

518.1 Mac and iOS Essentials

SANS

518.1

Mac and iOS

Essentials



Copyright © 2020, Sarah Edwards. All rights reserved to Sarah Edwards and/or SANS Institute.

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

With the 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, YOU AGREE TO BE BOUND BY THE TERMS OF THIS CLA. BY ACCEPTING THIS SOFTWARE, YOU AGREE 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 you do not agree, you 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 marks 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.




FOR518 Section I: Mac and iOS Essentials


© 2020 Sarah Edwards | All Rights Reserved | Version F01_01


Author: Sarah Edwards
oompa@csh.rit.edu
mac4n6.com
<http://twitter.com/iamevltwin>

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

SANS DFIR
DIGITAL FORENSICS & INCIDENT RESPONSE


FOR498
Battlefield Forensics & Data Acquisition



FOR500
Windows Forensic Analysis
GCFA



FOR518
Mac and iOS Forensic Analysis and Incident Response



FOR526
Advanced Memory Forensics & Threat Detection


FOR585
Smartphone Forensic Analysis In-Depth
GASF


FOR508
Advanced Incident Response, Threat Hunting, and Digital Forensics
GCFA


FOR572
Advanced Network Forensics: Threat Hunting, Analysis, and Incident Response
GNFA



FOR578
Cyber Threat Intelligence
GCTI



FOR610
REM: Malware Analysis Tools and Techniques
GREM


SEC504
Hacker Tools, Techniques, Exploits, and Incident Handling
GCIH



[@sansforensics](https://twitter.com/sansforensics)


[sansforensics](https://www.facebook.com/sansforensics)


[dfir.to/DFIRCast](https://www.youtube.com/channel/UCdFIR)


[dfir.to/MAIL-LIST](mailto:dfir@lists.sans.org)

This page intentionally left blank.

Lab 1.0: Lab Setup

WELCOME to FOR518: Mac and iOS Forensic Analysis and Incident Response!

Students, please start this lab in the Workbook before class.

Please let your instructor know if you had issues with the pre-class setup (Lab 0/Laptop Setup) or with this lab (Lab 1.0).

This page intentionally left blank.

Class Minutiae

Class Timing

Facilities (SANS Office / Restroom)

Cellphone Usage

Wi-Fi Password

Books/Workbook/Handouts

Class Notebook: for518.com/notebook

@Nights

Class Introductions

This page intentionally left blank.

Course Agenda

Section 1: Mac and iOS Essentials

Section 2: File Systems and System Triage

Section 3: User Data, System Configuration, and Log Analysis

Section 4: Application Data Analysis

Section 5: Advanced Analysis Topics

Section 6: Mac Forensic Challenge

This page intentionally left blank.

Mac and iOS Essentials

This page intentionally left blank.

Section I: Agenda

Part 1: Apple Essentials

Part 2: Mac Essentials and Acquisition

Part 3: iOS Essentials and Acquisition

Part 4: Disks and Partitions

This page intentionally left blank.

Section I: Part I

Apple Essentials

This page intentionally left blank.

Apple and Mac History

Apple Computer was established on April 1, 1976, in Cupertino, CA, by:

- Steve Jobs, Steve Wozniak, Ronald Wayne

Apple I (\$666.66)

First Macintosh released in 1984 (\$2,500)

macOS (Mac OS X) released in 2001, iOS in 2007

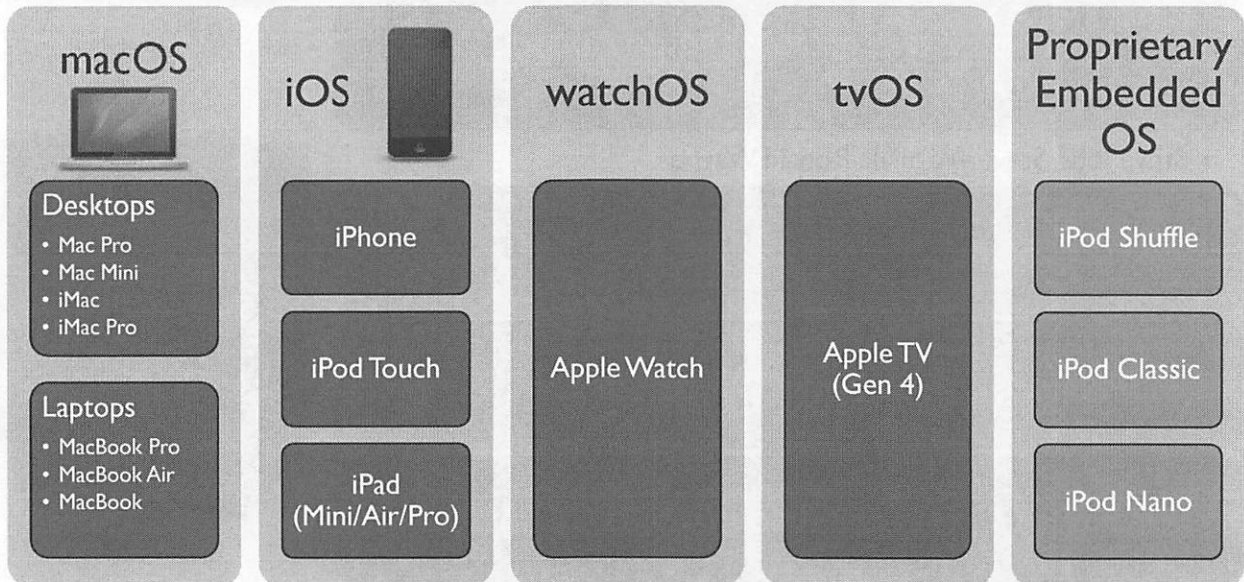


Apple Computer was established on April 1, 1976, (incorporated January 3, 1977) by Steve Jobs, Steve Wozniak, and Ronald Wayne.

The first computer they sold was the Apple I for \$666.66 in 1976. In 1984, the first Macintosh was introduced to the Super Bowl audience with the famous “1984” commercial. (<https://www.youtube.com/watch?v=2zfqw8nhUwA>) The Macintosh was touted as the first commercially successful personal computer with a graphical user interface. Also, it was \$2,500!

Macintosh computers use the Mac Operating System (macOS). The older Macs used the “Classic” macOS (System 1–9), while the newer Macs use Mac OS X (Ten)/macOS. Mac OS X introduced a Unix-based operating system in 2001. iOS, the mobile platform, was introduced in 2007.

Mac and iOS Systems



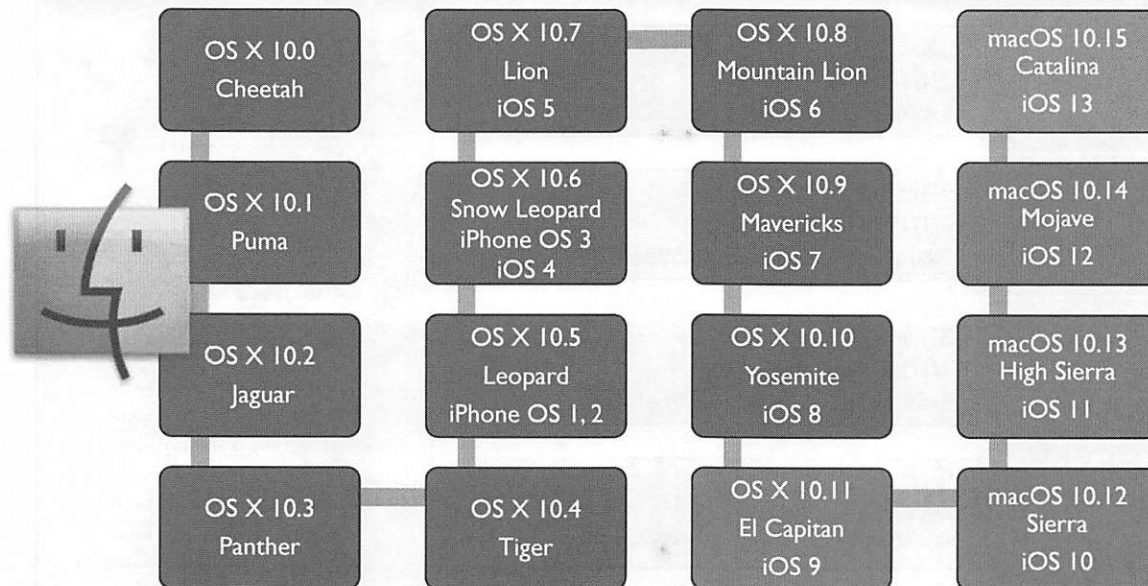
Modern Mac Systems are comprised of three types: macOS, iOS, and a proprietary embedded operating system.

macOS is featured on the desktop and laptop product lines. iOS is featured on mobile devices. iOS is based upon macOS, and could be considered macOS "lite". It has many of the same characteristics and file system intricacies but does not contain all the software and features of macOS.

Most of the mobile devices manufactured by Apple implement iOS, such as the iPhone, iPad, and iPod Touch, as well as the newer generations of Apple TV. (Fun fact, the original Apple TV implemented the full version of macOS). Other Apple mobile devices, such as the iPod Shuffle, Nano, or Classic, use a proprietary embedded operating system that will not be covered in this course.

Starting with the Apple Watch and Gen 4 Apple TV, we now have WatchOS and tvOS to be on the lookout for. While documentation is scarce, these appear to be similar to iOS.

Mac and iOS Versions



There have been many versions of OS X/macOS. The latest OSs, Catalina (10.15) and iOS 13, were released in October 2019. The most recent versions of each operating system are generally released around the same time in the Autumn months.

OS X introduced the Unix-based operating system with an easy-to-use GUI for the casual user. iOS has the same Unix history as well, and is also based on OS X.

Each version of the operating system changes slightly and can differ in various ways. Where appropriate, the version of the operating system will be listed, but be aware that changes may vary, and testing the various operating systems may help you with specific questions.

For the purposes of this class, OS X and macOS will be used interchangeably to call out the desktop (versus mobile) version of the operating system.

Timestamp Formats, BBT Epoch Converter, and Epochalypse.py

HFS+ Dates or
Mac OS

- 32bit
- 1/1/1904 00:00:00 UTC
- Example: 3487322096 (0xCFDC4FF0)

Unix Epoch

- 32bit and 64bit (APFS)
- 1/1/1970 00:00:00 UTC
- Example: 1404477296 (0x53B69F70)

Mac Epoch or
Mac Absolute or
Cocoa or WebKit

- 32bit
- 1/1/2001 00:00:00 UTC
- Example: 426170096 (0x1966D6F0)

Epochalypse.py

- Free command-line converter
- 64bit Unix epoch capability
- <https://github.com/pstirparo/utlis/blob/master/epochalypse.py>

The screenshot shows the Epoch Converter web application. At the top, it displays the current Mac epoch time as 3569877897. Below this, it shows the local time as 2/13/17 7:44:57 PM and the UTC time as 2/14/17 12:44:57 AM. The main interface features a dropdown menu to select the conversion direction: Integer to Date (selected) or Date to Integer. An input field contains the epoch value 1234567890, with a 'Convert' button to its right and a 'Clear All' button below it. The results are displayed in a list of date and time conversions for various systems:

System	Date
Mac OS Date	1943-02-13 19:31:30 Sat EWT
UTC	1943-02-13 23:31:30 Sat UTC
Unix Date	2009-02-13 18:31:30 Fri EST
UTC	2009-02-13 23:31:30 Fri UTC
Cocoa/WebKit Date	2040-02-14 18:31:30 Tue EST
UTC	2040-02-14 23:31:30 Tue UTC
Google Chrome Date	1600-12-31 19:20:34 Sun EST
UTC	1601-01-01 00:20:34 Mon UTC
Mozilla Firefox Date	1969-12-31 19:20:34 Wed EST
UTC	1970-01-01 00:20:34 Thu UTC
Microsoft Date	1600-12-31 19:02:03 Sun EST
UTC	1601-01-01 00:02:03 Mon UTC

At the bottom, there are checkboxes for 'Convert Multiple Epochs:' and 'Choose Epochs:'.

SANS | DFIR

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 12

You will encounter a variety of timestamps on a Mac system. While Unix epoch is relatively well known (the number of seconds from midnight on 1/1/1970), the HFS+ file system uses a different epoch date (midnight on 1/1/1904). Another epoch time, sometimes called Mac Absolute Time, starts at midnight on 1/1/2001.

Throughout the class, the various timestamps will be highlighted.

Convert timestamps using `date -ur <epochtime>`:

- Mac Epoch: add 978307200
- HFS+ Date: subtract 2082844800

For example:

```
$ date -ur 1234567890
Fri Feb 13 23:31:30 UTC 2009
```

BlackBag Technologies has a wonderful GUI-based timestamp converter. It has the ability to convert from timestamp integer to date, and vice versa.

This utility supports many timestamp formats in local system time and UTC:

- Mac OS (HFS+)
- Unix Epoch
- Cocoa/WebKit
- Google Chrome
- Mozilla Firefox
- Microsoft

This application is similar to Dcode by Digital Detective for Windows-based systems.

Reference:

Mac Developer Library – NSDate Class Reference

<https://developer.apple.com/documentation/foundation/nsdate>

* *

SQLite Databases

SQL-based relational database

File Signature: "SQLite format 3"

DB Browser for SQLite (PC/Mac)

Command Line: sqlite3 (Mac)

Database Coalescing (*.shm, *.wal)

File extensions: *.db, *.sqlite, *.storedata, *.sqlitedb, anything or nothing!

```
00000000: 5351 4c69 7465 2066 6f72 6d61 7420 3300  SQLite format 3.  
00000010: 1000 0101 0040 2020 0000 086f 0000 00b2  .....@ ...0....  
00000020: 0000 0000 0000 0000 0000 0002 0000 0001  .....  
00000030: 0000 0000 0000 0000 0000 0001 0000 0000  .....  
00000040: 0000 0000 0000 0000 0000 0000 0000 0000  .....
```

Software on Mac systems uses SQLite databases to store various types of information, such as settings and internet histories. SQLite is a SQL-based relational database. These databases can have multiple tables with multiple types of keys and values.

In the screenshot above, the signature used for SQLite databases is "SQLite format 3". An easy way to carve for these files would be to use that signature as a starting point.





SQLite files are non-system specific and can be viewed using different tools for Mac, Windows, or Linux systems. You may want to use DB Browser for SQLite. This application was recently updated and now handles newer SQLite files that it was not able to do before.

It is worth noting that SQLite database files may consist of not just the main database, but also a Shared Memory file (*.shm) and a Write Ahead Log (*.wal). To get the complete database you will want to extract all these files and coalesce them using a SQLite viewer. Not all tools do this, however DB Browser for SQLite does.





SQLite databases may have a variety of file extensions or none at all. Be sure to check the file signature to ensure it is a SQLite database.

SQLite Databases: DB Browser for SQLite

Database Structure Browse Data Edit Pragma Execute SQL

Table:    

	id	url	domain_expansion	visit_count	daily_visit_counts	weekly_visit_counts	incomplete_trigge	compute_derived_v	visit_count_score
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	https://learning.oreilly.com/subs...	learning.oreilly	1	BLOB	NULL	NULL	0	20
2	2	https://www.marriott.com/hotels...	marriott	2	BLOB	NULL	NULL	0	25
3	3	https://my.silversea.com/Account...	my.silversea	1	BLOB	NULL	NULL	0	20
4	4	https://www.google.com/search?...	google	3	BLOB	NULL	NULL	0	20
5	5	https://www.zappos.com/fdcup-...	zappos	1	BLOB	NULL	NULL	0	20

Table:    

	id	history_item	visit_time	title	load_successful	http_non_get	synthesized	redirect_source	redirect_destination	origin	generation	attributes	score
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	301	591441618.719632	Failed to open page	0	0	0	NULL	NULL	1	2	0	20
2	2	301	591441610.314028	Failed to open page	0	0	0	NULL	NULL	1	2	2	0
3	3	301	591441648.728751	Failed to open page	0	0	0	NULL	NULL	1	2	2	0
4	4	301	591441721.49229	for518 - Hiedat Coo...	0	0	0	NULL	5	1	2	4	5

These screenshots shows examples of SQLite Database Browser.

Each dropdown table may contain different types of data. This is an example of the Safari browsing history. The `history_items` and `history_visits` tables are the two most important in this database. To tie these two tables together we can use certain columns, in this example each table uses an `id` for each website visit.

Columns may contain a variety of different data types; strings, numbers, etc. BLOBs contain Binary Large Objects which may contain anything in binary; plists, pictures, protobufs, etc. Other numbers you may come across are timestamps. The example highlighted is a Mac Epoch timestamp.

This is a standalone application that does not require a browser. Each analyst may have their own favorite SQLite database viewer. Please choose what you prefer to use.

Property List Files (Plists)

XML – File Signature: “<?xml version=”

Binary – File Signature: “bplist00”

Open on Mac: Xcode, plutil

- New on 10.15 – Truncated BLOBs, use PlistBuddy (or plistutil from libmobiledevice or jlutil from Jonathan Levin)

Open on PC: iBackupBot Plist Editor, some commercial forensic suites



```
0000000: 3c3f 786d 6c20 7665 7273 696f 6e3d 2231 <?xml version="1
0000010: 2e30 2220 656e 636f 6469 6e67 3d22 5534 .0" encoding="UT
0000020: 462d 3822 3f3e 0a3c 2144 4f43 5450 5045 F-8"?>.<!DOCTYPE
0000030: 2070 6c69 7374 2050 5542 4c49 4320 222d plist PUBLIC "-

0000000: 6270 6c69 7374 3030 d501 0203 0405 0601 bplist00.....
0000010: 0c0d 0c5f 1010 546f 6767 6c65 436f 6e74 ..._.ToggleCont
0000020: 726f 6c4b 6579 5f10 1c44 6f6e 7451 7569 rolKey_..DontQui
0000030: 7457 6865 6e4c 6173 7457 696e 646f 7743 tWhenLastWindowC
```

SANS | **DFIR**

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 16

Property list files usually contain configuration data, similar to the Windows registry. These files come in two formats, XML or binary. The older format, XML is human-readable and has the file signature of “<?xml version=”. The newer binary format is used, as it is more efficient on disk space. This format has the file signature “bplist00”, for binary plist.

These files can be read using the Xcode or by the command-line tool `plutil`.

Xcode is not installed by default on Mac OS X, It can be installed by downloading it from Apple’s website, or by using the Mac App Store. Older versions of Xcode had a utility named “Property List Editor” that reads property list files without the extraneous bulk of Xcode. Newer versions of Xcode include a property list editor, but it may take a while to load the whole Xcode programming environment.

A command-line tool, `plutil`, can be used to convert property list files into other formats, such as binary to XML, or XML to JSON.

With 10.15 Catalina, Xcode and `plutil` still work well but have changed. If you are looking into embedded BLOB data you will need to use a different utility as these will now be truncated on newer OSs.

On Windows systems, there is one tool that the author has found to parse the property list files. It doesn’t work nearly as well as the Mac tools, but it works in a pinch. Many commercial forensic suites can parse these out too!

iBackupBot Plist Editor – <http://www.icopybot.com/download.htm>

Property List Files: Xcode Example

Xcode Tips and Tricks:

Option+Click Gray Triangles

- Expand/contract nested items

Command+F

- Find a keyword within plist file

Filter Plist Files

- Available when multiple plists are open in same window
- Filter by keyword in plist filename

Key	Type	Value
Root	Dictionary	(15 items)
SkipSystemFiles	Boolean	NO
ExcludeByPath	Array	(3 items)
BackupAlias	Data	<00000000 03980002 00010444 61746100 00000000>
MobileBackups	Boolean	YES
AutoBackup	Boolean	NO
AlwaysShowDeletedBackupsWarning	Boolean	NO
IncludeByPath	Array	(7 items)
LocalizedDiskImageVolumeName	String	Time Machine Backups
SkipPaths	Array	(0 items)
LastCompactTime	Number	1,398,670,537
HostUUIDs	Array	(2 items)
RootVolumeUUID	String	2603DEB0-8EBD-368D-ASF4-98946F8EE01
LastDestinationID	String	BBC17CF7-F535-4A88-9132-85DD483CD352
Destinations	Array	(1 items)
Item 0	Dictionary	(9 items)
RESULT	Number	7
BytesUsed	Number	1,954,026,766,336
DestinationUUIDs	Array	(2 items)
SnapshotDates	Array	(1 items)
BackupAlias	Data	<00000000 03980002 00010444 61746100 00000000>
DateOfLatestWarning	Date	Apr 28, 2014, 2:25:18 AM
BytesAvailable	Number	43,880,669,184
MessageParameters	Array	(2 items)
DestinationID	String	BBC17CF7-F535-4A88-9132-85DD483CD352
PreferencesVersion	Number	4

An example of a property list is shown in the screenshot above. This property list is being viewed with the internal Xcode plist viewer.

Each property list contains keys and values. Each value can have different data types, such as:

- Boolean: On/Off, Yes/No, 0/1
- Array: Contains additional keys/values
- Data: Binary Data Blob
 - Can be extracted and viewed in a hex editor
 - May contain embedded property lists
- Number
- String
- Date: Dates are shown in local host system time (make sure to check your time zones!)
- Dictionary: Contains additional keys/values (similar to Array)

Key	Type	Value
RecentsArrangeGroupViewBy	String	Date Last Opened
FXICloudDriveFirstSyncDownComplete	Boolean	YES
SearchRecentsSavedViewStyleVersion	String	%00%00%00%01
FXICloudDriveDocuments	Boolean	YES
NetworkViewSettings	Dictionary	(1 item)
CustomViewStyleVersion	Number	1
SidebarSharedSectionDisclosedState	Boolean	YES
FXICloudDriveEnabled	Boolean	YES
ShowSidebar	Boolean	YES
ComputerViewSettings	Dictionary	(4 items)
WindowState	Dictionary	(8 items)
ContainerShowSidebar	Boolean	YES
WindowBounds	String	{{335, 330}, {770, 436}}
ShowTabView	Boolean	NO

Property List Files: "plutil -p" Example

```
word:Preferences sledwards$ plutil -p com.apple.TimeMachine.plist | less
```

```
{
  "SkipSystemFiles" => 0
  "ExcludeByPath" => [
    0 => "/Users/Shared/adi"
    1 => "/Users/Shared/adi"
    2 => "/Library/Application Support/Microsoft/PlayReady"
  ]
  "BackupAlias" => <00000000 03980002 00010444 61746100 00000000 00000000
0000 00000000 00000000 00000000 00000000 0002caae 2d3e0000 00000000 0000f
0061000f 000a0004 00440061 00740061 00120000 0013000f 2f566f6c 756d6573 2
0000 00000000 00000844 656c6f72 65616e00 00000000 00000000 00000000 00000
000005f 6f6d7061 00000000 00000000 00000000 00000000 00000000 00000000 0
00 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0000000
000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 000
00 00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000 0000
"MobileBackups" => 0
"AutoBackup" => 0
"AlwaysShowDeletedBackupsWarning" => 0
  "IncludeByPath" => [
    0 => "/Applications"
    1 => "/Library"
    2 => "/System"
    3 => "/bin"
    4 => "/private"
    5 => "/sbin"
    6 => "/usr"
  ]
}
```

BLOBs
truncated in
10.15

You can also view property lists on the command line using the OS X native tool `plutil`. Use the `-p` parameter to "print" the property list. The default print view is JSON formatted.

You can also use `plutil` to convert property lists from binary to XML.

Reference:
[plutil Man Page](#)

10.15 – Truncated BLOBs with Xcode & plutil Native Utility: /usr/libexec/PlistBuddy (Path & Capitalization Matters!)

▼ FXRecentFolders	Array	(10 items)
▼ Item 0	Dictionary	(2 items)
file-bookmark	Data	{length = 756, bytes = 0x626f666b f4020000 00000410 30000000 ... 04000000 00000000 }
name	String	modules



```
oompa@Sarahs-Air Preferences % /usr/libexec/PlistBuddy -c Print:FXRecentFolders:0:file-bookmark com.apple.finder.plist | xxd
00000000: 626f 6f6b f402 0000 0000 0410 3000 0000  book.....0...
00000010: 0000 0000 0000 0000 0000 0000 0000 0000  .....
00000020: 0000 0000 0000 0000 0000 0000 0000 0000  .....
00000030: e401 0000 0400 0000 0303 0000 0002 0020  .....
00000040: 0500 0000 0101 0000 5573 6572 7300 0000  .....Users...
00000050: 0500 0000 0101 0000 6f6f 6d70 6100 0000  .....oompa...
00000060: 0900 0000 0101 0000 446f 776e 6c6f 6164  .....Download
00000070: 7300 0000 0d00 0000 0101 0000 4150 4f4c  s.....APOL
00000080: 4c4f 2d6d 6173 7465 7200 0000 0700 0000  LO-master.....
00000090: 0101 0000 6d6f 6475 6c65 7300 1400 0000  ...modules....
000000a0: 0106 0000 1000 0000 2000 0000 3000 0000  .....0...
000000b0: 4400 0000 5c00 0000 0800 0000 0403 0000  D...\.....
```

Now that Xcode and plutil will truncate BLOBs, we need to be creative to read them.

One native tool that can extract their data is PlistBuddy located in /usr/libexec/. When executing this on your system, you will need to use the full path to the utility as it is not located in your \$PATH. Also be sure to capitalize the P and B!

The example above extracts a single BLOB from a specified key.

- -c – Command, we are using Print to see the value.
 - Following Print, we use colons to specify a key. We want the file-bookmark key from Item 0 under the FXRecentFolders key.

This will print the binary data from this key, so we use xxd here to look at it in a hex format.

BlackLight Introduction

Forensic suite specially designed for Mac analysis (will also analyze Windows systems!)

Runs natively on Mac or Windows systems

Image Formats: dd, dmg, sparsebundle, sparseimage, vmdk, E01, L01, Cellebrite, Elcomsoft, GreyKey, etc.

Logical extraction from iDevices



SANS DFIR

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 21

We will be using the BlackLight forensic suite in class for some of the labs; however, it will be available during the whole of the class only via the Network License Server running on your instructor's laptop.

The suite runs natively on both Mac and Windows-based systems (normally with a USB dongle, or a Network License Server setup at your agency).

It also has cross-platform capability—the software can also analyze Windows file systems—which is convenient when a system has a Boot Camp partition. Most disk image formats are accepted, as well as many from the most popular iDevice acquisition suites.

While BlackLight is not an acquisition tool (MacQuisition is used for acquisition), it is able to acquire a logical extraction from iDevices. The BlackLight software is available from BlackBag Technologies. <http://blackbagtech.com>.

The Browser tab shows the file system as an investigator is most likely used to seeing it. This view shows the file system in a tree format with hidden files shown in gray, and other file metadata including timestamps and file size.

The lower pane (the bar may have to be moved up from the bottom of the window) shows the file. The views available include Hex, Strings, Preview, Metadata, Location, and Record. An analyst may also select the “eye”-shaped button to do a “Quick Look” on the file. The Data and Resource fork may also be chosen. In the Hex view, the data-type window on the right will be shown for the analyst to select various data types, if conversion is needed.

In the lower-left pane, the file metadata and extended attributes are shown. Everything from file size, filename, timestamps, Finder data, and disk location, to extended attributes are available in this window. Lots of good information may be found here!

Section I: Agenda

Part 1: Apple Essentials

Part 2: Mac Essentials and Acquisition

Part 3: iOS Essentials and Acquisition

Part 4: Disks and Partitions

This page intentionally left blank.

Section 1: Part 2

Mac Essentials and Acquisition

This page intentionally left blank.

macOS File System Domains

User	<ul style="list-style-type: none"> • User-Specific Files • Controlled by Each User • Hidden ~/Library Directory
Local	<ul style="list-style-type: none"> • Apps/Resources for Local System and Users • Controlled by System and Admin Users • /Library
System	<ul style="list-style-type: none"> • System Software Installed by Apple • Controlled by System • /System/Library
Network	<ul style="list-style-type: none"> • Apps/Resources on Local Network • Controlled by Network Administrator • Other systems, printers, Time Capsules, NAS, etc.

SANS | DFIR FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 24

The Mac OS X file system is made up of four domains: User, Local, System, and Network.

Each domain has a purpose; they may contain user files, system resources, or network data. The domains are used to separate files based on their usage. The breakdown of the domains helps implement access controls so a file is not intentionally or unintentionally modified. This is done to keep the system secure and functional.

The User Domain consists of all their user files; their documents, pictures, music, etc. Each user has their own home directory in the `/Users` folder, marked by a house icon in the screenshot above. If a user is logged on, default permissions will not allow them to view the files in another user's directory, noted by the red circle with the line icon. Most of the directories are self-explanatory in terms of the information they may contain. The public folder may contain data the user wants to share with other users. The user's Library directory contains app-specific data. This domain may also contain a "Sites" directory if web sharing is enabled. This would contain the user's personal website. Starting with Mac OS X Lion, the user's Library folder was hidden. Some users may prefer to be able to view the files easily. The author finds that it helps forensic research to have the Library directory viewable. The command to permanently change this for a specific account is below. It may also be accessed by holding down the "option" key when accessing it via the 'Go' menu in the Finder Toolbar.

```
chflags nohidden /Users/<username>/Library
```

The local domain consists of the applications folder, the (local) Library directory, and if installed, the Developer directory. This domain is used to store the files that may be shared among the users, such as applications. The `/Applications` directory contains the applications available to all users of the system. It is not necessary to run an application from the `/Applications` directory; it may be run from a user directory if needed. Applications from the Mac App Store are installed in the `/Applications` directory. The `/Developer` directory is created when Xcode is installed. The Developer directory may be located in either the root (`/`), under `/Library`, or embedded in the Xcode.app application in the `/Applications` directory. The Developer folder contains Apple Developer (Mac, iOS) related data and resources.

The System Domain is used to store Apple-specific system software. The System Domain contains the System Library directory, which contains files associated with Apple system resources. The primary files we will be looking at in this domain are system preferences and data files.

You may have noticed a common theme in the directory structure. There are three Library directories on Mac OS X, each with its own purpose. It is easy to get confused about which Library directory contains what data, and this class will discuss the difference between each Library directory and the contents located within.

- User Library: `/Users/<username>/Library/` or the shortcut `~/Library` as noted in the slides in this course.
- Local Library: `/Library/`
- System Library: `/System/Library/`

The Network Domain contains the network resources, such as network area storage, Time Capsules, printers, file shares, and other systems on the network.

Reference:

File System Programming Guide – File System Basics

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

User's Home Directory

```
Elwoods-Mac:~ elwoodblues$ ls -la
total 0
drwxr-xr-x+ 12 elwoodblues  staff   408 Sep 23 20:09 .
drwxr-xr-x   5 root         admin   170 Sep 23 11:25 ..
-rw-----   1 elwoodblues  staff    3 Sep 23 11:25 .CFUserTextEncoding
drwx-----   2 elwoodblues  staff   68 Sep 23 20:09 .Trash
drwx-----+   3 elwoodblues  staff  102 Sep 23 20:09 Desktop
drwx-----+   4 elwoodblues  staff  136 Sep 23 11:25 Documents
drwx-----+   4 elwoodblues  staff  136 Sep 23 11:25 Downloads
drwx-----@  30 elwoodblues  staff 1020 Sep 23 20:39 Library
drwx-----+   3 elwoodblues  staff  102 Sep 23 11:25 Movies
drwx-----+   3 elwoodblues  staff  102 Sep 23 11:25 Music
drwx-----+   4 elwoodblues  staff  136 Sep 23 11:25 Pictures
drwxr-xr-x+   5 elwoodblues  staff   170 Sep 23 11:25 Public
```

Similar to Windows systems, the user has folders for specific types of data (i.e., My Documents, My Music, My Pictures, etc.) The `Movies`, `Music`, and `Pictures` directories should, (but are not required to) contain items related to their respective directories.

The `.Trash` contains the items the user has chosen to “delete”, similar to the Windows Recycle Bin.

The `Downloads` directory is the default folder for downloads from web browsers and other applications. This directory may contain years' worth of user downloads!

The `Public` directory is used for items the user has chosen to share with other users.

The `Library` [User Library] contains many items related to the specific user, such as preferences and application data. This `Library` directory is different from the `System Library` and `Local Library` directories. These files are specific to a user account.

User's Library Directory: ~/Library/

Containers: An application sandbox that is used to protect your data against malicious software

- [10.7+] ~/Library/Containers/ – Per App Data
- [10.8.3] ~/Library/Group Containers/ – Shared App Data

Preferences: Configuration Files

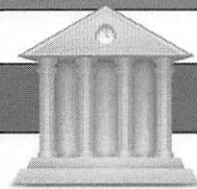
- “Reverse DNS” format (Ex: com.apple.iCal.plist) – <TLD>.<Company>.<Application>.plist
- ~/Library/Preferences/, ~/Library/[Group] Containers/<bundleid>/Data/Library/Preferences/

Application Support: App-Specific Data

- ~/Library/Application Support/

Caches: Per App Cached Data

- ~/Library/Caches



The user's `Library` directory contains many subdirectories of interest. The few listed here will be mentioned over and over. A forensic analyst may find lots of good forensic tidbits in this directory that they can tie to a specific user account. Each user account will have its own User Library directory.

It is worth noting here that the tilde (~) is used as a terminal shortcut to the current user's home directory.

The `Containers` directory, introduced in 10.7 (Group Containers in 10.8.3), contains data that is sandboxed. This data will be similar to that found in the `~/Library/Application Support/` directory for those applications that do not implement sandboxing. It is worth looking in both directories to find the specific app data you are looking for.

The `User Library` contains the user's preferences, usually in the form of a property list file (.plist). The format for this is sometimes called “reverse DNS” format. Reverse DNS format starts with the top-level domain of the company (such as “.com” or “.org”), the company name (i.e., “apple” or “microsoft”), and finally, a “.plist” file extension at the end.

The `Application Support` directory contains data specific to various applications. This is similar to the `AppData` directory on Windows systems. Each application directory may contain databases, property list files, or other proprietary data files. The method in which each application stores its data is up to the developer.

The `Caches` directory stores cached data in subdirectories that are specific to an application. It should be noted that the filenames may be in reverse DNS format (i.e., `com.app.Terminal`), or as the application or company name (Adobe). It is normal to see a company name with various application directories nested underneath. For example, the `Google` directory may contain a `Chrome` folder, while the `Microsoft` directory may contain an `Office` folder.

Containers and Application Sandboxing

User application data may be saved in one of two directories depending on whether or not an application is sandboxed:

- Legacy Location: `~/Library/Application Support/`
- Sandbox Location: `~/Library/[Group] Containers/<Bundle ID>/Data/Library/Application Support/<App Name>/`

```
com.amazon.Kindle
com.apple.FaceTime
com.apple.ImageKit.RecentPictureService
com.apple.Notes
com.apple.PassXCPService
com.apple.Preview
com.apple.ShareKitHelper
com.apple.TextEdit
nibble:Containers sledwards$ ls -l com.apple.TextEdit/
total 120
-rwxr--r--  1 sledwards  staff  58688 Sep 23 20:39 Container.plist
drwx----- 10 sledwards  staff   340 Jul 10 15:07 Data
nibble:Containers sledwards$ ls -l com.apple.TextEdit/Data/
total 40
lrwxr-xr-x  1 sledwards  staff   19 Jul 10 15:07 Desktop -> ../../../../Desktop
drwx-----  3 sledwards  staff  102 Jul 10 15:07 Documents
lrwxr-xr-x  1 sledwards  staff   21 Jul 10 15:07 Downloads -> ../../../../Downloads
drwx----- 28 sledwards  staff  952 Aug 13 20:45 Library
lrwxr-xr-x  1 sledwards  staff   18 Jul 10 15:07 Movies -> ../../../../Movies
lrwxr-xr-x  1 sledwards  staff   17 Jul 10 15:07 Music -> ../../../../Music
lrwxr-xr-x  1 sledwards  staff   20 Jul 10 15:07 Pictures -> ../../../../Pictures
nibble:Containers sledwards$ ls com.apple.TextEdit/Data/Library/
Application Scripts  ColorSync  Images  Preferences
Application Support  Colors     Input Methods  QuickLook
Audio                Components KeyBindings  Saved Application State
Autosave Information  Compositions  Keyboard Layouts  Sounds
```

Each “Container” in the Containers directory is named in the reverse DNS format. Each directory contains a Container.plist file and a Data directory.

The Container.plist file contains information about the sandbox application.

The Data directory contains a similar layout to the user folder with symbolic links to various directories. The important directories here are those that are **not** links. In the screenshot above, the Documents and Library directories contain the data of interest. While the nested directories may also contain linked data, the sandboxed data will not be linked.

References:

Apple Developer Documentation – App Sandbox Design Guide

https://developer.apple.com/library/archive/documentation/Security/Conceptual/AppSandboxDesignGuide/AboutAppSandbox/AboutAppSandbox.html#//apple_ref/doc/uid/TP40011183-CH1-SW1

<https://developer.apple.com/library/archive/documentation/Security/Conceptual/AppSandboxDesignGuide/MigratingALegacyApp/MigratingAnAppToASandbox.html>

```

com.amazon.Kindle
com.apple.FaceTime
com.apple.ImageKit.RecentPictureService
com.apple.Notes
com.apple.PassXCPCService
com.apple.Preview
com.apple.ShareKitHelper
com.apple.TextEdit
nibble:Containers sledwards$ ls -l com.apple.TextEdit/
total 120
-rwxr--r--  1 sledwards  staff  58688 Sep 23 20:39 Container.plist
drwx----- 10 sledwards  staff   340 Jul 10 15:07 Data
nibble:Containers sledwards$ ls -l com.apple.TextEdit/Data/
total 40
lrwxr-xr-x  1 sledwards  staff   19 Jul 10 15:07 Desktop -> ../../../../Desktop
drwx-----  3 sledwards  staff  102 Jul 10 15:07 Documents
lrwxr-xr-x  1 sledwards  staff   21 Jul 10 15:07 Downloads -> ../../../../Downloads
drwx----- 28 sledwards  staff  952 Aug 13 20:45 Library
lrwxr-xr-x  1 sledwards  staff   18 Jul 10 15:07 Movies -> ../../../../Movies
lrwxr-xr-x  1 sledwards  staff   17 Jul 10 15:07 Music -> ../../../../Music
lrwxr-xr-x  1 sledwards  staff   20 Jul 10 15:07 Pictures -> ../../../../Pictures
nibble:Containers sledwards$ ls com.apple.TextEdit/Data/Library/
Application Scripts  ColorSync          Images              Preferences
Application Support  Colors             Input Methods      QuickLook
Audio                Components         KeyBindings        Saved Application State
Autosave Information Compositions       Keyboard Layouts   Sounds

```

```

nibble:Application Support sledwards$ pwd
/Users/sledwards/Library/Application Support
nibble:Application Support sledwards$ ls
AddressBook          Dock                MobileSync          Ubiquity
Aperture            Firefox            Mozilla             VMware Fusion
Apple               Google             NotificationCenter  com.apple.QuickLook
BlackBagTech        Librarian          Preview             com.apple.TCC
CocoaPacketAnalyzer LittleSnapper       SyncServices        iCloud
Console             Mail               TextWrangler        iLifeAssetManagement
CrashReporter       Microsoft          The Omni Group      iLifePageLayout

```

```

nibble:Caches sledwards$ pwd
/Users/sledwards/Library/Caches
nibble:Caches sledwards$ ls
$(CFBundleIdentifier)
Adobe
Cleanup At Startup
DocSetAccess
Firefox
Google
Java
Metadata
Microsoft
QQCompositionRepository-com.apple.iTunes.cache
QuickTime
SubmitDiagInfo
TemporaryItems
com.amazon.Amazon-Software-Downloader
com.apple.NetworkBrowserAgent
com.apple.QuickLookDaemon
com.apple.QuickLookDaemon32
com.apple.Safari
com.apple.ScreenSaver.Engine
com.apple.Server.v2
com.apple.SoftwareUpdate
com.apple.SystemProfiler
com.apple.Terminal
com.apple.WebProcess
com.apple.appstore
com.apple.coreservices.uiagent
com.apple.dashboard.client
com.apple.dt.Xcode

```

Standard Unix Directories: *nix Forensics?

<code>/bin</code>	• Contains binaries such as <code>cat</code> , <code>echo</code> , and <code>mv</code>
<code>/sbin</code>	• Contains “system” binaries such as <code>fsck</code> , <code>mount</code> , and <code>ping</code>
<code>/dev</code>	• Contains “device” files such as <code>disk0s2</code> , <code>stdout</code> , and <code>zero</code>
<code>/opt</code>	• Contains “optional” software. Default install location for package management tools such as <code>homebrew</code> .
<code>/private/var</code>	• Contains “variable” directories. The contents changes often. Notable directories include <code>/log</code> , <code>/db</code> , and <code>/audit</code> .
<code>/private/etc</code>	• Contains system configuration data such as <code>passwd</code> , <code>hosts</code> , and <code>resolv.conf</code>
<code>/private/tmp</code>	• Contains temporary files

While macOS/iOS are based on *nix, the forensic artifacts are not necessarily the same or in the same locations. Some *nix experience can be helpful; however, Apple has put their own special spin on many artifacts that are specific to them.

Mac OS X systems contain standard Unix directories.

- `/bin` contains various command utilities
- `/sbin` contains system binaries
- `/dev` contains “device” files
- `/opt` (`/usr/local/opt`) contains “optional” software
- `/private/var` contains notable “variable” directories, such as `/log`, `/db`, and `/audit`
- `/private/etc` contains system configuration data, with notable files such as `hosts`, `passwd`, and `resolv.conf`
- `/private/tmp` contains temporary files

It is worth noting that the files `var`, `etc`, and `tmp` are all symbolic links to their `/private` equivalent.

Mac Acquisition: Caveats and Considerations

Desktop and Laptop Hard Drives

Flash SSD Drives w/ Proprietary Interfaces or Soldered to Board

Live vs. Dead Acquisition

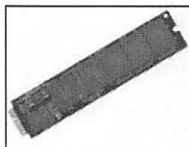
FileVault Encryption

System Integrity Protection (SIP)

EFI Passwords

Target Disk Mode

Disk Arbitration



Boot Camp

Fusion Drives

T2 Security Chip



SANS | DFIR

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 31

Many of the same tools used for other systems can be used to acquire a Mac. If necessary and physically possible, the hard drives can be removed for acquisition. The author recommends using the website <http://www.ifixit.com/> for hard drive removal instructions. Some Macs are known to be harder to extract the hard drive from than others.

The MacBook Air has introduced Solid State/flash hard drives to a wide variety of users. Most models do not use the “regular” 1.8” or 2.5” Solid State Drive (SSD); but rather they use a bank of flash chips on a board with a non-standard interface. As of this writing, Other World Computing makes an external enclosure for the MacBook Air SSD drives that will allow the use of standard interfaces. Newer systems have a proprietary interface for which there is no adapter yet. This nullifies the possibility of hard drive upgrades or use in external hard drive enclosures.

Encryption can make dead-drive acquisition difficult with Legacy FileVault and FileVault 2 encryption. Legacy FileVault encrypts the user's home directory, while FileVault2 implements full disk encryption. This does not make acquisition impossible. You may still image the drive, but you better hope you can get the password! If a system is using FileVault and is up and running, it is best to image unencrypted while you can.

Target Disk Mode (TDM) is used to acquire a Mac hard drive (primary drive only) while still in its original machine. This allows the computer to be seen by an analysis system as an external FireWire/Thunderbolt drive. Some systems, including older MacBook Airs, do not have FireWire or Thunderbolt ports; therefore, TDM is not possible. When a hard drive has a Boot Camp partition, you will see two volumes—one OS X, the other for Windows. If a hard drive has been encrypted using FileVault2, this volume will need to be unlocked. This is described in detail in Section 4. Booting the target system with the “T” key held down can access this mode.

It is worth noting that older systems may have an Open Firmware password set. There are a few methods that may be used (with varying levels of success) to bypass this:

- Change the amount of RAM in the system and reboot.
- Press Command+Option+P+R to clear the PRAM.

You may enter the Open Firmware terminal by using Command+Option+O+F key sequence.

On newer systems, if the EFI Firmware password is set, you will need to obtain a specific key hash and send it to Apple (subpoenas@apple.com). Instructions can be found here: <http://www.cnet.com/news/efi-firmware-protection-locks-down-newer-macs/>.

Disk arbitration is the process that automatically mounts disks. When acquiring disks using a Mac, you must determine if Disk Arbitration is enabled or not. If it is enabled, you may automatically mount an evidence disk (which makes writes to it). This can be turned on/off using the `launchctl` commands, but beware that these commands do not persist after a reboot.

Enable (Enabled by default):

```
sudo launchctl load  
/System/Library/LaunchDaemons/com.apple.diskarbitrationd.plist
```

Disable:

```
sudo launchctl unload  
/System/Library/LaunchDaemons/com.apple.diskarbitrationd.plist
```

To determine if disk arbitration is running, run the following command to search for the `diskarbitrationd` process (should be under the root user context):

```
ps auxw | grep diskarbitrationd
```

Boot Camp is used to create two bootable partitions, one for OS X and one for Windows. While in Target Disk Mode, if the Mac has a Boot Camp partition (a Windows installation on a separate partition), it should be noted that this may be automatically mounted on a Windows acquisition system. It is also possible to triple-boot (Linux!) Mac systems using third-party tools. The most notable is rEFIt (<http://www.rodsbooks.com/refind/>).

SSD picture from iFixit.com

<https://www.ifixit.com/Teardown/MacBook+Air+11-Inch+Late+2010+Teardown/3745>

The T2 Security Chip is the latest and greatest in the security and acquisition problems. More information can be found here: https://www.apple.com/mac/docs/Apple_T2_Security_Chip_Overview.pdf

Acquisition Tools

Hard Drive

Access Data
FTK Imager

Guidance Software
EnCase

BlackBag Technologies
MacQuisition

Magnet Acquire

dd / dcfldd / dc3dd

Live USBs: CAINE,
Sumuri Paladin/Recon

Network

dd and netcat

F-Response

FTK Enterprise

EnCase Enterprise

Memory

Johannes Stuetzgen/Google
osxpmem

Volexity Surge

BlackBag Technologies
MacQuisition

Sumuri
Recon

Many of the same acquisition tools can be used on hard drives. Hard drives are hard drives—those same tools you use for Windows-based imaging can be used for Mac systems.

Live acquisition should be used if you need to acquire volatile data or a drive that has implemented encryption. Many of the same tools used for dead-drive acquisition have live response functions as well.

It may be necessary to acquire a system over the network, whether for enterprise or simple distance issues. There are tools that range from a simple over-the-network implementation of `dd` and `netcat` to full-blown enterprise suites.

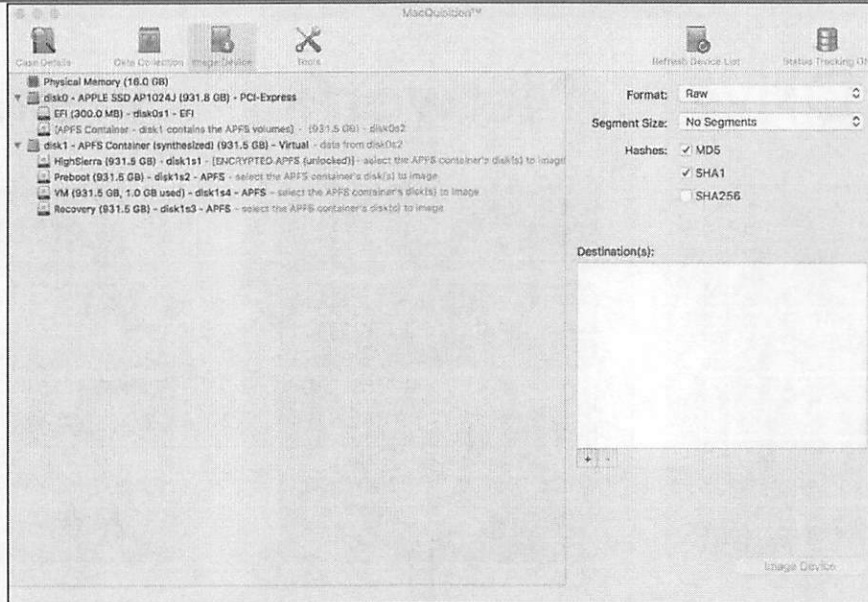
Memory acquisition may be needed for investigations that are highly dependent on volatile data, such as network investigations.

We will go more in-depth into Mac memory acquisition and analysis. The tools listed above are some of the recommended tools to acquire Mac memory.

Each memory acquisition tool has their own limitations and benefits, and it is up to the user to determine which tool is best for their current situation. For example, MacQuisition and Recon are part of an application that acquires volatile data, hard drives, and memory. Note: Each tool requires administrative system privileges to run.

The tools listed are not the only tools available. There are many options at various price points on the market, and it is up to you and/or your organization to find the right product for your needs.

MacQuisition and APFS



Newer versions of MacQuisition can access and image APFS disks without the need to enable/disable SIP. You may choose to image /disk0 for the whole disk (recommended) or just the logical APFS volume containing the user data /dev/disk1. If the data is spread across two physical drives (not the case here), it is recommended to image the logical APFS volume.

SIP Acquisition Errors

- Devices not available with SIP enabled
 - Even with root!
- Disable SIP using `csrutil` in Recovery Mode
 - Reboot, CMD+R
 - `csrutil` disable

```
MacBook-Pro:~ oompa$ diskutil list
/dev/disk0 (internal):
#:  
0:      GUID_partition_scheme          1.0 TB   disk0  
1:      EFI EFI                        314.6 MB disk0s1  
2:      Apple_APFS Container disk1     1.0 TB   disk0s2  
  
/dev/disk1 (synthesized):  
#:  
0:      APFS Container Scheme -        +1.0 TB   disk1  
        Physical Store disk0s2  
1:      APFS Volume HighSierra         680.2 GB disk1s1  
2:      APFS Volume Preboot            27.4 MB  disk1s2  
3:      APFS Volume Recovery           517.8 MB disk1s3  
4:      APFS Volume VM                 1.1 GB   disk1s4  
  
MacBook-Pro:~ oompa$ dd if=/dev/disk0 of=/tmp/test.dd  
dd: /dev/disk0: Operation not permitted  
MacBook-Pro:~ oompa$ sudo !!  
sudo dd if=/dev/disk0 of=/tmp/test.dd  
Password:  
dd: /dev/disk0: Operation not permitted  
MacBook-Pro:~ oompa$ sudo dd if=/dev/disk1 of=/tmp/test.dd  
dd: /dev/disk1: Operation not permitted  
MacBook-Pro:~ oompa$ sudo dd if=/dev/disk1s1 of=/tmp/test.dd  
dd: /dev/disk1s1: Operation not permitted
```

```
Terminal — -bash — 80x24  
-bash-3.2# csrutil disable  
Successfully disabled System Integrity Protection. Please restart the machine fo  
r the changes to take effect.  
-bash-3.2#
```

System Integrity Protection (SIP) may cause acquisition errors if it is enabled on the user's system. SIP can be enabled and disabled by rebooting into Recovery Mode and using `csrutil enable/disable`. This, however, becomes an issue when the user has encryption turned on and you do not have user credentials. A logical acquisition is recommended at this point.

Mounting APFS Images (w/ or w/o FileVault) [10.13+ Host]

```
1. $ sudo mkdir /Volumes/apfs_image/
2. $ sudo mkdir /Volumes/apfs_mounted/
3. $ sudo xmount --in ewf apfs.E01 --out dmg /Volumes/apfs_image/
4. $ hdiutil attach -nomount /Volumes/apfs_image/apfs.dmg
5. $ diskutil ap list
6. $ diskutil ap unlockVolume <Disk GUID> -nomount
7. $ sudo mount_apfs -o rdonly,noexec,noowners /dev/disk#
   /Volumes/apfs_mounted/
```

1. Create a mount point for the transformed image (E01 images can be transparently converted to DMG files using `xmount`).
2. Create a mount point for the mounted image. This will act as the root of the mounted volume for your analysis.
3. Use `xmount` to create a DMG file from an E01 image.
4. Use `hdiutil` to attach (but not mount) the image.
5. Using `'diskutil ap list'`, select the OS/User volume.
6. *If this APFS volume is encrypted with FileVault, the disk will need to be unlocked using `'diskutil ap unlockVolume'`, using the GUID of the disk from the previous command.*
7. Using the disk number and slice from the previous command, mount it using `mount_apfs` with the read-only, no execution, and no ownership options, onto the mount point that acts as the root of its file system.

Mounting Images: Disk Eject and Unmount

```
1. $ diskutil list
2. $ diskutil eject /dev/disk*
3. $ mount
4. $ sudo umount /Volumes/disk_image/
```

10.12+ /xmount Users: Will need to use "sudo" for commands.

The disk images should stay mounted and available until you reboot or until you eject/unmount them.

When you are finished with the mounted images, you will need to eject and unmount them.

1. Use the `diskutil list` command to view the list of mounted disks. Find the disk that you want to eject.
 2. Use the `diskutil eject` command on the disk you would like to eject. (This may also be done by pressing the eject button in the Finder application).
 3. Use the `mount` command to view the list of mounted disks. Find the disk that you want to unmount (likely `/Volumes/disk_image/`, if you are following the naming scheme from the examples).
 4. Use the `umount` command with the mount point to unmount the disk. A troublesome disk may have to be unmounted using the `-f` option to forcibly unmount the disk.
- Note: Since we entered "sudo" to create the virtual image, "sudo" will likely need to be employed using the `xmount` method.

Mounting HFS+ Images: Method I—xmount

```
1. $ mkdir /Volumes/dademurphy_image/
2. $ mkdir /Volumes/dademurphy_mounted/
3. $ sudo xmount --in ewf ~/FOR518/dademurphy.E01 --out dmg
   /Volumes/dademurphy_image/
4. $ hdiutil attach -nomount
   /Volumes/dademurphy_image/dademurphy.dmg
5. $ mount_hfs -j -o rdonly,noexec,noowners /dev/disk#
   /Volumes/dademurphy_mounted/
```

10.12+ Users: Will need to use “sudo” for all commands.

The first method to mount an image file is to use the `xmount` command from www.penguin.lu/index.php.

1. Use the `mkdir` command to create a mount point for the `xmount` output. In this class, the directory name `dademurphy_image` is used because it will only host the image file.
2. Use the `mkdir` command to create a mount point for the mounted drive. The directory `dademurphy_mounted` is used in this class to represent the mounted disk image.
3. Uses `xmount` to mount the `dademurphy.E01` image where you have your image located (the example shows `~/FOR518`), as a DMG file. This command requires you to use the `sudo` command, thus it will ask you for your administrator password when executed.
 - `--in`: Tells `xmount` what input file type to expect; our images are in a compressed EWF format.
 - `--out`: Tells `xmount` what output format you want. We want a DMG file so we can mount it in Finder.
 - `Input File`: Where the image file is located.
 - `Mount Point`: Newly created specifically for this image.
4. Uses the `hdiutil` command with the “attach” verb to mount the newly created DMG volume so it is available in Finder and Terminal application. Use the `-nomount` argument to suppress mounting (for now). The output from this command will display a `/dev/disk#`. Use the appropriate disk device in the next command.
5. Use the `mount_hfs` command with the following parameters to mount the `/dev/disk#` (from the previous command) to the `/Volumes/dademurphy_mounted/` mount point. This drive will now be available in the Finder or Terminal applications.
 - j: Ignore the journal
 - o: Options:
 - `rdonly`: Mount in read-only mode.
 - `noexec`: Do not allow execution of binaries on mounted system.
 - `noowners`: Ignore ownership on the mounted volume.

You can access this newly created mounted drive on `/Volumes/dademurphy_mounted/`.

Mounting HFS+ Images: Method 2—ewfmount

```
1. $ mkdir /Volumes/dademurphy_image/
2. $ mkdir /Volumes/dademurphy_mounted/
3. $ ewfmount ~/FOR518/dademurphy.E01 /Volumes/dademurphy_image/
4. $ ln -s /Volumes/dademurphy_image/ewf1 ~/FOR518/dadeimage.dmg
5. $ hdiutil attach -nomount ~/FOR518/dadeimage.dmg
6. $ mount_hfs -j -o rdonly,noexec,noowners /dev/disk#
   /Volumes/dademurphy_mounted/
```

10.12 Users: Will need to use “sudo” for all commands

10.13 Users: Unknown Fuse Error—use xmount method

It can never hurt to have more than one way to mount an image. A second way uses the `ewfmount` command from the `libewf` package available at <https://github.com/libyal/libewf>.

1. Use the `mkdir` command to create a mount point for the `ewfmount` output. The directory name `dademurphy_image` is used in the example.
2. Use the `mkdir` command again to create a mount point for the mounted disk image. `dademurphy_mounted` is used in the example above.
3. Use `ewfmount` to mount the `dademurphy.E01` image to the `/Volumes/dademurphy_image/` mount point.
4. Use the `ln -s` command to create a symbolic link for the `ewf1` file, and name the link `dadeimage.dmg`. (A DMG file is needed for `hdiutil` to recognize the file).
5. Uses the `hdiutil` command with the “attach” verb to mount the newly created DMG volume so it is available in Finder and Terminal application. Use the `-nomount` argument to suppress mounting (for now). The output from this command will display a `/dev/disk#`. Use the appropriate disk device in the next command.
6. Use the `mount_hfs` command with the following parameters to mount the `/dev/disk#` (from the previous command) to the `/Volumes/dademurphy_mounted/` mount point. This drive will now be available in the Finder or Terminal applications.

-j: Ignore the journal

-o: Options:

rdonly: Mount in read-only mode.

noexec: Do not allow execution of binaries on mounted system.

noowners: Ignore ownership on the mounted volume.

You can access this newly created mounted drive on `/Volumes/dademurphy_mounted/`.

Mounting HFS+ 4k Images (w/ or w/o FileVault) [10.13 Host]

Images on systems that use 4096-byte sectors may cause mounting issues. Only the 'hdiutil' binary on 10.13 has the `-blocksize` option.

```
1. $ sudo mkdir /Volumes/4k_image/
2. $ sudo mkdir /Volumes/4k_mounted/
3. $ sudo xmount --in ewf 4k.E01 --out dmg 4k.E01 /Volumes/4k_image/
4. $ hdiutil attach -nomount -blocksize 4096
   /Volumes/4k_image/4k.dmg
5. [Input Password in Prompt Window]*
6. $ diskutil cs list
7. $ sudo mount_hfs -o ronly,noexec,noowners /dev/disk#
   /Volumes/4k_mounted/
```

40

1. Create a mount point for the translated image (using `xmount` later in step 3).
2. Create a mount point for the mounted image. This will act as the root of the mounted volume for your analysis.
3. Use `xmount` to create a `dmg` from an `ewf` formatted image.
4. Use `hdiutil` to attach (but not mount) the image. Using the `-blocksize` (not in the man page) we can specify the 4k block size to be used.
5. Put the password in the password prompt. If you would like to do this all via command line, additional instructions are below.*
6. Use `diskutil cs list` to determine which HFS+ volume to mount (under Logical Volume GUID). It will not be listed in the output from `hdiutil` like before, since CoreStorage is being used. (Be sure not to use your host system's!)
7. Using that disk number and slice, mount it using `mount_hfs` with the read-only, no execution, and no ownership options onto the mount point that acts as the root of its file system.

*Instead of putting the password into the prompt window, you can pass `-stdinpass` to the `hdiutil attach` command. However, you will still need to unlock the volume by getting the Logical Volume GUID from the `diskutil cs list` command for the attached volume. With that GUID, you can use `diskutil cs unlockVolume <GUID>` to unlock the volume. Proceed to the `mount_hfs` command.

```
$ diskutil cs unlockVolume <LogicalVolumeGUID>
```

Lab 1.1

BlackLight Case Setup and Image Mounting

This page intentionally left blank.

Section I: Agenda

Part 1: Apple Essentials

Part 2: Mac Essentials and Acquisition

Part 3: iOS Essentials and Acquisition

Part 4: Disks and Partitions

This page intentionally left blank.

Section I: Part 3

iOS Essentials and Acquisition

This page intentionally left blank.

macOS vs. iOS

macOS

Intel x86-64

HFS+/APFS

Mouse/Keyboard/Trackpad

Sandboxing (Containers)

Finder.app

Many User Accounts

Unencrypted by Default (except T2)

iOS

ARM 32/64

HFS+/APFS

Multi-touch (Fingers)

Sandboxing (Jails, Containers)

Springboard.app

Two Accounts (root and mobile)

Data Partition Uses Per-File Encryption

In newer versions, macOS and iOS seem to be merging. Many of the files are similar, in that they both use sandboxing and implement HFS+, and the file directory structures are alike.

They are still different, in that they run on different architectures and use different ways to present the data to the user, and still allow the user to interact with the systems.

While Macs can have many user accounts, only two user accounts exist on iOS devices; the standard user “mobile”, and the privileged user “root”. The default password for the “root” and “mobile” accounts is “alpine”, if accessed on a jailbroken device.

macOS does not encrypt by default (except for systems with T2 chips), while the Data partition on iOS is encrypted by default user per-file encryption.

iOS Disks and Partitions

System

`/dev/disk0s1s1 (LwVM)`
(HFSX+/APFS)

Contains System Domain Artifacts

Mounted on `/`

Anywhere from ~1–6 GB in Size
Depends on OS Version

Not Encrypted (HFS)

Data

`/dev/disk0s1s2 (LwVM)`
(HFSX+/APFS)

Contains User Domain Artifacts

Mounted on `/private/var`

Up to 1 TB in Size
Depends on iDevice Storage

Per-File Encrypted

Each iDevice contains two partitions. The System partition contains the system-related files and binaries. It is mounted on “/”, or the root directory. Depending on the iOS version, it is usually about 1–6 gigabytes in size. This partition is not encrypted.

The Data partition is where the user data is stored: all the applications, phone records, photos, etc. It is mounted on `/private/var` and can be up to 1 Terabyte in size, depending on the size of the device. Encryption is used on this partition.

There are no external storage areas on the device other than the SIM card.

Partitioning on HFS based iOS devices uses Lightweight Volume Manager, thus the second disk “slice” for each partition. More information about LwVM can be found here:

<https://www.theiphonewiki.com/wiki/LwVM>. LwVM has been replaced by APFS volumes for devices on 10.3 and newer.

Reference:

<http://theiphonewiki.com/wiki//private/etc/fstab>

iOS Security Concepts

Apple Platform Security Guide

Embedded AES 256 Crypto Keys

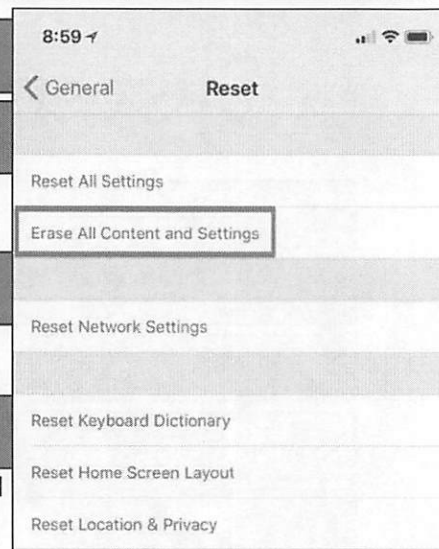
- UID: Burned in hardware, unique to **each** device
- GUID: Compiled into hardware, unique to each **type** of device

Secure Enclave

- Introduced in A7 chips (iPhone5S), provides crypto for key management for Data Protection and Touch/Face ID operations

Effaceable Storage

- Area on flash storage where data file encryption keys are stored
- Wiped when “Erase All Content and Settings” is selected or remote wipe is initiated



Cryptographic keys are burned or compiled into the physical hardware of the device. The Secure Enclave coprocessor provides the cryptographic processing using these keys for file system encryption and Data Protection, as well as usage of Touch ID. The Secure Enclave was introduced in the A7 chips in the iPhone 5S generation of devices.

These keys are used to cryptographically protect data on the device by using them as keys to create other keys to encrypt specific data files. These additional keys are stored in the Effaceable Storage area.

This storage area can be wiped by the user when they select the “Erase All Content and Settings” option available in the Settings | General | Reset menu. This activity resets the device to a point where a forensic analyst will not be able to recover any user data because it is encrypted by the keys that have just been wiped. The same functionality is performed when a user performs a remote wipe using MDM software or iCloud.

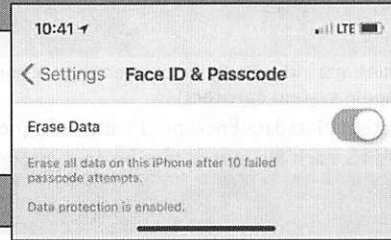
Reference:

Apple Platform Security Guide: <https://support.apple.com/en-ca/guide/security/welcome/web>

Passcodes, Passwords, Touch ID, and Face ID

Passcodes and Passwords

- **Simple:** Four Digits/Six Digits
- **Custom Numeric**
- **Complex:** Alphanumeric, Arbitrary Length
- Brute force must be done on device, uses hardware UID key
- User may choose to auto-wipe device after 10 failed attempts



Touch ID

- Fingerprint, passcode still required:
- On first unlock after boot, after 48 hours with no unlock, remotely locked, five unsuccessful Touch ID unlock attempts, enrolling new Touch IDs.
 - iOS 9 Added: When device has not been unlocked in the last 6 days w/ passcode and Touch ID was not used in the last 8 hours.
 - iOS 11 Added: Disabling of Touch ID by clicking side button five times.
- Stored in Secure Enclave, not forensically accessible.

Face ID

- Facial recognition, passcode still required:
- Required after first unlock, after 48 hours with no unlock, not unlocked with passcode in 6.5 days or in last 4 hours, remotely locked, five unsuccessful facial matches, disabling Face ID with iPhone side and volume buttons for two seconds.
- Stored in Secure Enclave; not forensically accessible.

There are many different methods to protect an iDevice.

There are a few types of passcodes that can be used. A simple passcode (four or six digit), custom numeric, or complex passcode that can be made up of alphanumeric characters of any length.

On newer devices, a user may implement Touch ID. Touch ID is used along with a passcode to allow a user to unlock their phone with their fingerprint. The passcode is still required in certain circumstances, such as after the phone starts up, or after a reboot.

On modern devices, Face ID is available which works nearly like Touch ID, but uses a person's face instead. At this time, only one face is allowed to unlock the device.

Reference:

Apple Platform Security Guide: <https://support.apple.com/en-ca/guide/security/welcome/web>

iOS File Data Encryption [1]

File System Encryption

- Most files are individually encrypted with a unique key that is stored in an extended attribute (`com.apple.system.cprotect`)
- File System Metadata Encrypted with APFS (not with HFS+)
- With APFS, each file extent may be encrypted with a different key

Data Protection

- Some files have additional protection:
 - iOS 4–7: Mail, Third-party Apps (if implemented)
 - iOS 8: Mail, Calendar, Contacts, Call History, Reminders, Notes, Messages, Photos, Health, and Third-party Apps (if implemented)
- Enabled and protected with passcode/Touch ID

Data Protection Classes

- Class A: Complete Protection (`NSFileProtectionComplete`)
- Class B: Protected Unless Open (`NSFileProtectionCompleteUnlessOpen`)
- Class C: Protected Until First User Authentication (`NSFileProtectionCompleteUntilFirstAuthentication`)
- Class D: No Protection (`NSFileProtectionNone`)

A large portion of the files located on the Data partition of an iOS device implement the “`NSFileProtectionNone`” class key, meaning they are only encrypted with the burned-into-hardware UID key. This class key is stored in the Effaceable Storage area, while the file key is stored in the `com.apple.system.cprotect` extended attribute for the file.

Data Protection is used to give extra protection for various file classes. Starting in iOS 4, the Mail application received this additional security. This additional security is enabled by use of a passcode and/or Touch ID. Some of the class keys that implement this protection are `NSFileProtectionComplete`, `NSFileProtectionCompleteUnlessOpen`, and `NSFileProtectionCompleteUntilFirstUserAuthentication`. These class keys require the user's passcode and/or Touch ID for access. This is why some data may not be accessible in a forensic acquisition without the user's credentials.

Encryption of file data

iOS 4-7: Mail, Third-party Apps (if implemented)

iOS 8: Mail, Calendar, Contacts, Call History, Reminders, Notes, Messages, Photos, Health, and Third-party Apps (if implemented)

References:

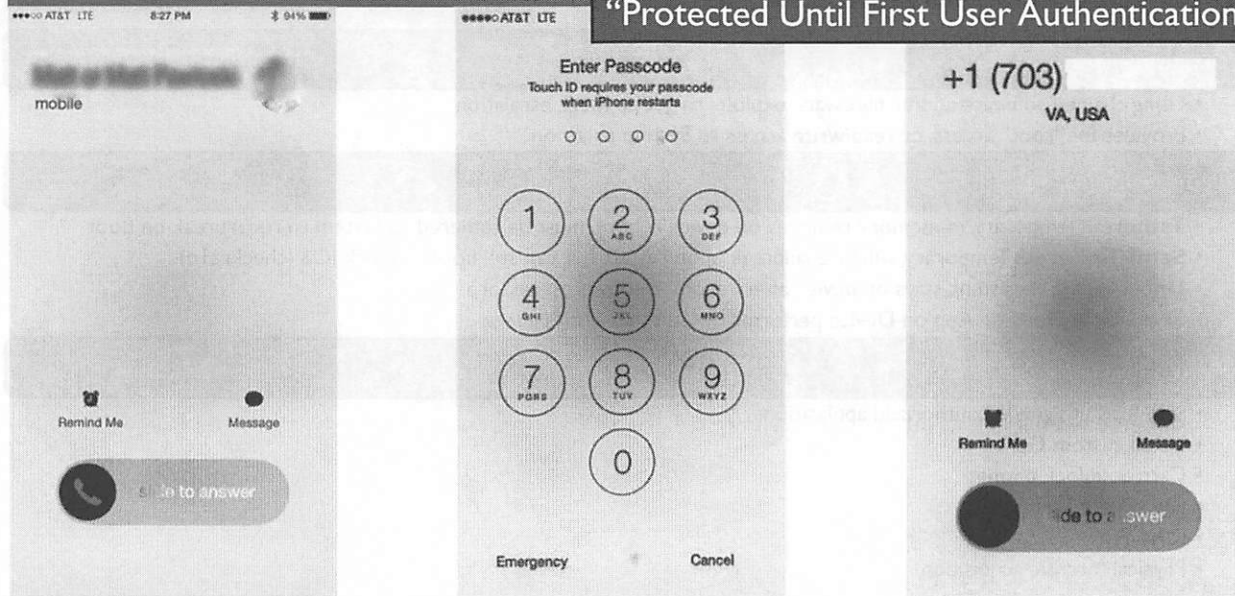
Apple Platform Security Guide: <https://support.apple.com/en-ca/guide/security/welcome/web>

iPhone Data Protection In Depth:

<http://esec-lab.sogeti.com/static/publications/11-hitbamsterdam-iphonedataprotection.pdf>

iOS File Data Encryption [2]

“Protected Until First User Authentication”



SANS | DFIR

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 49

Data Protection in Action!

The left screenshot is a contact in the Address Book that is calling on an already PIN-unlocked device.

The middle screenshot shows the PIN unlock screen that is provided for a user using Touch ID. They must input their PIN after a boot up or restart to unlock certain information, such as a phone number to contact relationship.

The third screenshot shows what rebooting and providing no PIN shows for the same caller.

iOS Jailbreaking: What Is Jailbreaking?

Breaking the iOS "Jail"

- Using chained software and/or hardware exploits to get privilege escalation
- Provides for "root" access, or read/write access to System partition

Types of Jailbreaks

- **Tethered:** Temporary, in-memory, removes on device reboot, must be tethered to system to re-jailbreak on boot
- **Semi-Tethered:** Temporary, will lose jailbreak upon restart but can still boot to stock iOS (checkra1n)
- **Untethered:** Persistent, stays on device after reboot, removes on restore
- **Semi-Untethered:** App on Device performs the jailbreak after reboot

Why Jailbreak?

- Side-load unsigned/unauthorized applications
- Install custom GUIs
- Carrier unlock phones
- File system access
- Research
- Physical forensic acquisition

The jailbreaking process allows a user to escalate their privileges on an iOS device using chained software and/or hardware exploits. Jailbreaking a device allows a user to gain root access to the System partition with read and write access, as opposed to just read-only access.

There are three types of jailbreaks: tethered, untethered, and semi-untethered. A tethered jailbreak is a temporary (in-memory) jailbreak that requires the device to be "tethered" to a system in order to keep the jailbreak through the reboot process. An untethered jailbreak is a persistent jailbreak that allows the user to reboot as needed without the requirement for another system to jailbreak. This jailbreak can be removed by doing a restore through iTunes. The semi-untethered jailbreak works through an app on the device, which allows a user to re-jailbreak their device when the app is run. A user can reboot the device to disable the jailbreak, however, the files are still on the device. This jailbreak can be removed using an iTunes restore.

Users jailbreak for many reasons:

- To install an application that is not an authorized app in the Apple App Store
- Install custom graphical user interfaces
- Unlock various carrier-based locks.
 - Users may want to unlock their phones to be able to use them on other networks
- Access to the file system.
 - Users may want to access some application data files or upload their own files
- Research: Hackers, developers, and/or researchers may use jailbreak for research purposes

As forensic analysts, we may also choose to jailbreak a device; some for the same reasons as users. We may need file system access for research purposes (i.e.: what data does this app store?), or to acquire a user's device in a physical format. Newer iOS devices require a forensic analyst to jailbreak the device to access the System partition or to acquire deleted/unallocated data from the User partition.

References:

<http://theiphonewiki.com/wiki/Jailbreak>

http://theiphonewiki.com/wiki/Tethered_jailbreak

http://theiphonewiki.com/wiki/Untethered_jailbreak

iOS Jailbreaking: Jailbreak Compatibility

Dependent on Device Hardware, iOS Version

- Current Compatibility Chart
 - <http://theiphonewiki.com/wiki/Jailbreak>

Popular Jailbreaking Software

- iOS 12.3/13 checkra1n – iPhone5S - iPhoneX
- iOS 12: unc0ver, RootlessJB, Chimera
- iOS 11: LiberiOS, Electra, unc0ver
- iOS 10: Yalu, Meridian, h3lix, doubleH3lix, g0blin, Saigon
- iOS 9: Pangu, Phoenix, Home Depot
- iOS 8: Pangu, TaiG, Home Depot

The jailbreak process is very dependent on device hardware and iOS version, down to the point releases. The specific device compatibility changes so frequently with updates from Apple and updates from the jailbreak hackers that it is best to determine if the device that you have in hand can be jailbroken. The charts on <http://theiphonewiki.com/wiki/Jailbreak> are updated frequently.

There are many different software packages available if you refer to the charts listed on theiphonewiki.com.

Reference:

<http://theiphonewiki.com/wiki/Jailbreak>

iOS Jailbreaking: Potential Evidence of...

- File System Table: `/private/etc/fstab`

```
fstab
/dev/disk0s1s1 / hfs ro 0 1
/dev/disk0s1s2 /private/var hfs rw, nosuid, noexec 0 2
```

```
fstab
/dev/disk0s1s1 / hfs rw 0 1
/dev/disk0s1s2 /private/var hfs rw 0 2
```

- App Stores
 - Cydia, cydiapackage, Bydia, Zydia, Installer, 25pp, Maiyadi, Cydia Lite
- GUI does not look like stock GUI
- Applications: iFile, SBSettings, SSH Apps, Tethering Apps, Configuration Apps
- Jailbreaking Apps: “Meridian”, “LiberiOS”, “mac_portal”, “Pangu App”, “unc0ver”, “rootlessJB”
- Jailbreaking Directories/Files: `/jb/`, `/meridian/`, `checkra1n.dmg`



The file system table located at `/private/etc/fstab` can be used to determine if a device has been jailbroken. This file is located on the System partition. This file shows how each partition is mounted.

- `/dev/disk0s1s1`: System Partition
- `/dev/disk0s1s2`: Data Partition (User)

The two screenshots above show two different devices. The left screenshot shows a non-jailbroken device, while the right screenshot shows a jailbroken device. Notice the “rw” and the “ro” mounting options. The right example shows the System partition was mounted as “rw”, or read/write (Jailbroken!), while the left shows the System partition was mounted as “ro”, or read-only (Default). It should be noted here that not all jailbreaking software will mount the System partition as read/write – it depends on the jailbreak.

The presence of unofficial app stores can also show that a device has been jailbroken. The main, unofficial app store used in the US is Cydia which downloads its applications into `/Applications` on the device. Be on the lookout for other, lesser known app stores or icons that play off of the Cydia app icon.

You may also tell if the device has been jailbroken by the way it looks. Look for non-stock icons, backgrounds, dock bar, notification center, etc.

Certain popular, unauthorized applications may be installed such as iFile and SBSettings, but always be on the lookout for SSH, tethering, or configuration applications.

Once an image is acquired, look at the file system on each partition. It should be obvious whether or not it was jailbroken by looking at various directories.

Files and Directories on Data partition

- /root/.ssh
- /root/dumpkeys6 (Elcomsoft), and other “forensic” utilities
- /stash

Files and Directories on System partition

- /private/etc/ssh
- /Library/LaunchDaemons/com.openssh.sshd.plist, *openssh* file and directories
- /private/etc/apt/sources.list.d/cydia.list
- /usr/libexec/cydia/
- /private/etc/apt/
- *untether* files and directories

iOS Acquisition Methods

Physical

Bit-for-bit copies of device partitions. Includes full file system and unallocated space

System Partition (HFS)

"Logical Physical" (Graykey, Cellebrite CAS, Tar Bundle)

Physical access, but only acquiring logical file system files

User Data Partition

File System/Backup (Logical)

A subset of file system files acquired through standard device communication protocols (Backup/AFC)

We will be using the following terms in this class to describe various acquisition types and the differences between them.

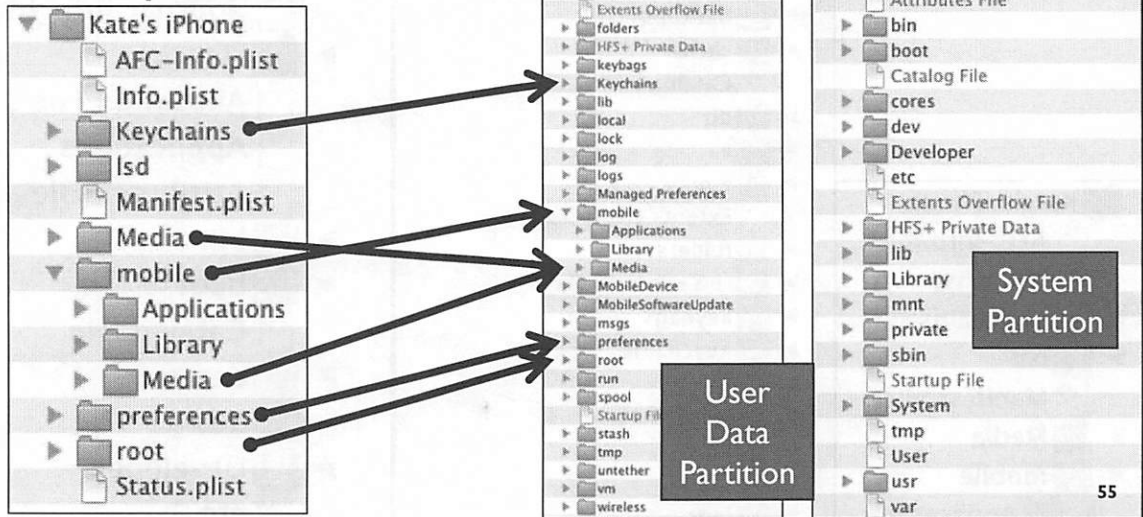
Physical: This is a full bit-for-bit copy of each partition or of the full disk. There are two partitions on iOS devices; the System and the User partitions. It is possible to get a full copy of each; however, the User partition uses per-file encryption, so it will likely limit your analysis capabilities. For the User partition, I would recommend using a "Logical Physical" described next.

"Logical Physical": This type of acquisition uses the physical access (via SSH or other means) to acquire the full file system of files, to include those that are not included in File System or Backup type acquisitions. This can be saved as raw files or in an archive such as a tarball.

File System/Backup (Logical): This acquisition may only acquire a subset of files available on the file system using specific protocols. The iTunes backup process (as well as most commercial tools) uses the Apple File Conduit Protocol (AFC) to acquire these files. The access provided by this protocol does limit which files can be acquired; however, some files have extra protections and cannot be gathered using this method.

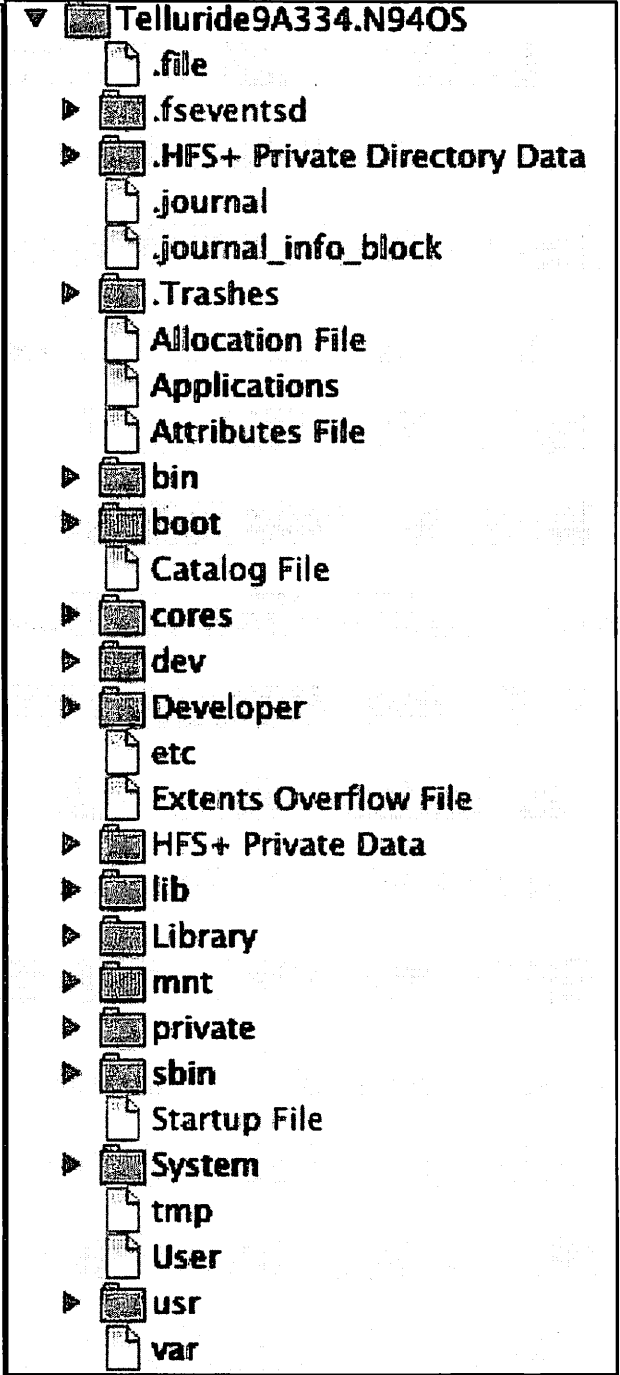
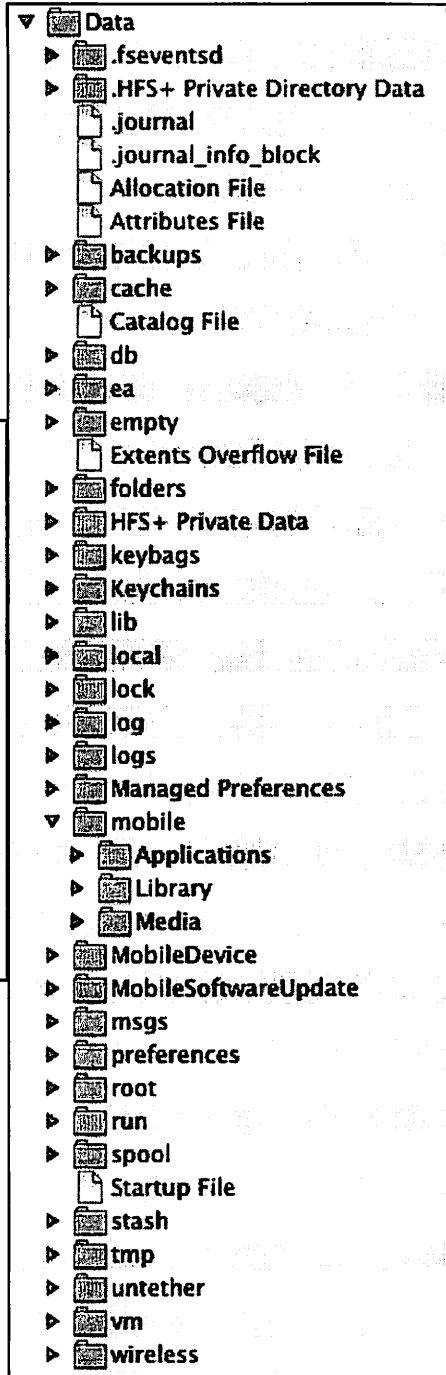
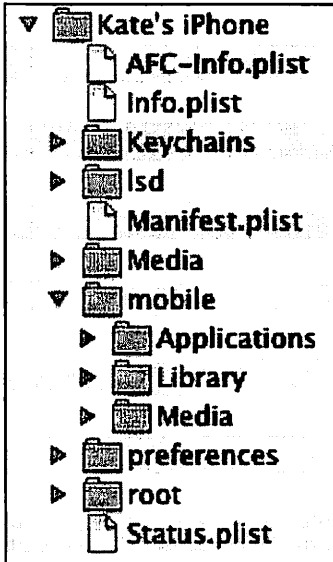
Acquisition Differences

- Backups/File System and Physical Acquisitions



Backups/File system extractions can be mapped back to directories found on physical images. You can see what all is missed by comparing the different types.

Note: `lsd` directory = `/db/lsd/`



Physical/Logical Acquisition How-To

Talk to the legal people

Determine if device can be jailbroken

- <https://www.theiphonewiki.com/wiki/Jailbreak>

Jailbreak the device

- Ensure you go to the legitimate Jailbreak website
- Follow directions carefully

Install OpenSSH via Cydia (if possible, if required)

- SSH sometimes preinstalled via Jailbreak (Dropbear, OpenSSH), otherwise install via Cydia
- SSH in using root/alpine and change password immediately (also change mobile/alpine!)
 - RootlessJB3 – Can't change password, System partition on writable. (/private/etc/master.passwd)

Dump Data using SSH using TAR

- `ssh root@127.0.0.1 -p 4242 '/jb/usr/bin/tar -c --posix f - /' > physical_logical.tar`

References:

iOS 10 and 11:

<https://www.mac4n6.com/blog/2018/1/7/ios-imaging-on-the-cheap-part-deux-for-ios-10-11>

Older iOS:

<https://www.mac4n6.com/blog/2016/3/23/ios-imaging-on-the-cheap>

Forgot the password you changed it to from 'alpine'?

https://www.gargan.org/en/Hardware_Toys/iPhone_iPod_touch/Reset_your_root_password/

Acquisition and Analysis Tools

Acquisition

Multi-platform

- BlackBag BlackLight, and Mobilyze
- libimobiledevice
- Now Secure Santoku
- Elcomsoft EIFT and EPB (Cloud)
- iTunes
- SSH/SFTP/AFC2
- Checkra1n JB - github.com/RealityNet/ios_bfu_triage
- Sysdiagnose - github.com/cheeky4n6monkey/iOS_sysdiagnose_forensic_scripts

Windows

- Cellebrite UFED (Cloud)
- Micro Systemation XRY
- Magnet Acquire
- Oxygen Forensic Suites (Cloud)

Analysis

Mac

- Native Mac OS

Multi-platform

- BlackBag, BlackLight, and Mobilyze
- Elcomsoft, EPV (Cloud)
- AFC utilities like iExplorer and iBackupBot
- F/OSS Scripts

Windows

- Cellebrite (Cloud)
- Micro Systemation XRY
- Magnet IEF/Axiom (Cloud)
- Oxygen Forensic Suites (Cloud)
- XWays/Encase/FTK

Depending on the platform that is being used for acquisition, you may choose to use different tools. It may also come down to what the budget allows! This list is not fully inclusive; there are new tools and utilities coming on the market all the time!

While few tools are Mac only, many tools are multi-platform between Mac and Windows. Some may even work on Linux (Santoku/libimobiledevice/SSH/SFTP).

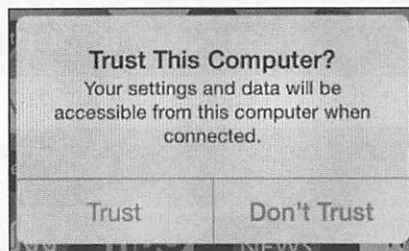
Each tool has its benefits and limitations. It is up to the analyst to determine which tool works best for their needs. Some analysts can simply not afford the 5-digit amounts that some of these tools command! There are free or cheaper options, so it's best to review all the capabilities of some of these tools to determine if all are required.

Tools for analysis differ as much as they do for acquisition. Some are free and some are very expensive. An analyst might choose one over another based on budget or if they do more iDevice backups versus physical acquisitions.

Many tools that analysts already have for disk analysis can be used for Physical/Logical-Physical/File System/Backup acquisitions. If you already have a tool that you are using, look into whether or not it supports the various file acquisition outputs that you get from your acquisition tools.

Passcode Bypass: "Trust This Computer?"

- Use Escrow Keybag/Lockdown Files
- System agnostic (Mac or PC)
- After first unlock
- May "time out"
- iOS 11- PIN required for trust



▼ Root	Dictionary	(9 items)
DeviceCertificate	Data	<2d2d2d2d 2d424547 494e2043 45525449 464
EscrowBag	Data	<44415441 000004f4 56455253 00000004 000
HostCertificate	Data	<2d2d2d2d 2d424547 494e2043 45525449 464
HostID	String	B198B9D6-67AE-40C5-B563-9A2B0A507E63
HostPrivateKey	Data	<2d2d2d2d 2d424547 494e2052 53412050 524
RootCertificate	Data	<2d2d2d2d 2d424547 494e2043 45525449 464
RootPrivateKey	Data	<2d2d2d2d 2d424547 494e2052 53412050 524
SystemUUID	String	1298A162-32FA-4969-927B-7B4E57876A79
WiFiMACAddress	String	00:26:08:79:8a:3b

Starting with iOS 7, once a device has been connected to a system, a "Trust This Computer?" window appears on the device. When the user selects "Trust", the lockdown files are created in the following locations.

```
OS X: /private/var/db/lockdown/
Windows XP: \Documents and Settings\\Application Data\Apple
Computer\Lockdown\
Windows Vista: \Users\\AppData\Roaming\Apple Computer\Lockdown\
Windows 7+: \Program Data\Apple\Lockdown\
```

Since these files are system agnostic, they may be copied to your analysis system or input into various acquisition software and used to access the iDevice. The lockdown files are plist files. Each plist file is named with the UDID of the device that has been trusted. In the screenshot above, this system has trusted six different iDevices. Each plist contains various certificates, keybags, and other information that can be used to access a locked device.

If provided an iDevice along with a computer system, we can determine if these devices are trusted devices using the paired files. These are located in /private/var/db/lockdown directory, they are only available on physical/file system dumps. The lockdown.log also contains timestamped records of when that trust relationship was created. Every time a device pairs with a system, a store_escrow_record gets created with the associated GUID. This is a nice way to corroborate when these records were created.

```
Mon Oct 28 19:25:20 2013 pid=45 (0x2ff0e000) store_escrow_record: Creating escrow bag
(hash=33e2c9b8fa39adcc4c75fa9f529e98a2b93d8e3d) for 9CB64DC2-F197-4BC3-82D7-6228B6C857D7
Mon Dec 23 17:53:33 2013 pid=45 (0x2fe93000) store_escrow_record: Creating escrow bag
(hash=aeb48b452dbc0e1c74fe1f44534a364ac3b65e59) for 85981632-B4E1-4B6F-ABDF-8689A6E0F0C4
Sat Apr 12 21:13:54 2014 pid=45 (0x2ff6e000) store_escrow_record: Creating escrow bag
(hash=9351849d7a04a6c87a02ebab55c8d0f58c58afef) for 2D874645-AD24-4E6D-81C3-689686486053
Sat Apr 12 21:14:44 2014 pid=45 (0x2ff7c000) store_escrow_record: Creating escrow bag
(hash=bb1659686d25846e4d314e41fad00734aba8b8e7) for 8C44265B-4510-48B1-8196-0FE5B47DFD54
Sat Apr 12 21:17:08 2014 pid=41 (0x2ff5d000) store_escrow_record: Creating escrow bag
(hash=8dbeee6bf93b41f3b7c85dc1589c1a65a8e36f0b) for 653446B0-EA63-41F1-9F00-F9EA85FCE13F
```

iOS Acquisition Caveats and Hints

Passcode is really important

- iOS 11 requires PIN for pairing certificate creation

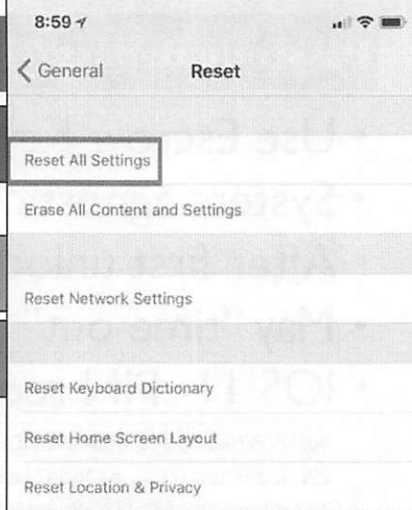
First unlock when using lockdown files

iOS backup password

- May be required for acquisition
- iOS 11: Remove backup password
 - Select “Reset All Settings”; not entirely forensically sound

iOS 11.4.1+: USB Restricted Mode

iOS 9+: Lockdown record expires w/o use after 6 months



Acquisition success is usually following the acquisition of the device passcode. Most access depends on that passcode.

If the device is powered off, lockdown files will be of no use, since the device will need to be unlocked with the passcode first (after startup) to use them.

The iOS backup password may be set on the device, which may limit creating a forensic backup. In iOS 11, Cindy Murphy discovered that resetting settings will also remove the backup passcode on the device. This action does, however, remove other settings, so it is not entirely forensically sound. Ensure you are keeping good notes and reasoning.

References:

iOS Security Guide: https://www.apple.com/business/docs/iOS_Security_Guide.pdf
<https://www.gillware.com/forensics/blog/digital-forensics-case-study/new-solution-encrypted-backups/>
USB Restricted Mode: <https://support.apple.com/en-us/HT208857>

iOS Backups: Types of Backups and Locations

iTunes

- Saved on OS X and Windows systems
- Manual backup (USB) or automatic (Wi-Fi)
- Unencrypted or encrypted
- Different formats (i.e., iOS 9-/iOS 10+)

iCloud

- Saved on Apple's servers
- "Encrypted"
- On-demand backup and/or automatic backup when connected to power and Wi-Fi
- iCloud backups / iCloud files

	iTunes Backup	iCloud Backup	Already on iCloud
Contacts	X		X
Calendar	X		X
Mail	X (Account Data)		X (Mail Messages)
Notes	X		X
Photos	X	X	X (Shared Photo Albums, Photo Library, Photo Stream)
Documents	X		X
Safari Data	X		X
Call History	X		
Health/Home	X (Encrypted Only)		
Keychain	X (Encrypted Only)		
Map Data	X		
Third-Party App Data	X	X	
Voicemail	X	X	
Purchase History		X	
Device Settings	X	X	
Messages	X	X	
Voice Messages	X		

iDevices can be backed up with iTunes locally on Mac or Windows systems, or in the iCloud.

On local systems, the iTunes backups use the same scheme, but the `/MobileSync/Backup/` directory location is different. iTunes backups may be backed up manually (USB), or automatically (Wi-Fi), and may use encryption or not.

Mac: `~/Library/Application Support/MobileSync/Backup/`

Windows XP: `\Documents and Settings\\Application Data\Apple Computer\MobileSync\Backup\`

Windows Vista+: `\Documents and Settings\Users\\AppData\Roaming\Apple Computer\MobileSync\Backup\`

Different types of iOS backups exist. Those on the local system, called iTunes Backups, can be encrypted or unencrypted, and are created by the user using iTunes. Different backups exist for iOS 9 and older as well as iOS 10.

iCloud backups are stored on Apple's servers in an encrypted, proprietary format. iCloud backups may be performed on demand or automatically but must be connected to power and Wi-Fi.

iTunes and iCloud backups may contain different items shown in the table above. If the user implements iCloud, data that is already synced with iCloud will not be backed up in the iCloud backups.

References:

<http://support.apple.com/kb/PH12519>

<http://support.apple.com/en-us/HT4946>

Artifacts on macOS Systems (Windows too!) ~/Library/Preferences/com.apple.iPod.plist

Model

Device Identifiers

- Serials
- MEID
- IMEI

iOS Version

Last Connected Timestamp

Device Connection (Use) Count

Key	Type	Value
Root	Dictionary	(3 items)
▼ Devices	Dictionary	(2 items)
▼ C8FF433A9B2112B0	Dictionary	(12 items)
Region Info	String	LL/A
Device Class	String	iPhone
IMEI	String	359407081420725
ID	String	C8FF433A9B2112B0
Serial Number	String	G8TVLBJCJCL9
Use Count	Number	1
Build Version	String	15B202
Family ID	Number	10,080
Firmware Version String	String	11.1.2
Connected	Date	Nov 26, 2017 at 5:47:25 PM
Firmware Version	Number	256
Product Type	String	iPhone10,6
▼ 756D85D85FEED0CE	Dictionary	(13 items)
Region Info	String	LL/A
Device Class	String	iPad
IMEI	String	359273061343901
ID	String	756D85D85FEED0CE
Serial Number	String	DLXQCOTWGHML
Use Count	Number	1
Build Version	String	14G60
Family ID	Number	10,045
Firmware Version String	String	10.3.3
Connected	Date	Dec 2, 2017 at 7:17:03 PM
Firmware Version	Number	256
Product Type	String	iPad5,2
MEID	String	35927306134390
com.apple.PreferenceSync.ExcludeAllSyncKeys	Boolean	YES
conn:128>Last Connect	Data	<0648acaf>

62

The `com.apple.iPod.plist` file located in the user's preferences directory contains all the iDevices attached to the system while logged in as that user.

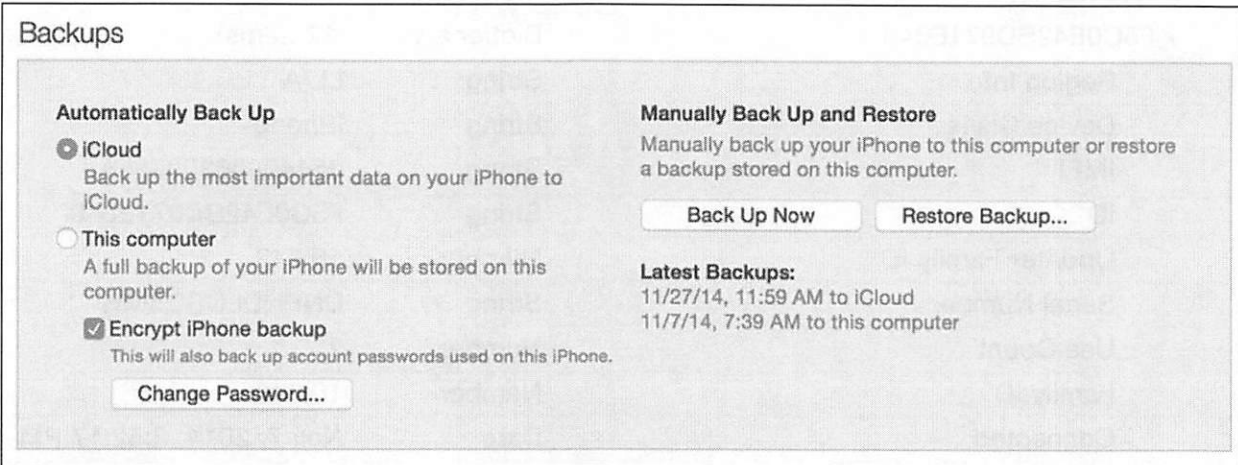
While the file is called `com.apple.iPod.plist`, it will contain information for iPods, iPads, and iPhones. Each 16-character alphanumeric key under `Devices` will contain the information for one device, to include:

- Device Class: The type of iDevice connected
- IMEI/MEID: Unique equipment identifiers
- Use Count: Number of times this device was connected
- Connected: The last time this device was connected
- Firmware Version String: The iOS version the device had when last connected

The `conn:128>Last Connect` key contains a hex representation of a Mac OS timestamp of the last device connection time in local system time.

Key	Type	Value
▼ Root	Dictionary	(3 items)
▼ Devices	Dictionary	(3 items)
▶ A3626E4E1B1F010E	Dictionary	(11 items)
▼ F5C0E42BD971E840	Dictionary	(12 items)
Region Info	String	LL/A
Device Class	String	iPhone
IMEI	String	354409063007558
ID	String	F5C0E42BD971E840
Updater Family ID	Number	10,042
Serial Number	String	DNPNDLQSG5MH
Use Count	Number	2
Family ID	Number	10,042
Connected	Date	Nov 7, 2014, 2:52:17 PM
Firmware Version String	String	8.1
Firmware Version	Number	256
MEID	String	35440906300755
▼ ADD8A302AB39D8EA	Dictionary	(11 items)
Device Class	String	iPhone
ID	String	ADD8A302AB39D8EA
Use Count	Number	2
Region Info	String	LL/A
IMEI	String	012659001279644
Firmware Version String	String	6.1.6
Updater Family ID	Number	10,004
Family ID	Number	10,004
Firmware Version	Number	256
Serial Number	String	851174G1EDG
Connected	Date	Nov 29, 2014, 11:20:43 AM
com.apple.PreferenceSync.ExcludeAllSyncKeys	Boolean	YES
conn:128>Last Connect	Data	<d09f5c8b>

iOS Backups: iTunes (or Finder with 10.15) Backups

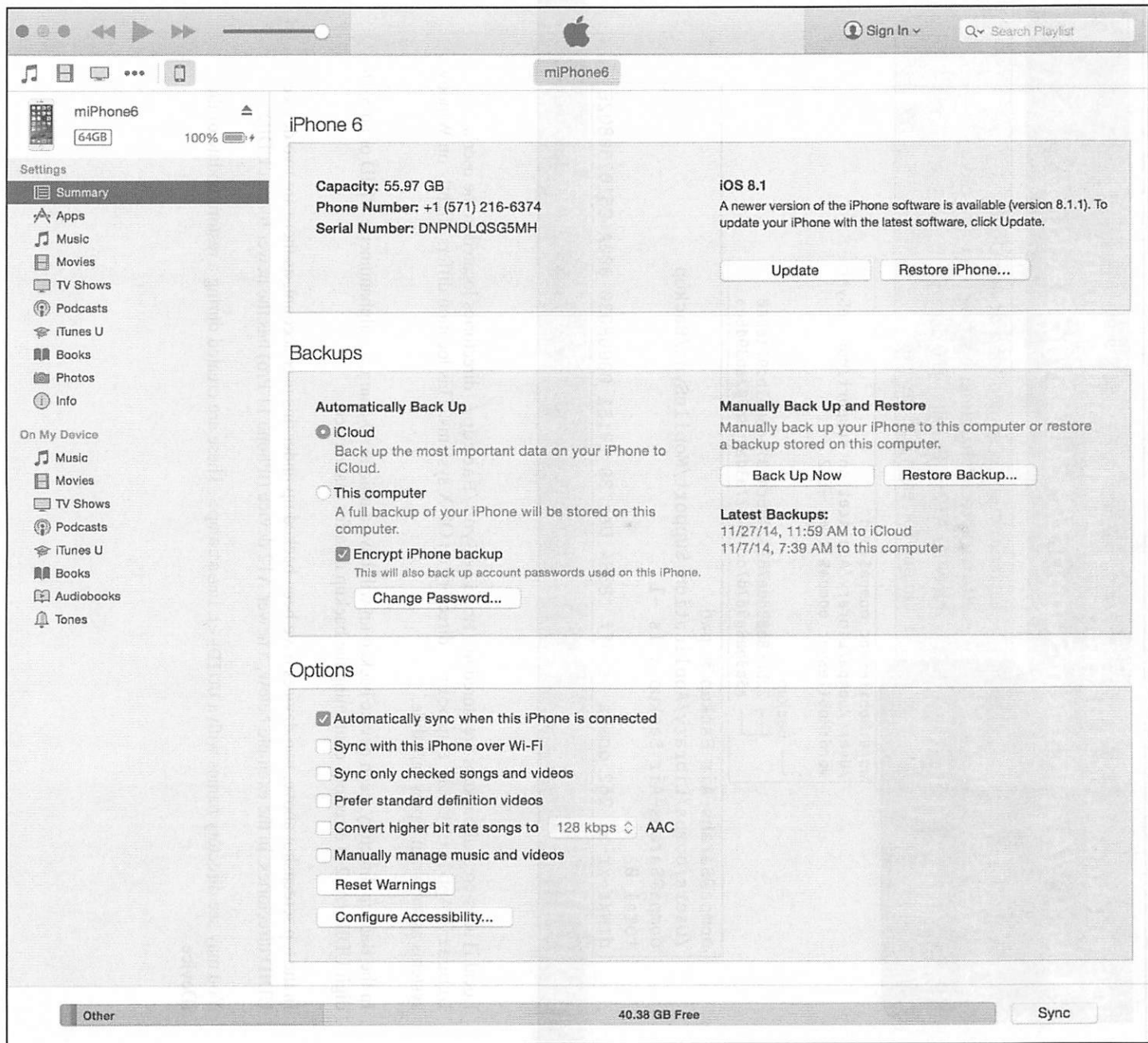


When a new iDevice connection is detected in iTunes (or just within Finder in macOS 10.15), a new “tab” will be available that looks like an iPhone/iPad/iPod icon. This view gives us basic information including the type of device, its capacity, name, iOS version (and updates available), and identifying information such as phone number (if available) and serial number.

The Backups section allows a user to select whether they want to back up their device using iCloud, iTunes, or both. Other information relating to previous backups or encryption may also be available.

The Options section allows a user to select other syncing options, such as whether to sync over Wi-Fi, automatically sync, and what to sync.

This section will focus on the types of backups, where backups are located on a system, what is backed up, and how to analyze backups.



iOS Backups: Local Backups—Location and Naming ~/Library/Application Support/MobileSync/Backup/

Universal Device Identifier (UDID)

- Each directory is named for a device UDID
- 40-character alphanumeric string (A11 and older)
- New ID A12+: [8 digits]-[16 digits]
- Unique for each iOS device

```
word:MobileSync oompa$ pwd
/Users/oompa/Library/Application Support/MobileSync
word:MobileSync oompa$ tree -L 2
```

```
├── Backup
│   ├── 22b8c8a80dde76332086c4a3f5c0e42bd971e840
│   └── a5a9ba8a967d7dc460677a3dadd8a302ab39d8ea
```

```
oompa@Sarahs-Air Backup % pwd
/Users/oompa/Library/Application Support/MobileSync/Backup
oompa@Sarahs-Air Backup % ls -l
total 0
drwxr-xr-x  262 oompa  staff  8384 Dec 30 19:51 00008030-000A4C510250802E
```

Local iTunes-created backups are stored in /MobileSync/Backup/ directories located in the user's Library/Application Support/ directory on OS X systems. This location differs slightly on Windows systems as shown in a previous slide.

In the Backup directory, each device backed up will have a unique 40-character alphanumeric UDID or an 8-16 digit UDID (A12+) directory containing the backup data for that device.

In the top screenshot above, two devices have been backed up under this user account, as there are two device UDID directories. In the example below, a newer A12 device (iPhone 11 Pro) has the newer format UDID.

You may see directory names with a UDID-`<timestamp>`. These are created during a restore/update of the iDevice.

iOS Backups: UDID Directory—Status.plist

SnapshotState	String	finished
Version	String	3.2
IsFullBackup	Boolean	NO
UUID	String	E75192A6-EE74-48EE-A40F-B090EEDA5F0F
Date	Date	Nov 26, 2017 at 6:15:51 PM
BackupState	String	new

The `Status.plist` file contains the following information:

- `SnapshotState`: Status of this backup snapshot
- `IsFullBackup`: Is this a full backup or not?
- `Date`: Timestamp of the backup
- `BackupState`: Type of Backup

iOS Backups: UDID Directory—Info.plist

Information Property List	Dictionary	(21 items)
▶ Applications	Dictionary	(139 items)
Build Version	String	15B202
Device Name	String	miPhoneX
Display Name	String	miPhoneX
GUID	String	25C20DB0DB0FA76A1CA4AB0EF89BBA24
ICCID	String	89014104279179902788
IMEI	String	359407081420725
▶ Installed Applications	Array	(139 items)
Last Backup Date	Date	2017-11-26T23:15:57Z
Phone Number	String	+1 (571) [REDACTED]
Product Name	String	iPhone X
Product Type	String	iPhone10,6
Product Version	String	11.1.2
Serial Number	String	G6TVLBJCJCL9
Target Identifier	String	e3fe6f9a2ec806326897fabfc8ff433a9b2112b0
Target Type	String	Device
Unique Identifier	String	E3FE6F9A2EC806326897FABFC8FF433A9B2112B0
iBooks Data 2	Data	<62706c69 73743030 d2010203 8253312e 325f10
▶ iTunes Files	Dictionary	(6 items)
▶ iTunes Settings	Dictionary	(1 item)
iTunes Version	String	12.7.1.14

The Info.plist file contains the following:

- Device Name
- Device Identifiers (GUID/ICCID/MEID/IMEI/Serial Number/UDID)
- Phone Number
- Make/Model/Build Data
- iOS Version
- Last Backup Date
- Installed Applications: Contains the bundle identifiers for each application installed, including native iOS apps.

If the applications were synced with an OS X system, you will see iPhone Application (IPA) files in the `~/Music/iTunes/iTunes Media/Mobile Applications/` directory. IPA files are ZIP archives of the iPhone applications.

Sometimes the `iTunes Settings/LibraryApplications/` key is populated with other applications that the user may have previously downloaded but does not currently have installed.

▶ iTunes Settings	Dictionary	(1 item)
▶ LibraryApplications	Array	(24 items)
Item 0	String	altsource.MyConsumerCellular
Item 1	String	cmmjhd01
Item 2	String	com.5thfinger.redshop.joann
Item 3	String	com.amazon.Lassen
Item 4	String	com.apple.iBooks
Item 5	String	com.chilingo.cuttherope
Item 6	String	com.clickgamer.AngryBirds

▼ Information Property List	Dictionary	(21 items)
▶ Applications	Dictionary	(139 items)
Build Version	String	15B202
Device Name	String	miPhoneX
Display Name	String	miPhoneX
GUID	String	25C20DB0DB0FA76A1CA4AB0EF89BBA24
ICCID	String	89014104279179902788
IMEI	String	359407081420725
▶ Installed Applications	Array	(139 items)
Last Backup Date	Date	2017-11-26T23:15:57Z
Phone Number	String	+1 (571) [REDACTED]
Product Name	String	iPhone X
Product Type	String	iPhone10,6
Product Version	String	11.1.2
Serial Number	String	G6TVLBJCJCL9
Target Identifier	String	e3fe6f9a2ec806326897fabfc8ff433a9b2112b0
Target Type	String	Device
Unique Identifier	String	E3FE6F9A2EC806326897FABFC8FF433A9B2112B0
iBooks Data 2	Data	<62706c69 73743030 d2010203 8253312e 325f10
▶ iTunes Files	Dictionary	(6 items)
▶ iTunes Settings	Dictionary	(1 item)
iTunes Version	String	12.7.1.14

iOS Backups: UDID Directory—Manifest.plist

IsEncrypted	Boolean	NO
Version	String	10.0
Date	Date	Nov 26, 2017 at 6:08:26 PM
SystemDomainsVersion	String	24.0
WasPasscodeSet	Boolean	YES
▼ Lockdown	Dictionary	(12 items)
▶ com.apple.MobileDeviceCrashCopy	Dictionary	(0 items)
▶ com.apple.TerminalFlashr	Dictionary	(0 items)
▶ com.apple.mobile.data_sync	Dictionary	(0 items)
▶ com.apple.Accessibility	Dictionary	(6 items)
ProductVersion	String	11.1.2
ProductType	String	iPhone10,6
BuildVersion	String	15B202
▶ com.apple.mobile.iTunes.accessories	Dictionary	(0 items)
▶ com.apple.mobile.wireless_lockdown	Dictionary	(1 item)
UniqueDeviceID	String	e3fe6f9a2ec806326897fabfc8ff433a9b2112b0
SerialNumber	String	G6TVLBJCJCL9
DeviceName	String	miPhoneX
▼ Applications	Dictionary	(706 items)
▶ com.xe.XECurrency.XECurrencyMessage	Dictionary	(3 items)
▶ com.google.Maps	Dictionary	(4 items)
▶ com.apple.CoreCDPUI.localSecretPrompt	Dictionary	(3 items)
▶ com.dayananetworks.myalitude.todaywidget	Dictionary	(3 items)
▶ com.contextoptional.OpenTable.Maps	Dictionary	(3 items)

The Manifest.plist file contains the following information:

- IsEncrypted: Is the backup encrypted or not?
- Date: When was the backup created?
- WasPasscodeSet: Was the passcode set on the device?

The Manifest.plist file also contains the Lockdown key, which contains device identification information such as name, serial number, and UDID. It also contains version information for the installed iOS, device make/model, and build data.

IsEncrypted	Boolean	NO
Version	String	10.0
Date	Date	Nov 26, 2017 at 6:08:26 PM
SystemDomainsVersion	String	24.0
WasPasscodeSet	Boolean	YES
▼ Lockdown	Dictionary	(12 items)
▶ com.apple.MobileDeviceCrashCopy	Dictionary	(0 items)
▶ com.apple.TerminalFlashr	Dictionary	(0 items)
▶ com.apple.mobile.data_sync	Dictionary	(0 items)
▶ com.apple.Accessibility	Dictionary	(6 items)
ProductVersion	String	11.1.2
ProductType	String	iPhone10,6
BuildVersion	String	15B202
▶ com.apple.mobile.iTunes.accessories	Dictionary	(0 items)
▶ com.apple.mobile.wireless_lockdown	Dictionary	(1 item)
UniqueDeviceID	String	e3fe6f9a2ec806326897fabfc8ff433a9b2112b0
SerialNumber	String	G6TVLBJCJCL9
DeviceName	String	miPhoneX
▼ Applications	Dictionary	(706 items)
▶ com.xe.XECurrency.XECurrencyMessage	Dictionary	(3 items)
▶ com.google.Maps	Dictionary	(4 items)
▶ com.apple.CoreCDPUI.localSecretPrompt	Dictionary	(3 items)
▶ com.dayananetworks.myaltitude.todaywidget	Dictionary	(3 items)
▶ com.contextoptional.OpenTable.Maps	Dictionary	(3 items)

iOS 10+ Backup: Manifest.db (Unencrypted) [1]

Table: Files

	fileID	domain	relativePath	flags	file
	Filter	Filter	Filter	Filter	Filter
324	6db31c77b29b247e0bad991a458cb7edabeb97d2	AppDomain-com.google.chrome.ios	Library/Preferences/.GlobalPreferences.plist	4	BLOB
325	e8d25bf5fe034266c8f62d2344239ead41a4d839	AppDomain-com.google.chrome.ios	Library/Cookies	2	BLOB
326	f8719258535ebf4c1647327c15d569c2d69c0fec	AppDomain-com.google.chrome.ios	Library/Application Support	2	BLOB
327	c7b1ef7d7b32ac1d15d18fde05ccf52166be4cff	AppDomain-com.google.chrome.ios	Library/Application Support/Google	2	BLOB
328	d681ca2b563b0d5bd13a6486089dcde7b3c9ead3	AppDomain-com.google.chrome.ios	Library/Application Support/Google/RLZ	2	BLOB
329	de6e7d4fc4d46348302f1be92e3f5fe684ca7d3a	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome	2	BLOB
330	1666e3f01b893352e821fde2ecc22527734cfba1	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome/Default	2	BLOB
331	3ca82df2993b7daca471a13ce7c3743557aaf352	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome/Default/Thumbnail	2	BLOB
332	3a1a72d9879f24917d4c79ac207b43e1e9cb57c8	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome/Default/Sync Data	2	BLOB
333	972e270ad38a7fc8603cdf4057c736266677908	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome/Default/GCM Store	2	BLOB
334	98f937d09c471804b55236dfabcb90cc9ed8fa32	AppDomain-com.google.chrome.ios	Documents	2	BLOB
335	9fa6b36c65f1f90e3eba6ce1bddb8fd70b9b99d	AppDomain-com.google.chrome.ios	Documents/ChromeToDevice	2	BLOB

The Manifest.db file contains the same data as found in the older Manifest.mbdb file but in an SQLite database structure. The BLOB in the file column contains the specific file metadata for each backed up file.

iOS 10+ Backup: Manifest.db (Unencrypted) [2]

- File Metadata BLOB
- Binary Plist
- NSKeyedArchiver
- Contains File Metadata:
 - Timestamps
 - Size
 - Inode
 - Protection Classes
 - UID/GID
 - Etc.

Key	Type	Value
▼ Root	Dictionary	(4 items)
\$version	Number	100,000
▼ \$objects	Array	(4 items)
Item 0	String	\$null
▼ Item 1	Dictionary	(9 items)
UserID	Number	501
Mode	Number	16,877
LastModified	Number	1,442,681,217
Size	Number	0
InodeNumber	Number	39,182
LastStatusChange	Number	1,473,869,351
GroupID	Number	501
Birth	Number	1,441,331,275
ProtectionClass	Number	0
Item 2	String	Library
▼ Item 3	Dictionary	(2 items)
\$classname	String	MBFile
▼ \$classes	Array	(2 items)
Item 0	String	MBFile
Item 1	String	NSObject
\$sarchiver	String	NSKeyedArchiver
▼ \$top	Dictionary	(0 items)

The BLOB in the file column contains the specific file metadata for each backed up file in an NSKeyedArchiver format.

The encrypted version of the Manifest.db file is now fully encrypted and has no metadata available. In very early versions of iOS 10, the file paths were still available as seen below; however, the metadata was still encrypted and encoded into base64.

fileID	domain	relativePath	flags	file
340	AppDomain-com.google.chrome.ios	Library/Preferences/.GlobalPreferences.plist	4	4K9zzDGp0GjLq77qoNI41Au...
341	AppDomain-com.google.chrome.ios	Library/Cookies	2	VVxQNJQUZnMrRVfSvQVHCp...
342	AppDomain-com.google.chrome.ios	Library/Application Support	2	VVxQNJQUZnMrRVfSvQVHCp...
343	AppDomain-com.google.chrome.ios	Library/Application Support/Google	2	VVxQNJQUZnMrRVfSvQVHCp...
344	AppDomain-com.google.chrome.ios	Library/Application Support/Google/RLZ	2	VVxQNJQUZnMrRVfSvQVHCp...
345	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome	2	VVxQNJQUZnMrRVfSvQVHCp...
346	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome/Default	2	VVxQNJQUZnMrRVfSvQVHCp...
347	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome/Default/Thumbnail	2	/4HD32OKyR1h0IdSxQq0Wy...
348	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome/Default/Sync Data	2	/4HD32OKyR1h0IdSxQq0Wy...
349	AppDomain-com.google.chrome.ios	Library/Application Support/Google/Chrome/Default/GCM Store	2	/4HD32OKyR1h0IdSxQq0Wy...
350	AppDomain-com.google.chrome.ios	Documents	2	TOWLvyekCdaf94Q5/qyenUj...
351	AppDomain-com.google.chrome.ios	Documents/ChromeToDevice	2	/4HD32OKyR1h0IdSxQq0Wy...

Cracking iTunes Encrypted Backups

Password in user's login keychain?

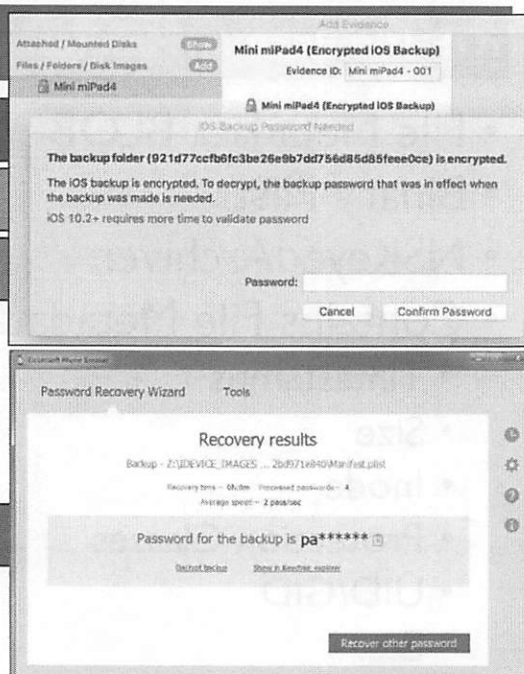
Try other known passwords/patterns

Elcomsoft Phone Breaker

- Brute force for Windows only
- (Mac: Decrypt with known password)
- Home/Professional/Forensic Editions
- Demo version available
- Full version required to retrieve passcode
- Determine if passcode cracking is possible!

Other tools

- Oxygen Forensic Extractor
- Passware
- Hashcat (Free!)



Cracking into an encrypted iTunes backup is possible. There are a few different ways to get into them, and some are easier/faster than others.

I would first look into reviewing the user's login keychain file. This is easy if you have the user's login password, but can also be brute forced, as described in this course. I would try running a strings query first to see if the "iPhone/iPad Backup" password is stored in the keychain file.

You may also want to try various password combinations. You get unlimited tries but it can be a manual process.

Lastly, you can try brute forcing the backup itself. There are a few tools on the market right now that support this feature. One that will be described is Elcomsoft Phone Breaker. The demo version of this tool can be used to see if it is possible to get the password. If the analyst is lucky with their files and dictionaries, they will be presented with the screenshot above. Since this is the demo version only, the first two characters are provided. To get the full password, you will need to buy the product, but at least you know it's possible to break it! The analyst may only need the first couple characters to start guessing the password. Maybe it's the start of a family name or pet! In the screenshot above, this was the start to a very unsecure password of "password"!

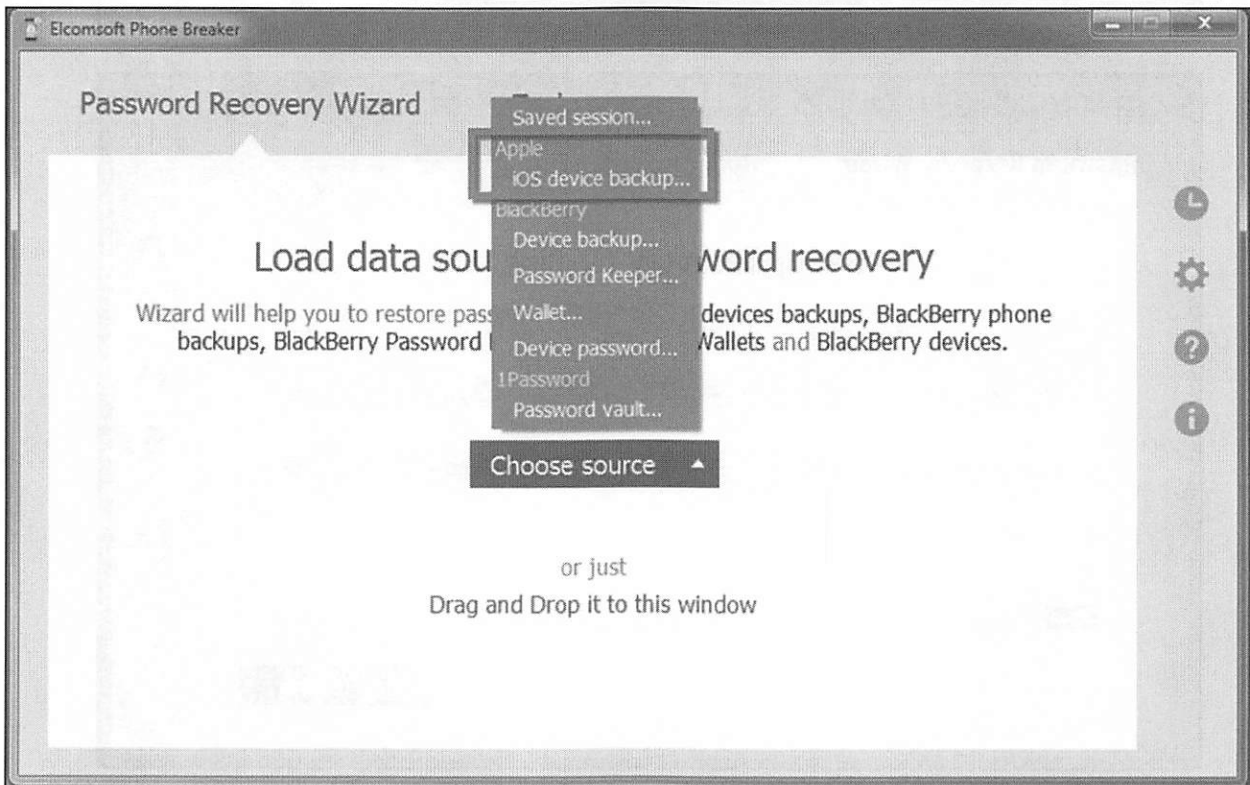
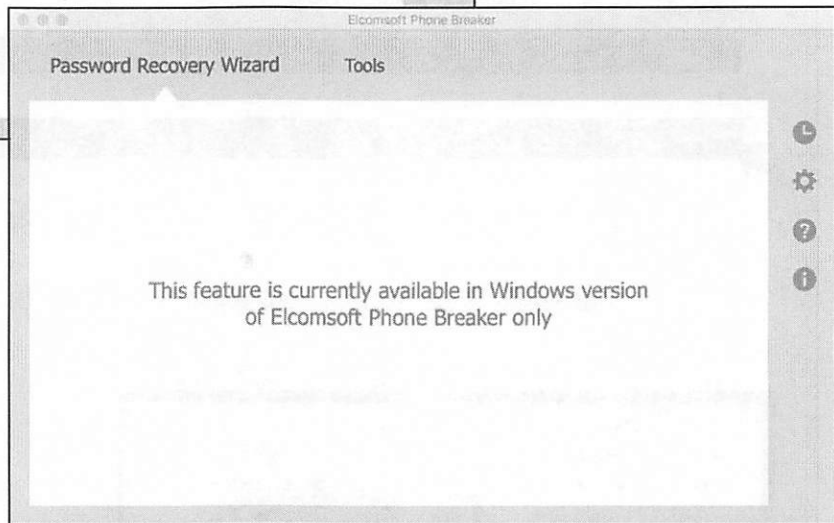
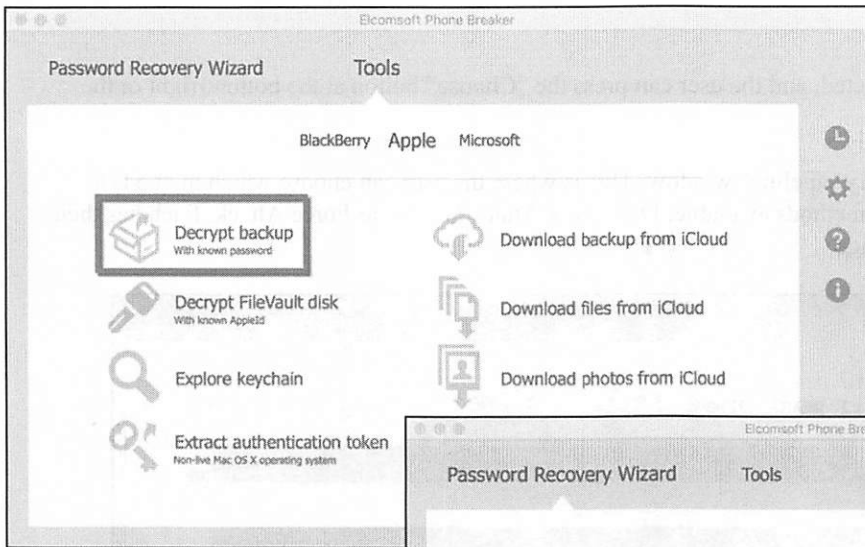
Using the demo version of Elcomsoft Phone Breaker, we can select the "Decrypt Backup" button and feed it the encrypted backup. It is worth noting at the time of this writing that only the Windows version of this tool supports the iTunes decrypt feature as shown in the screenshot above.

Select the "iOS device backup..." from the "Apple" menu in the "Choose source" listing. You can also just drag and drop the backup folder to this window.

References:

<http://www.oxygen-forensic.com/en/products/oxygen-forensic-detective/detective/encrypted-backups>

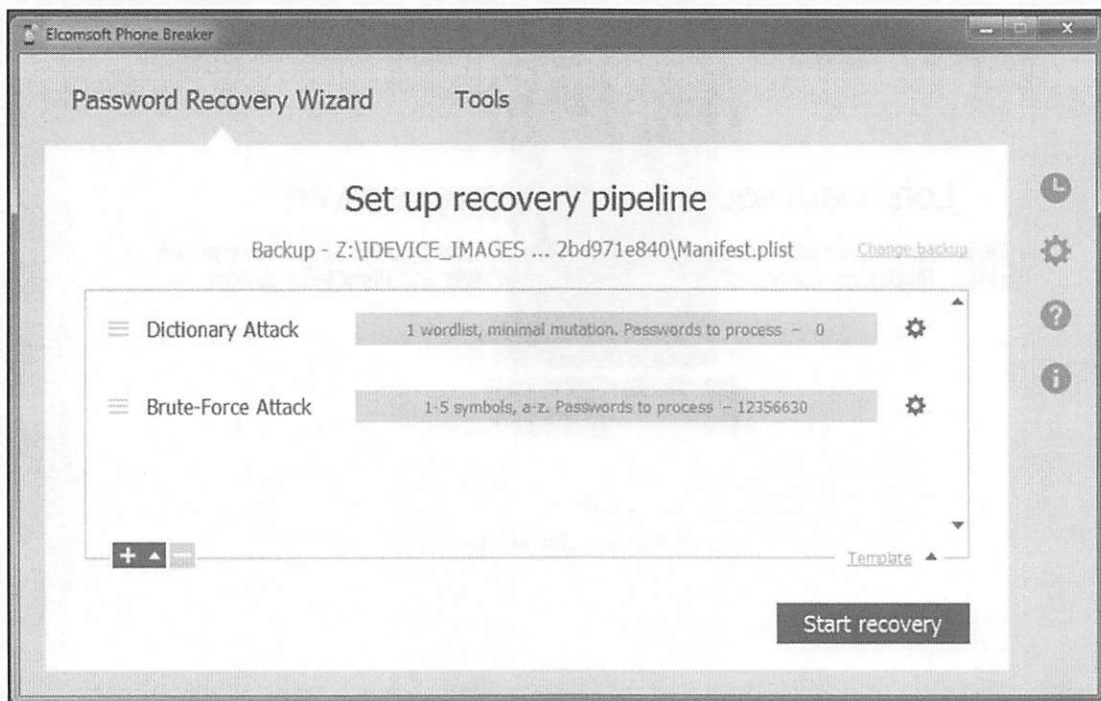
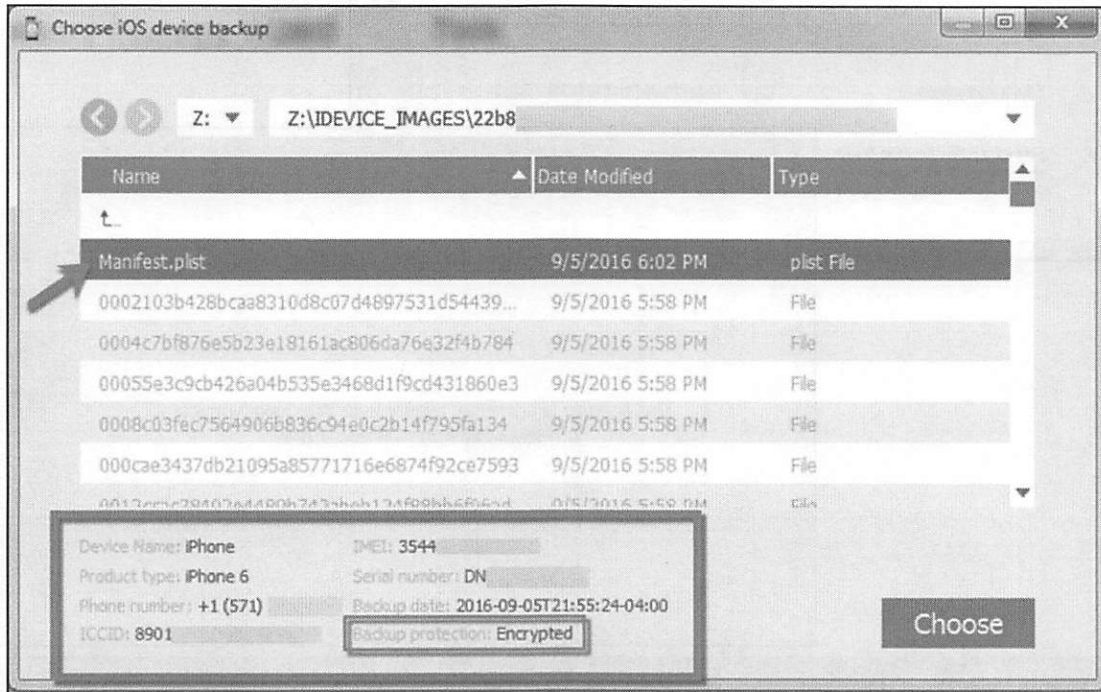
Hashcat Cracking: <https://www.youtube.com/watch?v=MMYsnPzsPYU>



Once loaded, EPB will show some of the device information, such as device name and product type, as well as identifying information.

The Manifest.plist file should be selected, and the user can press the “Choose” button at the bottom-right of the window.

This will bring up the “Set up recovery pipeline” window. This is where the user can choose which methods to acquire the password. There are two methods available, Dictionary Attack and Brute-Force Attack. Each has their own settings described on the next page.

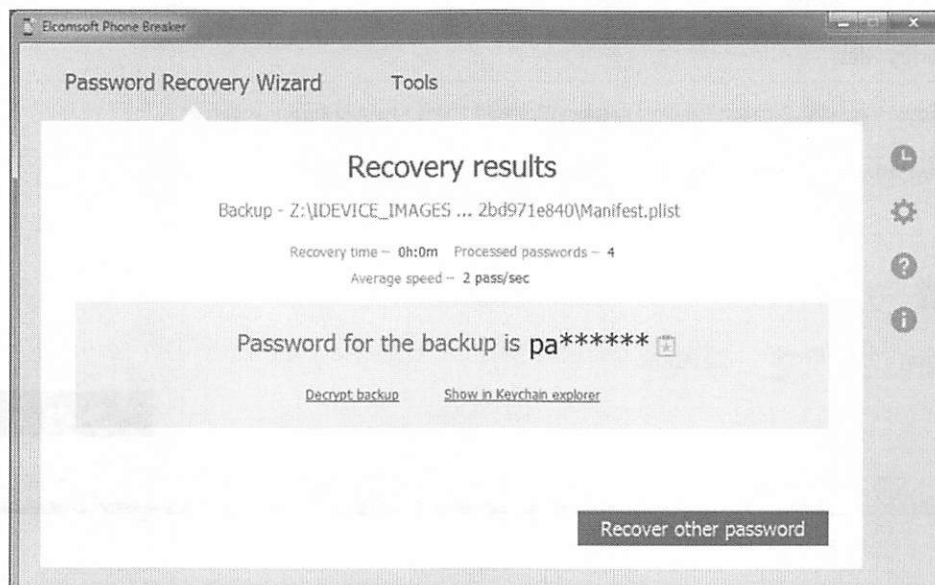
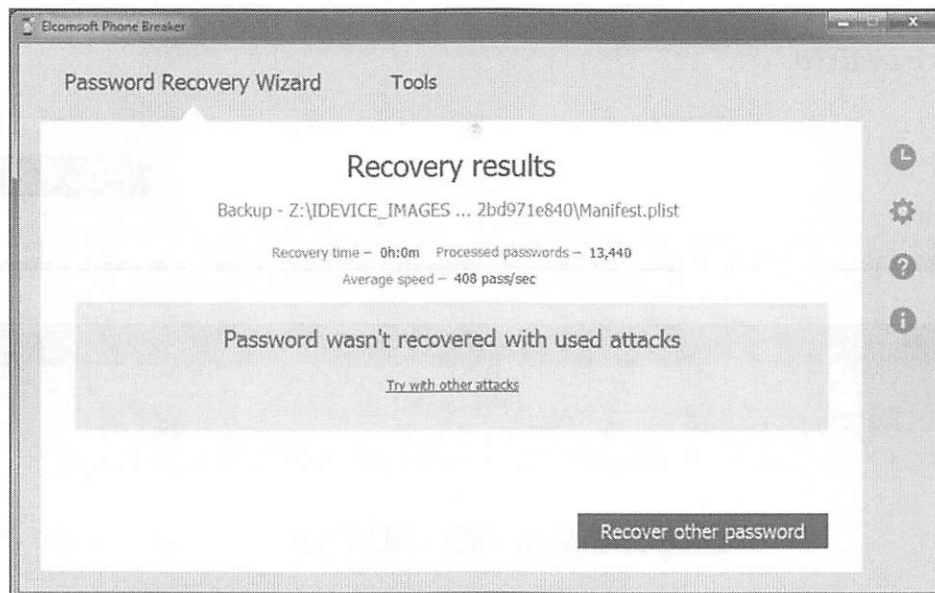


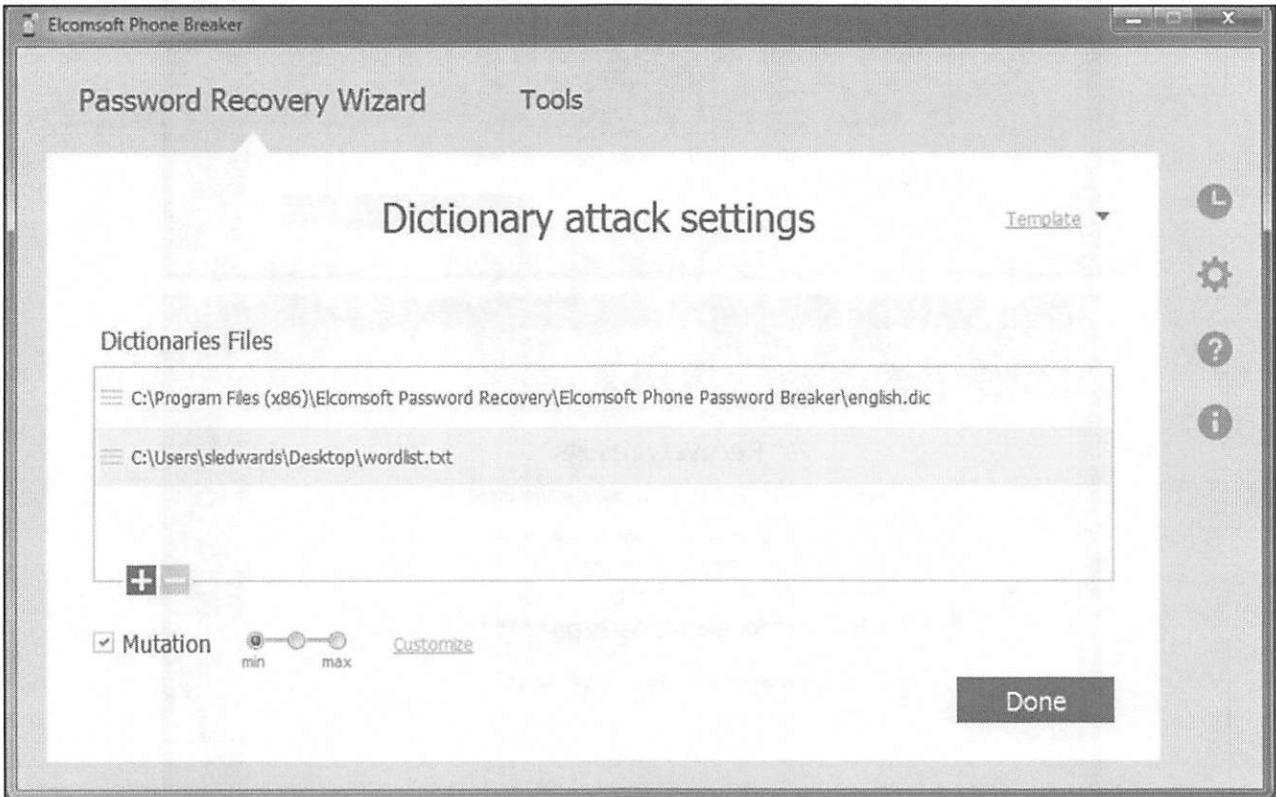
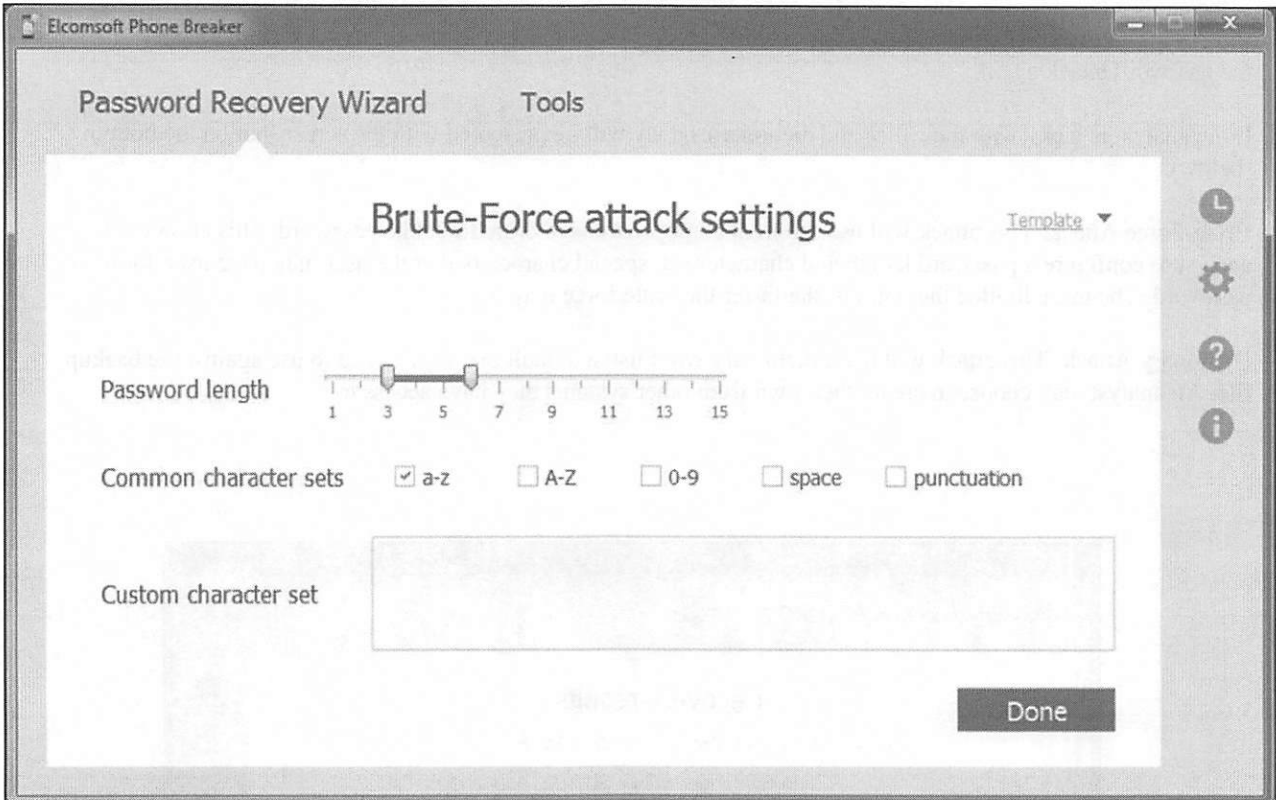
If the analyst has not provided a good dictionary file or brute force configurations, they will be shown the screenshot on the top (below). The analyst will have to provide additional dictionaries or configurations to attempt the password attack again.

If the analyst is lucky with their files and dictionaries, they will be presented with the screenshot on the bottom (below).

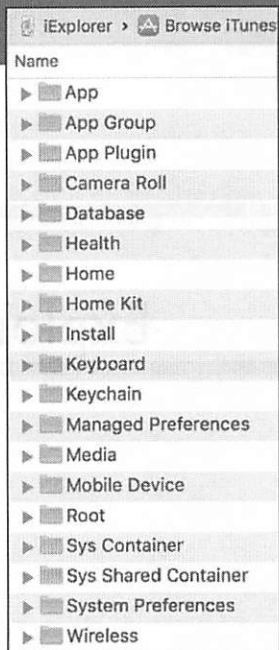
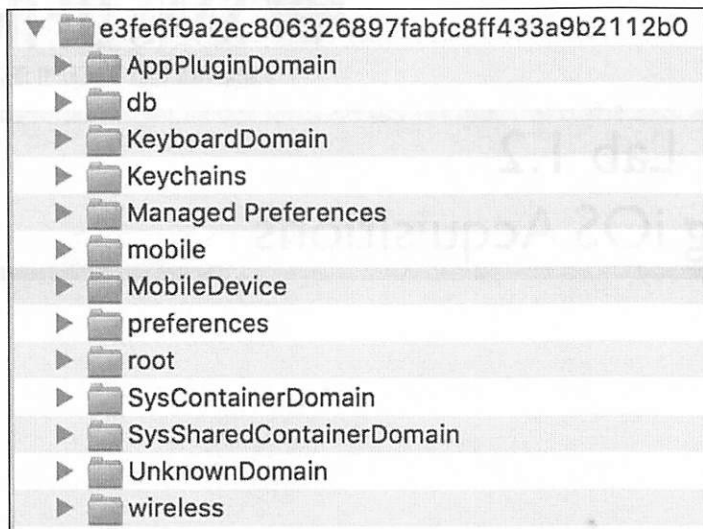
Brute-Force Attack: This attack will use different configurations to brute force the password. This allows the analyst to configure a password length and character set, special characters that the user may have used for their password. The more limited the options, the faster the brute force may be.

Dictionary Attack: This attack will use a dictionary word list; a default one is provided to use against the backup file. An analyst may choose to create their own from other systems they have access to.





Normalization of "Backup File System"



Once opened, a backup file will look very similar to a File System acquisition of a device. Each backup analysis tool may organize the "file system" of the backup differently.

The two examples above are the same backup. On the left it is shown in Blacklight, while on the right it is shown with iExplorer.



SANS DFIR

DIGITAL FORENSICS & INCIDENT RESPONSE

Lab 1.2

Exploring iOS Acquisitions

SANS DFIR

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 80

This page intentionally left blank.

Section 1: Agenda

Part 1: Apple Essentials

Part 2: Mac Essentials and Acquisition

Part 3: iOS Essentials and Acquisition

Part 4: Disks and Partitions

This page intentionally left blank.



SANS DFIR

DIGITAL FORENSICS & INCIDENT RESPONSE

Section I: Part 4

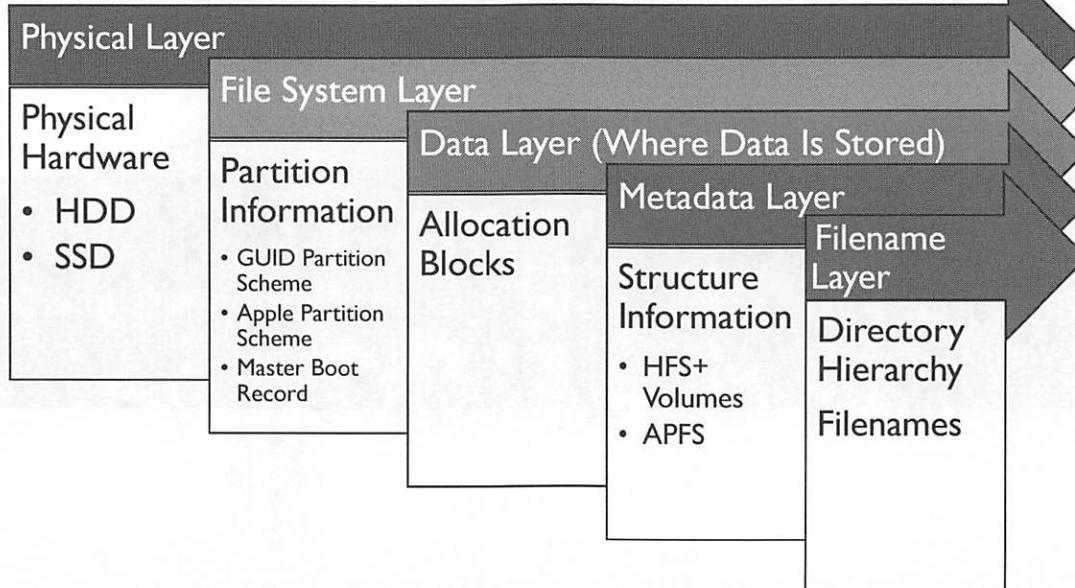
Disks and Partitions

SANS DFIR

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 82

This page intentionally left blank.

File System: Five Layers



A file system is comprised of five layers: Physical, file system, data, metadata, and filename. These will be discussed in this section in detail.

The physical and file system layers contain non-specific file system information such as the hardware and partitioning scheme. These will be discussed first to give a basis for the file system data that is presented on top of these structures.

The primary partitioning schemes found on Mac systems are the Globally Unique Identifier (GUID) Partition Scheme, Apple Partition Scheme, and the Master Boot Record (MBR) partitions. Mac systems primarily use three types of partition schemes for different purposes. A default installation of macOS will partition the drive using the GUID Partition Scheme, creating an Extensible Firmware Interface (EFI) partition and a protective MBR. Disk image files (.dmg) can use the GUID Partition Scheme, the Apple Partition Scheme, a Master Boot Record, or no partition at all, depending on what the user wants to do with the disk image.

The Allocation blocks used in the data layer are analogous to “Clusters” used in NTFS file systems.

The data, metadata, and filename layers contain the majority of the data that is presented and reviewed by forensic analysts.

Apple Partition Map vs. GUID Partition Table

APM

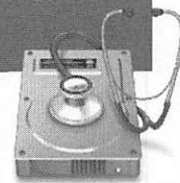
- Open Firmware
- PowerPC

GPT

- Extensible Firmware Interface
- Intel x86

Mac OS X installations on PowerPC hardware will have an Apple Partition Map and use Open Firmware, while Intel x86-based systems will implement GUID Partition Table with the Extensible Firmware Interface.

“diskutil list” Command: HFS+ Disks



```
byte:~ ompa$ diskutil list  
/dev/disk0
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	GUID_partition_scheme		*500.1 GB	disk0
1:	EFI		209.7 MB	disk0s1
2:	Apple_HFS	Macintosh HD	499.2 GB	disk0s2
3:	Apple_Boot	Recovery HD	650.0 MB	disk0s3

```
/dev/disk0 (internal, physical):
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	GUID_partition_scheme		*500.3 GB	disk0
1:	EFI	EFI	209.7 MB	disk0s1
2:	Apple_CoreStorage	Sierra	499.4 GB	disk0s2
3:	Apple_Boot	Recovery HD	650.0 MB	disk0s3

```
/dev/disk1 (internal, virtual):
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:		Sierra	+499.1 GB	disk1
		Logical Volume on disk0s2		
		DC2AA68E-4956-4745-BA70-28E8E7155818		
		Unlocked Encrypted		

To view the disk partitions on the drives associated with a specific Mac, you can issue the `diskutil list` command. The output of the command shown in the screenshot above shows two types of internal operating system disks. The output above also shows the number of partitions each disk contains, with the size and disk identifier.

This disk identifier is formatted in the format `disk#s#`, where the first number is the disk number and the second is the partition, or “slice”. This will be used later to reference a specific partition on a drive.

The same information can be found using the `Disk Utility.app` application located in `/Applications/Utilities/`.

The top screenshot shows an older system with no CoreStorage or FileVault and using an older version of the `diskutil` command.

- `/dev/disk0` contains the Mac OS X operating system. This disk uses the GUID Partition Scheme (shown as `GUID_partition_scheme`). Note the EFI partition, Macintosh HD HFS+ partition, and Recovery HD volumes (10.7 introduced Recovery HD volumes).

The bottom screenshot shows a newer `diskutil` output for the OS disk. This newer output includes terms that show whether the disk is internal or external, physical or virtual, as well as other items associated with CoreStorage, FileVault, and other metadata.

- `/dev/disk0` is a physical and internal drive that contains a GPT partition scheme, an EFI and Recovery partition, and an HFS+ CoreStorage volume named “Sierra”. This disk is using FileVault and CoreStorage, so it will mount a virtualized version of this “Sierra” volume for the user to interact with on `/dev/disk1`.
- `/dev/disk1` is an internal but virtual disk containing the “unlocked” but still encrypted FileVault disk. This is the drive you would want to image to get a logical view of the file system without having to decrypt the drive on `/dev/disk0`.

```

[word:~ oompa$ diskutil list
/dev/disk0 (internal, physical):
#:          TYPE NAME                SIZE      IDENTIFIER
0:          GUID_partition_scheme    *500.3 GB disk0
1:          EFI EFI                  209.7 MB  disk0s1
2:          Apple_CoreStorage Sierra  499.4 GB  disk0s2
3:          Apple_Boot Recovery HD   650.0 MB  disk0s3

/dev/disk1 (internal, virtual):
#:          TYPE NAME                SIZE      IDENTIFIER
0:          Sierra                    +499.1 GB disk1
           Logical Volume on disk0s2
           DC2AA68E-4956-4745-BA70-28E8E7155818
           Unlocked Encrypted

/dev/disk2 (internal, physical):
#:          TYPE NAME                SIZE      IDENTIFIER
0:          FDisk_partition_scheme    *519.6 GB disk2
1:          Apple_HFS SDCARD         519.5 GB  disk2s1

/dev/disk3 (external, physical):
#:          TYPE NAME                SIZE      IDENTIFIER
0:          FDisk_partition_scheme    *15.9 GB  disk3
1:          DOS_FAT_32 DISK_IMG       15.9 GB  disk3s1

/dev/disk4 (external, physical):
#:          TYPE NAME                SIZE      IDENTIFIER
0:          GUID_partition_scheme    *2.0 TB   disk4
1:          EFI EFI                  209.7 MB  disk4s1
2:          Apple_HFS DATA          2.0 TB   disk4s2

/dev/disk5 (disk image):
#:          TYPE NAME                SIZE      IDENTIFIER
0:          GUID_partition_scheme    +268.6 MB disk5
1:          Apple_HFS Google Earth   268.5 MB  disk5s1

/dev/disk6 (disk image):
#:          TYPE NAME                SIZE      IDENTIFIER
0:          Apple_partition_scheme    +29.4 MB  disk6
1:          Apple_partition_map      32.3 KB   disk6s1
2:          Apple_HFS KeePassX       29.3 MB   disk6s2

/dev/disk7 (disk image):
#:          TYPE NAME                SIZE      IDENTIFIER
0:          FUSE for macOS            +16.8 MB  disk7

```

“diskutil list” Command: APFS Disk (w/ or w/o FileVault)

```
MacBook-Pro:/ oompa$ diskutil list
/dev/disk0 (internal):
#:#:          TYPE NAME          SIZE          IDENTIFIER
0:  GUID_partition_scheme      1.0 TB        disk0
1:  EFI EFI                    314.6 MB      disk0s1
2:  Apple_APFS Container disk1  1.0 TB        disk0s2

/dev/disk1 (synthesized):
#:#:          TYPE NAME          SIZE          IDENTIFIER
0:  APFS Container Scheme -      +1.0 TB       disk1
    Physical Store disk0s2
1:  APFS Volume HighSierra      313.7 GB      disk1s1
2:  APFS Volume Preboot         23.4 MB       disk1s2
3:  APFS Volume Recovery        520.8 MB      disk1s3
4:  APFS Volume VM              2.1 GB        disk1s4
```

APFS introduces a new disk structure. It still implements the GPT partitioning scheme and has an EFI volume; however, everything else is quite different. /dev/disk0 contains the APFS Container within which the other volumes are contained, as shown in the “synthesized” /dev/disk1 using the “APFS Container Scheme”.

- /dev/disk1s1: The OS volume, contains user data.
- /dev/disk1s2: The Preboot volume, contains data to boot the system.
- /dev/disk1s3: The Recovery volume.
- /dev/disk1s4: The VM volume, contains the swap and hibernate images.

APFS Containers may contain additional data volumes and may look like the one below that had another volume added to it named “AnotherAPFSVolume”.

```
oompas-Mac:~ oompa$ diskutil list
/dev/disk0 (internal, physical):
#:#:          TYPE NAME          SIZE          IDENTIFIER
0:  GUID_partition_scheme      *42.9 GB      disk0
1:  EFI EFI                    209.7 MB      disk0s1
2:  Apple_APFS Container disk1  42.7 GB       disk0s2

/dev/disk1 (synthesized):
#:#:          TYPE NAME          SIZE          IDENTIFIER
0:  APFS Container Scheme -      +42.7 GB      disk1
    Physical Store disk0s2
1:  APFS Volume Macintosh HD    18.6 GB       disk1s1
2:  APFS Volume Preboot         20.8 MB       disk1s2
3:  APFS Volume Recovery        520.8 MB      disk1s3
4:  APFS Volume VM              20.5 KB       disk1s4
5:  APFS Volume AnotherAPFSVolume 1.1 MB        disk1s5
```

Reference:
Man Page: [newfs_apfs](#)

“diskutil list” Command: APFS Disk (w/ or w/o FileVault) 10.15+

```
oompa@Sarahs-Air ~ % diskutil list
/dev/disk0 (internal, physical):
#:          TYPE NAME              SIZE          IDENTIFIER
0:          GUID_partition_scheme  *1.5 TB       disk0
1:          EFI EFI                 314.6 MB      disk0s1
2:          Apple_APFS Container disk1  1.5 TB        disk0s2

/dev/disk1 (synthesized):
#:          TYPE NAME              SIZE          IDENTIFIER
0:          APFS Container Scheme -   +1.5 TB       disk1
              Physical Store disk0s2
1:          APFS Volume Macintosh HD - Data  1.0 TB        disk1s1
2:          APFS Volume Preboot             94.1 MB       disk1s2
3:          APFS Volume Recovery            528.5 MB      disk1s3
4:          APFS Volume VM                  4.3 GB        disk1s4
5:          APFS Volume Macintosh HD        11.0 GB       disk1s5
```

```
oompa@Sarahs-Air ~ % mount
/dev/disk1s5 on / (apfs, local, read-only, journaled)
devfs on /dev (devfs, local, nobrowse)
/dev/disk1s1 on /System/Volumes/Data (apfs, local, journaled, nobrowse)
/dev/disk1s4 on /private/var/vm (apfs, local, journaled, nobrowse)
map auto_home on /System/Volumes/Data/home (autofs, automounted, nobrowse)
```

With 10.15 Catalina, the System and Data partitions have now been split (like iOS devices have been for years!)

The top screenshot shows the same ‘diskutil list’ output with a small change. /dev/disk1s1 is the Data partition, while /dev/disk1s5 is the new System partition.

The bottom screenshot shows the mount points for these partitions. Mounted at the root ‘/’ is the new read only System partition with the operating system, while the Data partition is mounted on /System/Volumes/Data/.

“diskutil list” Command: Disk Images



```
/dev/disk5 (disk image):
#:          TYPE NAME          SIZE      IDENTIFIER
0:          GUID_partition_scheme  +268.6 MB  disk5
1:          Apple_HFS Google Earth  268.5 MB  disk5s1

/dev/disk6 (disk image):
#:          TYPE NAME          SIZE      IDENTIFIER
0:          Apple_partition_scheme  +29.4 MB   disk6
1:          Apple_partition_map     32.3 KB   disk6s1
2:          Apple_HFS KeePassX     29.3 MB   disk6s2

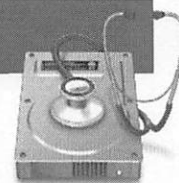
/dev/disk7 (disk image):
#:          TYPE NAME          SIZE      IDENTIFIER
0:          FUSE for macOS          +16.8 MB   disk7
```

The `diskutil list` output also shows information pertaining to Disk Images and labels them as “disk image”.

In the screenshot above, you can see that the format may be different for each disk image. Three software DMG files were opened and mounted on this system.

The first on `/dev/disk5`, Google Earth, shows that it uses a GPT partitioning scheme. The second on `/dev/disk6`, KeePassX, shows it uses the Apple Partitioning Scheme, while the third on `/dev/disk7`, FUSE for macOS, uses no partitioning scheme.

“diskutil list” Command: External Media



```
/dev/disk2 (internal, physical):
#:#:      TYPE NAME              SIZE      IDENTIFIER
0:      FDisk_partition_scheme *519.6 GB disk2
1:      Apple_HFS SDCARD          519.5 GB  disk2s1

/dev/disk3 (external, physical):
#:#:      TYPE NAME              SIZE      IDENTIFIER
0:      GUID_partition_scheme *2.0 TB  disk3
1:      EFI EFI                209.7 MB  disk3s1
2:      Apple_HFS DATA          1.9 TB   disk3s2
3:      Apple_HFS MORE DATA     55.5 GB  disk3s3

/dev/disk4 (external, physical):
#:#:      TYPE NAME              SIZE      IDENTIFIER
0:      FDisk_partition_scheme *15.9 GB  disk4
1:      DOS_FAT_32 DISK_IMG       15.9 GB  disk4s1
```

External media may also show up in the `diskutil list` output.

- `/dev/disk2`: This is an HFS+ formatted volume named “SDCARD”. Notice how it shows as an “internal, physical” drive. This is an SD card inserted into the physical/internal SD card slot of a laptop system.
- `/dev/disk3`: This drive is an external hard drive that shows a few partitions. This disk is using GPT and has three partitions: EFI, and two HFS+ partitions named “DATA” and “MORE DATA”.
- `/dev/disk4`: This is a thumb drive that uses an MBR (FDisk_partition_scheme) that contains a FAT32 volume named “DISK_IMG”.

```
byte:~ oompa$ diskutil list
```

```
/dev/disk0
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	GUID_partition_scheme		*500.1 GB	disk0
1:	EFI		209.7 MB	disk0s1
2:	Apple_HFS	Macintosh HD	499.2 GB	disk0s2
3:	Apple_Boot	Recovery HD	650.0 MB	disk0s3

```
/dev/disk1
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	FDisk_partition_scheme		*8.0 GB	disk1
1:	DOS_FAT_32	NO NAME	8.0 GB	disk1s1

```
/dev/disk2
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	FDisk_partition_scheme		*2.0 TB	disk2
1:	Windows_NTFS	WDPassport	2.0 TB	disk2s1

```
/dev/disk3
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	FDisk_partition_scheme		*3.5 GB	disk3
1:	DOS_FAT_32	Kindle	3.5 GB	disk3s1

```
/dev/disk4
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	FDisk_partition_scheme		*1.0 GB	disk4
1:	DOS_FAT_16	ORANGE	1.0 GB	disk4s1

“diskutil info” Command on a Disk

<pre>byte:~ oompa\$ diskutil info disk0 Device Identifier: disk0 Device Node: /dev/disk0 Part of Whole: disk0 Device / Media Name: TOSHIBA MK5065GSXF Media Volume Name: Not applicable (no file system) Mounted: Not applicable (no file system) File System: None Content (IOContent): GUID_partition_scheme OS Can Be Installed: No Media Type: Generic Protocol: SATA SMART Status: Verified Total Size: 500.1 GB (500107862016 Bytes) (exactly 976773168 512-Byte-Blocks) Volume Free Space: Not applicable (no file system) Device Block Size: 512 Bytes Read-Only Media: No Read-Only Volume: Not applicable (no file system) Ejectable: No Whole: Yes Internal: Yes Solid State: No OS 9 Drivers: No Low Level Format: Not supported Device Location: "Lower"</pre>	<pre>MacBook-Pro:/ oompa\$ diskutil info /dev/disk0 Device Identifier: disk0 Device Node: /dev/disk0 Whole: Yes Part of Whole: disk0 Device / Media Name: APPLE SSD AP1024J Volume Name: Not applicable (no file system) Mounted: Not applicable (no file system) File System: None Content (IOContent): GUID_partition_scheme OS Can Be Installed: No Media Type: Generic Protocol: PCI-Express SMART Status: Not Supported Disk Size: 1.0 TB (1000555581440 Bytes) (exactly 1954210120 512-Byte-Units) Device Block Size: 4096 Bytes Read-Only Media: No Read-Only Volume: Not applicable (no file system) Device Location: Internal Removable Media: Fixed Solid State: Yes OS 9 Drivers: No Low Level Format: Not supported</pre>
--	---

The `diskutil info` command is used on a specific disk or partition to display additional information about the disk or partition.

In the screenshot above, `disk0` was used. The output shows various information such as:

- Make/Model of the hardware (Toshiba MK5065GSXF).
- The drive interface (SATA).
- Drive size in bytes and blocks.
- Block Size: Be aware that newer Mac hardware will likely use 4096-byte sectors instead of 512-byte sectors.
- Read-Only.
- Location: Useful for systems with multiple internal disks such as Mac Pro systems.


```

byte:~ oompa$ diskutil info disk0
Device Identifier:      disk0
Device Node:           /dev/disk0
Part of Whole:         disk0
Device / Media Name:   TOSHIBA MK5065GSXF Media

Volume Name:           Not applicable (no file system)

Mounted:               Not applicable (no file system)

File System:           None

Content (IOContent):   GUID_partition_scheme
OS Can Be Installed:   No
Media Type:            Generic
Protocol:              SATA
SMART Status:          Verified

Total Size:            500.1 GB (500107862016 Bytes) (exactly 976773168
512-Byte-Blocks)
Volume Free Space:     Not applicable (no file system)
Device Block Size:     512 Bytes

Read-Only Media:       No
Read-Only Volume:     Not applicable (no file system)
Ejectable:             No

Whole:                 Yes
Internal:              Yes
Solid State:           No
OS 9 Drivers:         No
Low Level Format:      Not supported
Device Location:       "Lower"

```

```

[MacBook-Pro:/ oompa$ diskutil info /dev/disk0
Device Identifier:      disk0
Device Node:           /dev/disk0
Whole:                 Yes
Part of Whole:         disk0
Device / Media Name:   APPLE SSD AP1024J

Volume Name:           Not applicable (no file system)
Mounted:               Not applicable (no file system)
File System:           None

Content (IOContent):   GUID_partition_scheme
OS Can Be Installed:   No
Media Type:            Generic
Protocol:              PCI-Express
SMART Status:          Not Supported

Disk Size:             1.0 TB (1000555581440 Bytes) (exactly 1954210120
512-Byte-Units)
Device Block Size:     4096 Bytes

Read-Only Media:       No
Read-Only Volume:     Not applicable (no file system)

Device Location:       Internal
Removable Media:       Fixed

Solid State:           Yes
OS 9 Drivers:         No
Low Level Format:      Not supported

```

```

byte:~ oompa$ diskutil info disk0s2
Device Identifier:      disk0s2
Device Node:           /dev/disk0s2
Part of Whole:         disk0
Device / Media Name:   Customer

Volume Name:           Macintosh HD
Escaped with Unicode:  Macintosh%FF%FE%20%00HD

Mounted:               Yes
Mount Point:           /
Escaped with Unicode:  /

File System Personality:  Journaled HFS+
Type (Bundle):           hfs
Name (User Visible):     Mac OS Extended (Journaled)
Journal:                 Journal size 40960 KB at offset 0xe38a000
Owners:                  Enabled

Partition Type:         Apple_HFS
OS Can Be Installed:    Yes
Media Type:             Generic
Protocol:               SATA
SMART Status:           Verified
Volume UUID:            C51CD139-A54F-3988-A787-213C0CBA6D71

Total Size:             499.2 GB (499248103424 Bytes) (exactly 975093952
512-Byte-Blocks)
Volume Free Space:      272.1 GB (272126107648 Bytes) (exactly 531496304
512-Byte-Blocks)
Device Block Size:     512 Bytes

Read-Only Media:       No
Read-Only Volume:     No
Ejectable:             No

Whole:                 No
Internal:              Yes
Solid State:          No
Device Location:      "Lower"

```

```

Sarahs-MBP:/ oompa$ diskutil info /dev/disk1s1
Device Identifier:      disk1s1
Device Node:           /dev/disk1s1
Whole:                 No
Part of Whole:         disk1

Volume Name:           HighSierra
Mounted:               Yes
Mount Point:           /

Partition Type:         41504653-0000-11AA-AA11-00306543ECAC
File System Personality:  APFS
Type (Bundle):         apfs
Name (User Visible):   APFS
Owners:                Enabled

OS Can Be Installed:    Yes
Booter Disk:           disk1s2
Recovery Disk:         disk1s3
Media Type:            Generic
Protocol:              PCI-Express
SMART Status:          Not Supported
Volume UUID:           1D19162C-518C-3A34-A02C-2D428A4BC44E
Disk / Partition UUID: 1D19162C-518C-3A34-A02C-2D428A4BC44E

Disk Size:             1.0 TB (1000240963584 Bytes) (exactly 1953595632
512-Byte-Units)
Device Block Size:     4096 Bytes

Volume Total Space:    1.0 TB (1000240963584 Bytes) (exactly 1953595632
512-Byte-Units)
Volume Used Space:     314.6 GB (314610802688 Bytes) (exactly 614474224
512-Byte-Units) (31.5%)
Volume Available Space: 685.6 GB (685630160896 Bytes) (exactly 1339121408
512-Byte-Units) (68.5%)
Allocation Block Size: 4096 Bytes

Read-Only Media:       No
Read-Only Volume:     No

Device Location:      Internal
Removable Media:      Fixed
Solid State:          Yes

```

Another example of the `diskutil info` command is used on a specific partition or volume (`/dev/disk0s2` and `/dev/disk1s1`). The example on the left is an HFS+ OS volume, while the example on the right is an APFS OS volume.

This will show additional information about the volume:

- Volume Name (Macintosh HD/HighSierra)
- File System (HFS+, Mac OS Extended /APFS)
- Disk Protocol (SATA/PCI-Express)
- Volume Universal Unique Identifier
- Volume Size and Free Space
- Read-Only status
- If the volume is “ejectable” or removable
- Internal or External Volume
- Solid State status
- Etc.

```

byte:~ oompa$ diskutil info disk0s2
Device Identifier:      disk0s2
Device Node:           /dev/disk0s2
Part of Whole:         disk0
Device / Media Name:   Customer

Volume Name:           Macintosh HD
Escaped with Unicode:  Macintosh%FF%FE%20%00HD

Mounted:               Yes
Mount Point:           /
Escaped with Unicode:  /

File System Personality:  Journaled HFS+
Type (Bundle):           hfs
Name (User Visible):     Mac OS Extended (Journaled)
Journal:                 Journal size 40960 KB at offset 0xe38a000
Owners:                  Enabled

Partition Type:         Apple_HFS
OS Can Be Installed:    Yes
Media Type:              Generic
Protocol:                SATA
SMART Status:           Verified
Volume UUID:            C51CD139-A54F-3988-A787-213C0CBA6D71

Total Size:              499.2 GB (499248103424 Bytes) (exactly 975093952
512-Byte-Blocks)
Volume Free Space:      272.1 GB (272126107648 Bytes) (exactly 531496304
512-Byte-Blocks)
Device Block Size:      512 Bytes

Read-Only Media:        No
Read-Only Volume:      No
Ejectable:              No

Whole:                  No
Internal:               Yes
Solid State:            No
Device Location:        "Lower"

```

```

[Sarahs-MBP:/ oompa$ diskutil info /dev/disk1s1
Device Identifier:      disk1s1
Device Node:           /dev/disk1s1
Whole:                 No
Part of Whole:         disk1

Volume Name:           HighSierra
Mounted:               Yes
Mount Point:           /

Partition Type:        41504653-0000-11AA-AA11-00306543ECAC
File System Personality: APFS
Type (Bundle):         apfs
Name (User Visible):   APFS
Owners:                Enabled

OS Can Be Installed:   Yes
Booter Disk:           disk1s2
Recovery Disk:         disk1s3
Media Type:            Generic
Protocol:              PCI-Express
SMART Status:          Not Supported
Volume UUID:           1D19162C-518C-3A34-A02C-2D428A4BC44E
Disk / Partition UUID: 1D19162C-518C-3A34-A02C-2D428A4BC44E

Disk Size:              1.0 TB (1000240963584 Bytes) (exactly 1953595632
512-Byte-Units)
Device Block Size:     4096 Bytes

Volume Total Space:    1.0 TB (1000240963584 Bytes) (exactly 1953595632
512-Byte-Units)
Volume Used Space:     314.6 GB (314610802688 Bytes) (exactly 614474224
512-Byte-Units) (31.5%)
Volume Available Space: 685.6 GB (685630160896 Bytes) (exactly 1339121408
512-Byte-Units) (68.5%)
Allocation Block Size: 4096 Bytes

Read-Only Media:       No
Read-Only Volume:     No

Device Location:       Internal
Removable Media:       Fixed

Solid State:           Yes

```

GUID Partition Table on macOS

Booting from a GPT Partitioned disk available on Intel-based Macs. (~2006+)

Mounting a GPT Partitioned disk available in 10.4+

Apple Partition Map (APM) limited to 2TB disks

Advantages over MBR

- 128 Primary Partitions, Unique IDs (GUID) for Disks/Volumes, Alternate Headers/Partition Tables w/Checksums

Little Endian (mostly)

Technical Note 2166: Secrets of the GPT

The GUID Partition Table implemented on Intel-based Mac systems started in 2006 with the transition to Intel hardware; however, mounting a GPT partitioned disk was available in Mac OS X 10.4.

The switch to GPT from APM was due to the limited disk size that APM could handle. The Master Boot Record also had this drive size limit.

There were also limits with the number of partitions. MBR is limited to four primary partitions, while GPT has room for 128 primary partitions.

Other advantages of GPT are the use of unique identification (GUID) for disks, partitions, and backups of the GPT header and partition table.

Technical Note 2166 provides information pertaining to GPT used on Mac systems.

References:

TN2166: Secrets of the GPT

https://developer.apple.com/library/mac/technotes/tn2166/_index.html

“Forensic Analysis of GPT Disks and GUID Partition Tables,” by Bruce J. Nikkel

<http://www.digitalforensics.ch/nikkel09.pdf>

GPT Disk Layout: 512- or 4096-Byte Sectors

Protective MBR [1 Sector (512 bytes)]

Primary GPT Header [1 Sector (512/4096 bytes)]

Primary GPT Table [128 entries x 128 bytes (16384 bytes)]

[Volume Data]

Secondary GPT Table [128 entries x 128 bytes (16384 bytes)]

Secondary GPT Header [1 Sector (512/4096 bytes)]

A GUID partition table is comprised of five parts:

- Protective Master Boot Record
- Primary GUID Partition Table Header
- Primary GUID Partition Table
- Secondary GUID Partition Table
- Secondary GUID Partition Table Header

The secondary GPT Header and tables are backups of their primary counterparts.

GPT Partition Type – APFS Volume

```
Saraha-Air:~ oompa$ diskutil info disk1
Device Identifier:      disk1
Device Node:           /dev/disk1
Whole:                 Yes
Part of Whole:         disk1
Device / Media Name:   APPLE SSD AP1536M

Volume Name:           Not applicable (no file system)
Mounted:               Not applicable (no file system)
File System:           None

Content (IOContent):   EF57347C-0000-11AA-AA11-00306543ECAC
OS Can Be Installed:   No
Media Type:            Generic
Protocol:              PCI-Express
SMART Status:          Verified
Disk / Partition UUID: 510512DF-91CB-440C-B969-318B489CD3C8

Disk Size:             1.5 TB (1500518756352 Bytes) (exactly 2930700696 512-Byte-Units)
Device Block Size:     4096 Bytes

Read-Only Media:       No
Read-Only Volume:     Not applicable (no file system)

Device Location:       Internal
Removable Media:       Fixed

Solid State:           Yes
Virtual:                Yes
Hardware AES Support:  Yes
```

This screenshot shows the `diskutil info` output for the APFS Container. This provides us the GPT Content type GUID.

The screenshot below shows the output of the command “`sudo gpt -r show`” and “`sudo gpt -r show -l`”. This command is used to print the GUID partition table information for the specified disk. The “`-l`” option can be used to toggle between the display of the partition label and the GUID number. (Newer macOSs may not show the labels on new clean installs as shown below). The `-r` argument is used to show the GPT information in a read-only format. The output of the command contains the starting sector, the size of the partition (in sectors), and the contents. The PMBR is the Protective Master Boot Record. Note the primary and secondary GPT header and tables. Each partition type has an associated GUID that will be the same across all partitions of the same type.

With 10.13, SIP must now be disabled to run this command as shown below. If the system is using the APFS file system, it will show some different GPT GUIDs than with HFS+ volumes.

The GUID for Apple_APFS Partition or “APFS Container”: `7C3457EF-0000-11AA-AA11-00306543ECAC`, although it is listed as `EF57347C-0000-11AA-AA11-00306543ECAC` in the `diskutil info` output above.

```
MacBook-Pro:/ oompa$ sudo gpt -r show /dev/disk0
start      size  index  contents
0          1      1      PMBR
1          1      1      Pri GPT header
2          4      4      Pri GPT table
6          76800  1      GPT part - C12A7328-F81F-11D2-BA4B-00A0C93EC93B
76806      244199454  2      GPT part - 7C3457EF-0000-11AA-AA11-00306543ECAC
244276260  4      4      Sec GPT table
244276264  1      1      Sec GPT header
MacBook-Pro:/ oompa$ sudo gpt -r show -l /dev/disk0
start      size  index  contents
0          1      1      PMBR
1          1      1      Pri GPT header
2          4      4      Pri GPT table
6          76800  1      GPT part - "EFI System Partition"
76806      244199454  2      GPT part - ""
244276260  4      4      Sec GPT table
244276264  1      1      Sec GPT header
```

Common GPT Partition Type GUIDs

Partition Type GUID: Unique to Partition Type
Volume GUID: Unique across All Volumes Globally

Type	GUID
EFI System Partition	C12A7328-F81F-11D2-BA4B-00A0C93EC93B
HFS+ Partition	<u>48465300-0000-11AA-AA11-00306543ECAC</u> H F S
Apple Boot Partition	<u>426F6F74-0000-11AA-AA11-00306543ECAC</u> B o o t
Apple Core Storage (FileVault)	<u>53746F72-6167-11AA-AA11-00306543ECAC</u> S t o r a g
APFS Partition	7C3457EF-0000-11AA-AA11-00306543ECAC
Basic Data Partition (Boot Camp)	EBD0A0A2-B9E5-4433-87C0-68B6B72699C7

Similar to the MBR, each partition type has its own identifier. GPT uses GUIDs. In the GPT Table examples shown later, there are three partitions:

- EFI System Partition: C12A7328-F81F-11D2-BA4B-00A0C93EC93B (EFI System Partition)
- Untitled: 48465300-0000-11AA-AA11-00306543ECAC (HFS+ Partition)
- Recovery HD: 426F6F74-0000-11AA-AA11-00306543ECAC (Apple Boot Partition)

The Apple-created GUIDs contain “hidden” hints.

- HFS+ Partition GUIDs contain the string “HFS”
- Apple Boot Partition GUIDs contain the string “Boot”
- Apple Core Storage Partition GUIDs contain the string “Storag”

More GUID partition types can be found on the Wikipedia article for GUID Partition Table (http://en.wikipedia.org/wiki/GUID_Partition_Table).

GPT GUID Format

On-Disk Format:

726F7453-6761-AA11-AA11-00306543ECAC

Converted Format:

53746F72-6167-11AA-AA11-00306543ECAC

53746F72	6167	11AA	AA11	00306543ECAC
----------	------	------	------	--------------

Little Endian

Big Endian

GPT GUIDs use a combination of little and big endian.

The converted format uses little endian (by byte) for the first three sections of the GUID, while using big endian for the last two.

Show GPT with The Sleuth Kit (TSK) “mmls <image>”

```
[Sarahs-Air:Mac oompa$ mmls galaga.E01
GUID Partition Table (EFI)
Offset Sector: 0
Units are in 512-byte sectors
```

	Slot	Start	End	Length	Description
000:	Meta	0000000000	0000000000	0000000001	Safety Table
001:	-----	0000000000	0000000039	0000000040	Unallocated
002:	Meta	0000000001	0000000001	0000000001	GPT Header
003:	Meta	0000000002	0000000033	0000000032	Partition Table
004:	000	0000000040	0000409639	0000409600	EFI System Partition
005:	-----	0000409640	0000411647	0000002008	Unallocated
006:	001	0000411648	0062533255	0062121608	FILES
007:	-----	0062533256	0062533295	0000000040	Unallocated

The Sleuth Kit (TSK) can be used to review the GPT partition scheme using the `mmls` command. TSK is a free, open-source tool used to interact with disks or disk images to provide forensic analysis via the command line.

In the screenshot above, `mmls` is directed at an APFS EWF image. The Sleuth Kit calls the Protective MBR the Safety Table.

The Sleuth Kit can be downloaded from <http://sleuthkit.org/>. TSK man pages are located here: <http://www.sleuthkit.org/sleuthkit/man/>.

Protective MBR (Safety Table)

LBA 0

Backward Compatibility

Same Format

- 512 Bytes
- Partition Table: Offset 446
- 4 16-byte Partition Entries
- One Partition Entry: Entire Disk
 - Partition Type:
 - 0xEE (EFI GPT Disk)
 - 0xAF: Apple Mac OS X HFS and HFS+
- Three Partition Entries: Zeros

000	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
022	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
044	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
066	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
088	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
110	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
132	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
154	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
176	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
198	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
220	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
242	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
264	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
286	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
308	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
330	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
352	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
374	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
396	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
418	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
440	00 00 00 00 00 00 FE FF EE FE FF FF 01 00 00 AF 4B F9 00k
462	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
484	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
506	00 00 00 00 55 AAU

The Protective Master Boot Record, or Safety Table as it may be called, is used for backward compatibility. If the software does not recognize the GPT format, the Protective MBR may be used to safeguard the GPT data.

This MBR uses the same format as any other MBR, with the exception that it only has one partition with the partition type 0xEE for EFI GPT Disk. The remaining three primary partitions are left blank.

Typical Mac-related Partition Types:

- 0xAF : Apple Mac OS X HFS and HFS+
- 0xEE : EFI GPT Disk (EFI Protective MBR)

The partition type outlined in the box, 0xEE, is associated with an EFI GPT Disk partition. The size of the partition is 234,441,647 (0xAF4BF90D) in sectors. Note the 0x55AA signature at the bottom. For more information about the Master Boot Record, please review the book *File System Forensic Analysis*, by Brian Carrier.

Note: GPT data uses little endian

CHS = Cylinder-Head-Sectors

Reference:

Brian Carrier, *File System Forensic Analysis*, p.64, Table 5.2

GPT Header Example

000	45 46 49 20	50 41 52 54	00 00 01 00	5C 00 00 00	1D B2 06 32	00 00 00 00	EFI PART.....\.....2....
024	01 00 00 00	00 00 00 00	AF 48 F9 0D	00 00 00 00	22 00 00 00	00 00 00 00K....."
048	8E 4B F9 0D	00 00 00 00	4D 5B EB 71	99 57 24 42	9E 64 3A 5F	11 A5 D0 50	.K.....M[q.w\$B.d:..P
072	02 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	A6 EA 73 5C	00 00 00 00s\....
096	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
120	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
144	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
168	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
192	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
216	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
240	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
264	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
288	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
312	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
336	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
360	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
384	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
408	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
432	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
456	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
480	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
504	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00

The GPT Header is 512 bytes and contains a human-recognizable signature, “EFI PART”. Items of interest include:

- At offset 24, the LBA (Little Endian) of the GPT Header: 1 (0x0100000000000000)
- At offset 32, the LBA (Little Endian) of the alternate GPT header: 234441647 (0xAF4BF90D)
- At offset 56, the unique GUID of this disk:
 - 71EB5B4D-5799-4224-9E64-3A5F11A5D050
- At offset 72, the LBA (Little Endian) of the GUID Partition Table: 2 (0x0200000000000000)

Note: GPT data uses little endian

- GUID Endian-ness: The first three sections are in little endian, while the last two are not.

References:

TN2166: Secrets of the GPT
https://developer.apple.com/library/mac/technotes/tn2166/_index.html
http://www.uefi.org/sites/default/files/resources/2_4_Errata_B.pdf (Section 5.3.2)

Offset	Size (bytes)	Field
0	8	Signature (EFI PART)
8	4	Revision (1.0)
12	4	Size of Header (bytes)
16	4	Header CRC32
20	4	Reserved
24	8	LBA of GPT Header
32	8	LBA of Backup GPT Header
40	8	First Usable LBA
48	8	Last Usable LBA
56	16	Disk GUID
72	8	Starting LBA of GUID Partition Table
80	4	Number of Partition Entries Available
84	4	Size of Partition Entry
88	4	Partition Entry Array CRC32
92	Rest	Reserved

GPT Table Example

00000	28 73 2A C1	1F F8 D2 11	BA 4B 00 A0	C9 3E C9 3B	(s*.....K...>;
00016	CF A2 13 DE	62 C7 B2 4D	AE 50 8B 63	C7 7C 4C 36	...b..M.P.c. L6
00032	28 00 00 00	00 00 00 00	27 40 06 00	00 00 00 00	(.....'@.....
00048	00 00 00 00	00 00 00 00	45 00 46 00	49 00 20 00E.F.I..
00064	53 00 79 00	73 00 74 00	65 00 6D 00	20 00 50 00	S.y.s.t.e.m. .P.
00080	61 00 72 00	74 00 69 00	74 00 69 00	6F 00 6E 00	a.r.t.i.t.i.o.n.
00096	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00112	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00128	00 53 46 48	00 00 AA 11	AA 11 00 30	65 43 EC AC	.SFH.....0eC..
00144	E5 40 CD 83	CE 72 54 4C	AE 90 2C 0D	60 9F 44 64	.@...rTL...` Dd
00160	28 40 06 00	00 00 00 00	67 EC E5 0D	00 00 00 00	(@.....g.....
00176	00 00 00 00	00 00 00 00	55 00 6E 00	74 00 69 00U.n.t.i.
00192	74 00 6C 00	65 00 64 00	00 00 00 00	00 00 00 00	t.l.e.d.....
00208	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00224	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00240	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00256	74 6F 6F 42	00 00 AA 11	AA 11 00 30	65 43 EC AC	tooB.....0eC..
00272	28 C7 C9 0C	03 86 9E 4E	A5 D7 21 6E	97 90 53 B9	(.....N...In..S.
00288	68 EC E5 0D	00 00 00 00	87 4B F9 0D	00 00 00 00	h.....K.....
00304	00 00 00 00	00 00 00 00	52 00 65 00	63 00 6F 00R.e.c.o.
00320	76 00 65 00	72 00 79 00	20 00 48 00	44 00 00 00	v.e.r.y. .H.D...
00336	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00352	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00368	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00

The GPT Table is made up of an array of partition table entries (128 entries of 128 bytes each) with the format shown in the table here. In the screenshot above, a line splits each partition entry. The **middle** partition table will be analyzed. This partition entry contains the following items:

At offset 128, the **partition type** GUID is:
48465300-0000-11AA-AA11-00306543ECAC
(0x005346480000AA11AA1100306543ECAC) .

This is a unique GUID to this **type** of partition.

At offset 144, the **unique partition** GUID is:
83CD40E5-72CE-4C54-AE90-2C0D609F4464
(0xE540CD83CE72544CAE902C0D609F4464) .

This is a unique GUID only to **this** partition.

At offset 160, the starting LBA is 409640
(0x2840060000000000) .

At offset 168, the ending LBA is 233172071
(0x67ECE50D00000000) .

At offset 184, the partition name (in Unicode) is "Untitled".

Offset	Size (bytes)	Field
0	16	Partition Type GUID
16	16	Unique Partition GUID
32	8	Starting LBA (Little Endian)
40	8	Ending LBA (Little Endian)
48	8	Attributes
56	72	Partition Name
128	Rest	Reserved

Note: GPT data uses little endian

- GUID Endian-ness: The first three sections are in little endian, while the last two are not.

References:

TN2166: Secrets of the GPT
https://developer.apple.com/library/mac/technotes/tn2166/_index.html
http://www.uefi.org/sites/default/files/resources/2_4_Errata_B.pdf
 (Section 5.3.3)

00000	28 73 2A C1	1F F8 D2 11	BA 4B 00 A0	C9 3E C9 3B	(s*.....K...>.;
00016	CF A2 13 DE	62 C7 B2 4D	AE 50 8B 63	C7 7C 4C 36b..M.P.c. L6
00032	28 00 00 00	00 00 00 00	27 40 06 00	00 00 00 00	(.....'@.....
00048	00 00 00 00	00 00 00 00	45 00 46 00	49 00 20 00E.F.I. .
00064	53 00 79 00	73 00 74 00	65 00 6D 00	20 00 50 00	S.y.s.t.e.m. .P.
00080	61 00 72 00	74 00 69 00	74 00 69 00	6F 00 6E 00	a.r.t.i.t.i.o.n.
00096	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00112	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00128	00 53 46 48	00 00 AA 11	AA 11 00 30	65 43 EC AC	.SFH.....0eC..
00144	E5 40 CD 83	CE 72 54 4C	AE 90 2C 0D	60 9F 44 64	.@...rTL...'.Dd
00160	28 40 06 00	00 00 00 00	67 EC E5 0D	00 00 00 00	(@.....g.....
00176	00 00 00 00	00 00 00 00	55 00 6E 00	74 00 69 00U.n.t.i.
00192	74 00 6C 00	65 00 64 00	00 00 00 00	00 00 00 00	t.l.e.d.....
00208	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00224	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00240	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00256	74 6F 6F 42	00 00 AA 11	AA 11 00 30	65 43 EC AC	tooB.....0eC..
00272	28 C7 C9 0C	03 86 9E 4E	A5 D7 21 6E	97 90 53 B9	(.....N..!n..S.
00288	68 EC E5 0D	00 00 00 00	87 4B F9 0D	00 00 00 00	h.....K.....
00304	00 00 00 00	00 00 00 00	52 00 65 00	63 00 6F 00R.e.c.o.
00320	76 00 65 00	72 00 79 00	20 00 48 00	44 00 00 00	v.e.r.y. .H.D...
00336	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00352	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00
00368	00 00 00 00	00 00 00 00	00 00 00 00	00 00 00 00

FileVault 2

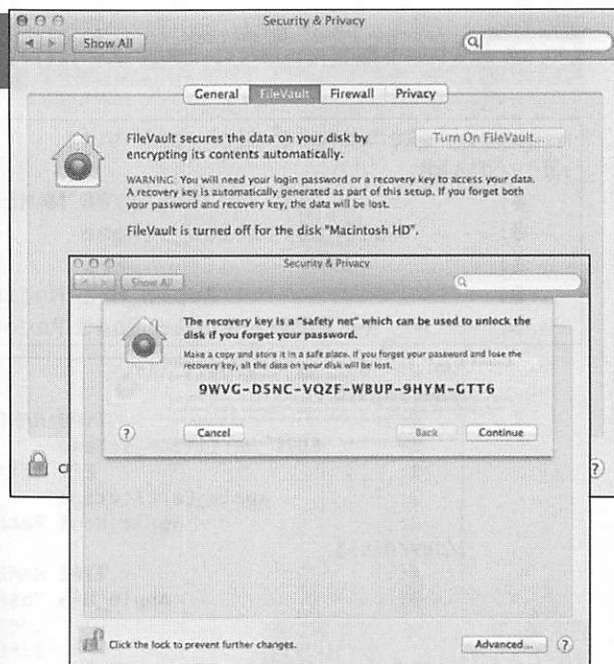
Full Disk Encryption

10.7+

Additional Recovery HD Partition

HFS+ and APFS

Legacy FileVault only encrypted user's home directory



FileVault 2 was introduced in Mac OS X Lion and encrypts the whole disk, except for EFI and Recovery partitions.

FileVault 2 uses a similar setup screen, as shown above. Before the encryption process takes place, the setup will display a recovery key that the user can choose to store with Apple. This recovery key can be used to recover a FileVault 2 encrypted volume.

There are various enterprise-level escrow key utilities available.

- Cauliflowervest: <https://github.com/google/cauliflowervest>
- Crypt: <https://github.com/grahamgilbert/Crypt>
- JAMF Casper Suite: <http://www.jamfsoftware.com/>

A great blog that discusses enterprise-level OS X networks and FileVault 2 specifically is Der Flounder: <https://derflounder.wordpress.com/>.

FileVault (now called Legacy FileVault) was introduced in 10.3. FileVault, if implemented, encrypts the home directory of a user. All other user directories and system files remain unencrypted. The encrypted home directory is stored in a sparse disk image or a sparse bundle.

GPT HFS+ FileVault 2 Example: "diskutil list"

```
nibble:~ sledwards$ diskutil list  
/dev/disk0
```

No FileVault

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	GUID_partition_scheme		*121.3 GB	disk0
1:	EFI		209.7 MB	disk0s1
2:	Apple_HFS	Macintosh HD	120.5 GB	disk0s2
3:	Apple_Boot	Recovery HD	650.0 MB	disk0s3

```
word:~ oompa$ diskutil list  
/dev/disk0
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	GUID_partition_scheme		*500.3 GB	disk0
1:	EFI	EFI	209.7 MB	disk0s1
2:	Apple_CoreStorage		499.4 GB	disk0s2
3:	Apple_Boot	Recovery HD	650.0 MB	disk0s3

```
/dev/disk1
```

#:	TYPE	NAME	SIZE	IDENTIFIER
0:	Apple_HFS	Yosemite	*499.1 GB	disk1
		Logical Volume on disk0s2		
		33F6B43B-7A75-4B04-A04D-0AC6A4321BF5		
		Unlocked Encrypted		

FileVault Enabled

SANS|DFIR

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 110

In the top example, the screenshot shows the output of the `diskutil list` command on a disk that does not have FileVault 2 enabled.

The bottom example shows the same disk with FileVault 2 enabled. Note the addition of the `Apple_CoreStorage` partition and how "Macintosh HD" is now on `/dev/disk1`. The Apple CoreStorage partition is the encrypted partition of the FileVault volume on `/dev/disk1`.

CoreStorage: “diskutil cs list”

Logical Volume Group (LVG)

Physical Volume (PV)

Logical Volume Family (LVF)

Logical Volume (LV)

CoreStorage (unencrypted) on by default on NEW installs of 10.10

```
nibble:/sledwards$ diskutil cs list
CoreStorage logical volume groups (1 found)
|
+-- Logical Volume Group 068CB516-9C26-493A-9967-FC7977FE6855
|-----|
| Name:      Macintosh HD
| Status:    Online
| Size:      499418034176 B (499.4 GB)
| Free Space: 16777216 B (16.8 MB)
|-----|
+--< Physical Volume CAD328AC-A06F-466C-9145-5A312C782C83
|-----|
| Index:     0
| Disk:      disk0s2
| Status:    Online
| Size:      499418034176 B (499.4 GB)
|-----|
+--> Logical Volume Family E4A6F218-0490-424C-AF11-4C208E052F5B
|-----|
| Encryption Status:  Unlocked
| Encryption Type:    AES-XTS
| Conversion Status:  Complete
| Conversion Direction: -none-
| Has Encrypted Extents: Yes
| Fully Secure:       Yes
| Passphrase Required: Yes
|-----|
+--> Logical Volume D0650536-C674-48DD-BD9E-9EEB8FE857EE
|-----|
| Disk:      disk1
| Status:    Online
| Size (Total): 499082485760 B (499.1 GB)
| Conversion Progress: -none-
| Revertible: Yes (unlock and decryption required)
| LV Name:    Macintosh HD
| Volume Name: Macintosh HD
| Content Hint: Apple_HFS
```

SANS|DFIR

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 111

CoreStorage is Apple’s version of logical volume management—or “virtual volumes”—used with FileVault 2 full disk encryption and Fusion Drives.

Logical Volume Group (LVG): The top level, associated with a specific Physical Volume.

Physical Volume (PV): The physical disk or disk image. May contain more than one physical disk, as seen in Fusion Drives.

Logical Volume Family (LVF): Contains one or more Logical Volumes.

Logical Volume (LV): Where the file system is stored on a /dev/disk*.

As of 10.10, CoreStorage is now implemented by default on new installs of 10.10. CoreStorage does not necessarily mean the volume is encrypted. New installs will have an unencrypted CoreStorage implementation; however, users can still encrypt it using FileVault. Upgraded installs will not force CoreStorage to be implemented. CoreStorage can still be used without FileVault 2.

The screenshot shows the output from the `diskutil cs list` command. This command lists all the CoreStorage Logical Volume Groups.

This system has only one LVG for a 499.4GB physical disk. Each part (LVG, PV, LVF, LV) has their own UUID.

We can see from the information found in the LVF section that this disk is encrypted with AES-XTS (FileVault 2).

We can see from the information found in the LV section that this disk is currently unlocked and could be decrypted if needed. The volume is named “Macintosh HD” and is 499.1GB in size.

Reference:
[diskutil Man Page](#)


```

nibble:/ sledwards$ diskutil cs list
CoreStorage logical volume groups (1 found)
|
+--- Logical Volume Group 068CB516-9C26-493A-9967-FC7977FE6855
-----
Name:          Macintosh HD
Status:        Online
Size:          499418034176 B (499.4 GB)
Free Space:    16777216 B (16.8 MB)
|
+--< Physical Volume CAD328AC-A06F-466C-9145-5A312C7B2C83
|-----
| Index:       0
| Disk:        disk0s2
| Status:      Online
| Size:        499418034176 B (499.4 GB)
|
+--> Logical Volume Family E4A6F218-0490-424C-AF11-4C208E052F5B
-----
Encryption Status:    Unlocked
Encryption Type:      AES-XTS
Conversion Status:    Complete
Conversion Direction: -none-
Has Encrypted Extents: Yes
Fully Secure:         Yes
Passphrase Required:  Yes
|
+--> Logical Volume DD650536-C674-4BDD-BD9E-9EEB8FE857EE
-----
Disk:          disk1
Status:        Online
Size (Total):  499082485760 B (499.1 GB)
Conversion Progress: -none-
Revertible:    Yes (unlock and decryption required)
LV Name:       Macintosh HD
Volume Name:   Macintosh HD
Content Hint:  Apple_HFS

```



```
[Sarahs-MBP:/ oompa$ diskutil ap list
APFS Container (1 found)
```

```
|
+-- Container disk1 7AB09481-C4B7-4A1D-B551-6ADAA5D8ED24
```

```
=====
APFS Container Reference:    disk1
Capacity Ceiling (Size):    1000240963584 B (1.0 TB)
Capacity In Use By Volumes: 315068219392 B (315.1 GB) (31.5% used)
Capacity Available:         685172744192 B (685.2 GB) (68.5% free)
```

```
|
+--< Physical Store disk0s2 FC80FCEE-E44F-4F25-BA57-F303E87FB108
```

```
-----
APFS Physical Store Disk:   disk0s2
Size:                       1000240963584 B (1.0 TB)
```

```
|
+--> Volume disk1s1 1D19162C-518C-3A34-A02C-2D428A4BC44E
```

```
-----
APFS Volume Disk (Role):   disk1s1 (No specific role)
Name:                      HighSierra (Case-insensitive)
Mount Point:               /
Capacity Consumed:         311094956032 B (311.1 GB)
Encrypted:                 Yes (Unlocked)
```

```
|
+--> Volume disk1s2 670CBB6D-1FE7-446D-A76A-C549A159F34F
```

```
-----
APFS Volume Disk (Role):   disk1s2 (Preboot)
Name:                      Preboot (Case-insensitive)
Mount Point:               Not Mounted
Capacity Consumed:         23408640 B (23.4 MB)
Encrypted:                 No
```

```
|
+--> Volume disk1s3 30817573-A0CE-4CF7-AC2B-D7C7E921B424
```

```
-----
APFS Volume Disk (Role):   disk1s3 (Recovery)
Name:                      Recovery (Case-insensitive)
Mount Point:               Not Mounted
Capacity Consumed:         520765440 B (520.8 MB)
Encrypted:                 No
```

```
|
+--> Volume disk1s4 38B6B6EE-C683-4FF1-8100-EE6C1A551668
```

```
-----
APFS Volume Disk (Role):   disk1s4 (VM)
Name:                      VM (Case-insensitive)
Mount Point:               /private/var/vm
Capacity Consumed:         3221245952 B (3.2 GB)
Encrypted:                 No
```

Fusion Drive

Hybrid Drive

Hard Disk Drive + Solid State Drive

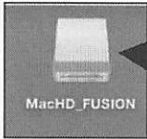
Size and Speed

Implements CoreStorage

- Combined to create a single volume

Most-accessed files located on SSD

Mac Mini and iMac Systems



```
word:~ oompa$ diskutil list
/dev/disk0
#:          TYPE NAME          SIZE          IDENTIFIER
0:      GUID_partition_scheme    *500.3 GB     disk0
1:          EFI EFI              209.7 MB     disk0s1
2:      Apple_CoreStorage         499.4 GB     disk0s2
3:      Apple_Boot Recovery HD    650.0 MB     disk0s3
/dev/disk1
#:          TYPE NAME          SIZE          IDENTIFIER
0:      Apple_HFS Yosemite        *499.1 GB     disk1
                Logical Volume on disk0s2
                33F6B43B-7A75-4B04-A04D-0AC6A4321BF5
                Unlocked Encrypted
/dev/disk4
#:          TYPE NAME          SIZE          IDENTIFIER
0:      GUID_partition_scheme    *31.0 GB     disk4
1:          EFI EFI              209.7 MB     disk4s1
2:      Apple_CoreStorage         30.7 GB     disk4s2
3:      Apple_Boot Boot OS X      134.2 MB     disk4s3
/dev/disk5
#:          TYPE NAME          SIZE          IDENTIFIER
0:      GUID_partition_scheme    *2.0 TB     disk5
1:          EFI EFI              209.7 MB     disk5s1
2:      Apple_HFS Time Machine Backups 2.0 TB     disk5s2
/dev/disk6
#:          TYPE NAME          SIZE          IDENTIFIER
0:      GUID_partition_scheme    *8.1 GB     disk6
1:          EFI EFI              209.7 MB     disk6s1
2:      Apple_CoreStorage         7.7 GB     disk6s2
3:      Apple_Boot Boot OS X      134.2 MB     disk6s3
/dev/disk7
#:          TYPE NAME          SIZE          IDENTIFIER
0:      Apple_HFS MacHD_FUSION    *33.2 GB     disk7
                Logical Volume on disk4s2, disk6s2
                B4E6C99F-20E7-41CC-9031-85C0F456B17F
                Unencrypted Fusion Drive
```

SANS|DFIR

FOR518.1 | Mac and iOS Forensic Analysis and Incident Response 115

The Fusion Drive comes with only a few specific systems, namely late model Mac Minis and iMacs.

These drives use a combination of a normal hard disk drive and a solid state drive to get the best of speed and drive size.

To create and present a single volume to the user, CoreStorage is implemented. The most-used files are located on the physical SSD hard drive to speed up file access.

In the example above, a Fusion Drive was created using two USB thumb drives.

- /dev/disk4: ~32GB
- /dev/disk6: ~8GB

/dev/disk7 (MacHD_FUSION) is the combination of the two drives acting as a single drive/partition. This is an unencrypted usage of CoreStorage.

Disk Image (.dmg)

Create with Disk Utility.app or hdiutil

Any Size

Formats:

- Mac OS Extended
- APFS
- FAT
- ExFAT (over 32GB)

Save As: Untitled
Tags:
Where: oompa
Name: Untitled
Size: 100 MB
Format: APFS
Encryption: none
Partitions: Single partition - GUID Partition Map
Image Format: read/write disk image
Cancel Save

May configure with encryption, partitions, and image formats

Sparse Disk Image/Bundle

SANS DFIR

Likely the most well-known volume is the DMG file, or Disk Image file. Mac systems use this format for everything from software installation to simple data archives. Disk Image files can be any size and any format that `hdiutil` or Disk Utility.app can implement: HFS+, APFS, FAT, or ExFAT. These volumes can mix and match various types of encryption, partitions, and image formats. Please refer to the options listed using the command “`hdiutil create -help`” or review the options in Disk Utility.app.

Legacy FileVault implements the Sparse Disk Image and the Sparse Bundle to store encrypted data. These file formats may also be used without the context of FileVault. The Disk Utility.app can be used to create encrypted and unencrypted volumes using these file types. The Sparse Disk Image is a single file with a `.sparseimage` file extension. It is one file that grows as data is added. The file has the signature “`sprs`”, as shown in the screenshot below.

```
0000000: 7370 7273 0000 0003 0000 0800 0000 0001 sprs.....  
0000010: 0001 3880 0000 0000 0000 0000 0000 0000 ..8.....  
0000020: 0001 3880 0000 0000 0000 0000 0000 0000 ..8.....  
0000030: 0000 0000 0000 0000 0000 0000 0000 0000 .....  
0000040: 0000 0001 0000 0002 0000 0014 0000 0026 .....&  
0000050: 0000 0027 0000 0028 0000 0005 0000 0003 ...'.....
```

The Sparse Bundle file is made up of multiple files (in a software bundle) with the file extension `.sparsebundle`.

```
byte:SECTION_1 oompa$ ls -laR sparse_bundle.sparsebundle/  
total 16  
drwxr-xr-x@ 6 oompa  staff  204 Aug 11 12:33 .  
drwxr-xr-x  17 oompa  staff  578 Aug 11 13:17 ..  
-rw-r--r--  1 oompa  staff  495 Aug 11 12:21 Info.bckup  
-rw-r--r--  1 oompa  staff  495 Aug 11 12:21 Info.plist  
drwxr-xr-x  5 oompa  staff  170 Aug 11 12:21 bands  
-rw-r--r--  1 oompa  staff   0 Aug 11 12:21 token  
  
sparse_bundle.sparsebundle//bands:  
total 30920  
drwxr-xr-x  5 oompa  staff    170 Aug 11 12:21 .  
drwxr-xr-x@ 6 oompa  staff    204 Aug 11 12:33 ..  
-rw-r--r--  1 oompa  staff 4689920 Aug 11 12:33 0  
-rw-r--r--  1 oompa  staff 3735552 Aug 11 12:21 2  
-rw-r--r--  1 oompa  staff 7405568 Aug 11 12:21 4
```

“hdiutil imageinfo *.dmg” or /dev/disk# (SIP Disabled)

• Rename .dd to .dmg

```
(word:Downloads oompa$ hdiutil imageinfo /dev/disk0
Backing Store Information:
  URL: file:///dev/rdisk0
  Name: rdisk0
  Class Name: CDevBackingStore
Class Name: CRawDiskImage
Checksum Type: none
Size Information:
  Total Bytes: 500277790720
  Compressed Ratio: 1
  Sector Count: 977105060
  Total Non-Empty Bytes: 500277790720
  Compressed Bytes: 500277790720
  Total Empty Bytes: 0
Format: RAW*
Format Description: raw read/write
Checksum Value:
Properties:
  Encrypted: false
  Kernel Compatible: false
  Checksummed: false
  Software License Agreement: false
  Partitioned: false
  Compressed: no
Segments:
  0: /dev/rdisk0
```

```
partitions:
  partition-scheme: GUID
  block-size: 512
  partitions:
    0:
      partition-name: Protective Master Boot Record
      partition-start: 0
      partition-synthesized: true
      partition-length: 1
      partition-hint: MBR
    1:
      partition-name: GPT Header
      partition-start: 1
      partition-synthesized: true
      partition-length: 1
      partition-hint: Primary GPT Header
    2:
      partition-name: GPT Partition Data
      partition-start: 2
      partition-synthesized: true
      partition-length: 32
      partition-hint: Primary GPT Table
    3:
      partition-name:
      partition-start: 34
      partition-synthesized: true
      partition-length: 6
      partition-hint: Apple_Free
    4:
      partition-UUID: 663BC686-4EAD-4834-9205-7DE18FD59776
      partition-name: EFI System Partition
      partition-hint-UUID: C12A7328-F81F-11D2-BA4B-00A0C93EC93B
      partition-start: 40
      partition-number: 1
      partition-length: 409600
      partition-hint: C12A7328-F81F-11D2-BA4B-00A0C93EC93B
      partition-fileystems:
        FAT32: EFI
```

The `hdiutil` command is used to interface with disk image files (*.dmg) or disk device (/dev/disk#).

This command must be run with a *.dmg file; if you have a raw DD file, you can add a .dmg extension.

The `imageinfo` parameter is used to display information about the disk image. (Due to the verbose output, the screenshot had to be shortened to fit on the slide.)

Information that can be gleaned from an image includes:

- Image Size (in bytes)
- Data about the compression—this image was not compressed
- Image Format—this example is labeled RAW Read/Write
 - A list of formats can be viewed in the man page for `hdiutil`
- Image location
- Detailed Partition Information
 - Partition Name
 - Start Block
 - Length (in blocks)
 - Partition Hint: Type of Partition
 - Partition Hint UUID (unique to type of partition)
 - Partition UUID (unique to partition)
 - File System (HFS+, FAT32, etc).
- Image file resizing information: Can be used to shrink a volume


```

0:      partition-name: Protective Master Boot Record
        partition-start: 0
        partition-synthesized: true
        partition-length: 1
        partition-hint: MBR
1:
        partition-name: GPT Header
        partition-start: 1
        partition-synthesized: true
        partition-length: 1
        partition-hint: Primary GPT Header
2:
        partition-name: GPT Partition Data
        partition-start: 2
        partition-synthesized: true
        partition-length: 32
        partition-hint: Primary GPT Table
3:
        partition-name:
        partition-start: 34
        partition-synthesized: true
        partition-length: 6
        partition-hint: Apple_Free
4:
        partition-UUID: 663BC686-4EAD-4B34-9205-7DE18FD59776
        partition-name: EFI System Partition
        partition-hint-UUID: C12A7328-F81F-11D2-BA4B-00A0C93EC93B
        partition-start: 40
        partition-number: 1
        partition-length: 409600
        partition-hint: C12A7328-F81F-11D2-BA4B-00A0C93EC93B
        partition-fileystems:
            FAT32: EFI
5:
        partition-UUID: 0E6B6127-93A3-4C49-98F2-45DD7C3D8C63
        partition-name: Untitled 2
        partition-hint-UUID: 53746F72-6167-11AA-AA11-00306543ECAC
        partition-start: 409640
        partition-number: 2
        partition-length: 975425848
        partition-hint: 53746F72-6167-11AA-AA11-00306543ECAC
6:
        partition-UUID: 484EAAB8-1F98-4CD5-B676-310996C5419C
        partition-name: Untitled 3
        partition-hint-UUID: 426F6F74-0000-11AA-AA11-00306543ECAC
        partition-start: 975835488
        partition-number: 3
        partition-length: 1269536
        partition-hint: 426F6F74-0000-11AA-AA11-00306543ECAC
        partition-fileystems:
            HFS+:
7:
        partition-name:
        partition-start: 977105024
        partition-synthesized: true
        partition-length: 3
        partition-hint: Apple_Free
8:
        partition-name: GPT Partition Data
        partition-start: 977105027
        partition-synthesized: true
        partition-length: 32
        partition-hint: Backup GPT Table
9:
        partition-name: GPT Header
        partition-start: 977105059
        partition-synthesized: true
        partition-length: 1
        partition-hint: Backup GPT Header

```

Lab 1.3

Disks and Partitions

This page intentionally left blank.

FOR518.1

Mac and iOS Forensic Analysis and Incident Response



FOR518 Section 1: Mac and iOS Essentials

© 2020 Sarah Edwards | All Rights Reserved | Version F01_01

Author: Sarah Edwards
oompa@csh.rit.edu
mac4n6.com
<http://twitter.com/iamevltwin>

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

“As usual, SANS courses pay for themselves by Day 2. By Day 3, you are itching to get back to the office to use what you've learned.”

Ken Evans, Hewlett Packard Enterprise - Digital Investigation Services

SANS Programs
sans.org/programs

GIAC Certifications
Graduate Degree Programs
NetWars & CyberCity Ranges
Cyber Guardian
Security Awareness Training
CyberTalent Management
Group/Enterprise Purchase Arrangements
DoDD 8140
Community of Interest for NetSec
Cybersecurity Innovation Awards



Search SANSInstitute

SANS Free Resources
sans.org/security-resources

- E-Newsletters
 - NewsBites*: Bi-weekly digest of top news
 - OUCH!*: Monthly security awareness newsletter
 - @RISK*: Weekly summary of threats & mitigations
- Internet Storm Center
- CIS Critical Security Controls
- Blogs
- Security Posters
- Webcasts
- InfoSec Reading Room
- Top 25 Software Errors
- Security Policies
- Intrusion Detection FAQ
- Tip of the Day
- 20 Coolest Careers
- Security Glossary

SANS Institute

8120 Woodmont Avenue | Suite 310
Bethesda, MD 20814
301.654.SANS(7267)
info@sans.org