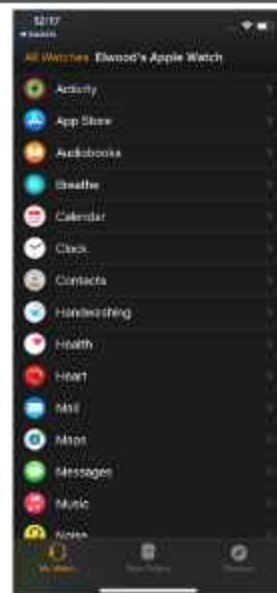
“System in Package” Chip

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.

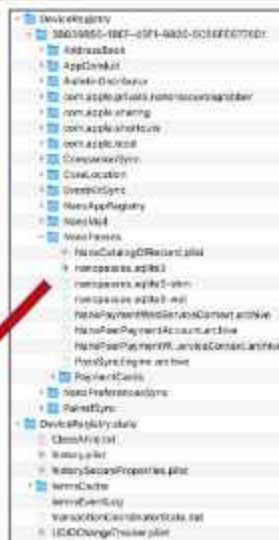
Apple Watch - Data Directory /mobile/Library/DeviceRegistry/<GUID>

DeviceRegistry/

- App Data

DeviceRegistry.state/

- historySecureProperties.plist
- Serial, IMEI, Bluetooth/Wi-Fi MAC addresses, etc.



```
1 select
2 unique_id, type_id, encoded_pass, ingested_date, localized_description from pass
```

	unique_id	type_id	encoded_pass	ingested_date	localized_description
1	-foqul/WkOyF5Xm2oBqgy5QnY+	pass.eventbrite.ticket	---	644018923	Event Ticket for College Park Farmer's Market @ Port
2	OYMF9KQeQmM6U3mymM-639d9KwQ+	pass.com.starbucks.card	---	644018923	Starbucks Card
3	mbG7Yu24splogW7V0T0BACj8cou	pass.eventbrite.ticket	---	644018923	Event Ticket for SUNDAY BRUNCH & DAY PARTY
4	TPQ7NBAjLWV6QmsVcxQe55h6g+	pass.eventbrite.ticket	---	644018923	Event Ticket for Sunday jazz brunch
5	ja1b5pqtUeWqR1s2mMOejIG2g+	paymentpass.com.apple	---	644018923	Apple Cash

SANS DFIR

FOR518.4 | Mac and iOS Forensic Analysis and Incident Response 181

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.

Lab 4.4 - Applications: Part II

Choose your own adventure!

This page intentionally left blank.