585.1

Smartphone Overview, Fundamentals of Analysis, SQLite Introduction, Android Forensics Overview, and Android Backups

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

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

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

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

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

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

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

This page intentionally left blank.

FOR585
Dedications

FOR585 | Smartphone Forensic Analysis In-Depth

This course material is dedicated to:

“This work is dedicated to my co-author Lee, SANS peers, and students who drive me to stay ahead of the curve and uncover cool artifacts. To my patient family and friends, I love you so much. Jack and Natalie, you are my breath of fresh air at the end of really long days." —Heather Mahalik

“This work is dedicated to Heather for urging me to change career paths, the FOR585 author team for holding me to deadlines and keeping me on task, and my parents for teaching me to continue to ask questions and investigate those things that I do not understand.” —Domenica Lee Crognale

A special thanks to the brilliant researchers who contributed their knowledge and time to this course material.
We appreciate you working with us and continuing to research to give back to the community.

Jared Barnhart Mattia

Epifani Josh Hickman Ian Whiffin

Lab 0

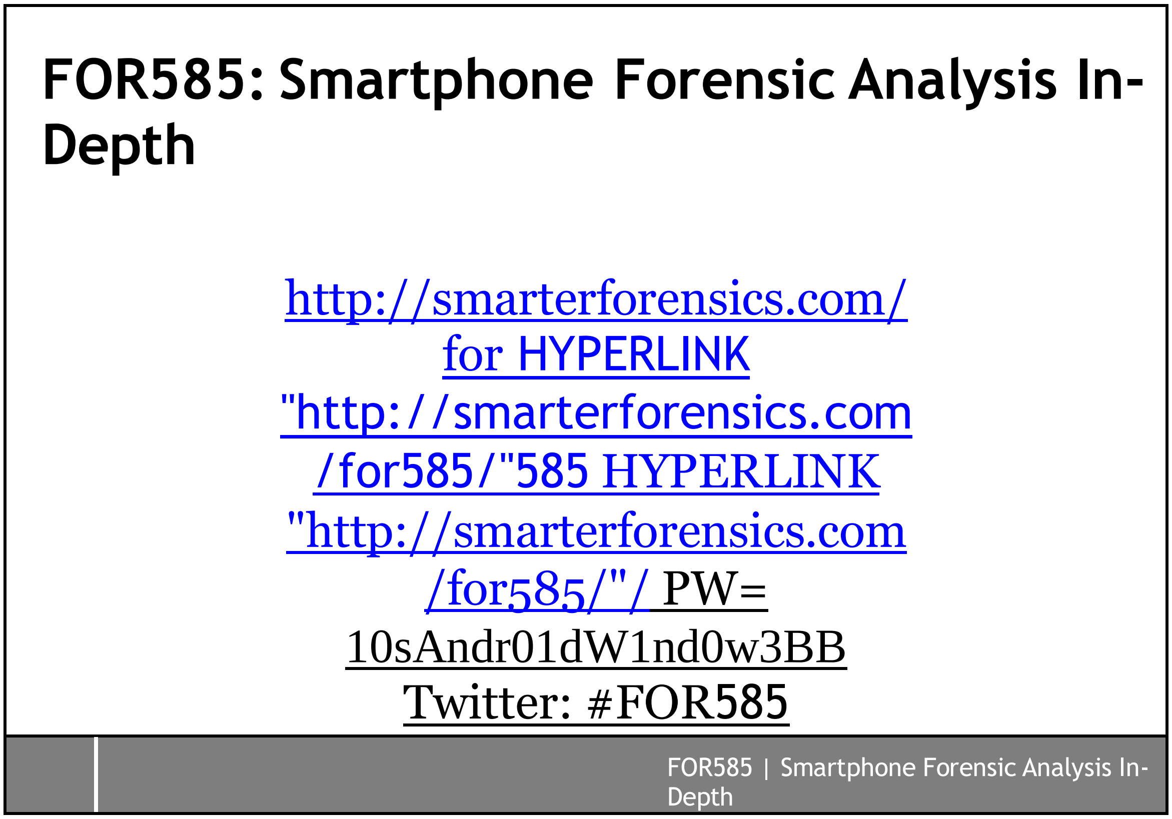
FOR585—SIFT Installation Extract and Prepare the SIFT VM. Follow the instructions exactly as they are presented in your workbook!

FOR585 | Smartphone Forensic Analysis In-Depth

The FOR585 SIFT installation lab takes some time to complete, as it's a large amount of data to extract. The steps must be exactly followed to prevent any issues when performing the labs in this course. Start this lab as soon as you have access to the course media files. Make sure to keep track of where you are in the lab and continue when the prior step completes. We recommend checking off each step in the workbook to ensure that you do not miss a critical step.

A Dropbox for this course has been created for you. If you are taking the class live or via Simulcast, your instructor will provide you with a link. If you are taking this class via OnDemand, your Dropbox access will match your course number. For example, if your book version is G01_01, then your Dropbox link will be https://for585.com/g01.

This page intentionally left blank.



FOR585: Smartphone Forensic Analysis In-Depth

http://smarterforensics.com/ for HYPERLINK "http://smarterforensics.com/for585/"585 HYPERLINK "http://smarterforensics.com/for585/"/ PW= 10sAndr01dW1nd0w3BB

Twitter: #FOR585

www.smarterforensics.com is maintained and updated by Heather Mahalik. She has created a password- protected section of the site to host updated materials that are shareable with FOR585 alumni. Expect to see updates in this portion of the website every time the authors complete an update with new cheat sheets, white papers, references, and possibly test data. (NOTE: The use of numbers for the lower-case vowels along with the "s" in windows for the password.)

Do not share this password. You paid to be here, and you should reap the benefits.

We have created an alumni Google group for this course for discussions pertaining to mobile devices (https://groups.google.com/forum/#!forum/for585-alumni). To join the group, please email HYPERLINK "mailto:585forensics@gmail.com"585forensics@gmail.com and state which class you took and the email address you wish to use to join. Please place your questions here, as they are beneficial to everyone: FOR585-Alumni.

Your course dropbox will have a lot of relevant information for you to include bonus material, cheat sheets, a sample GIAC index, the course forensic capstone and more! If you are taking this course in OnDemand, make sure to listen to the recording to know how to leverage and access your Dropbox.

This page intentionally left blank.

This page intentionally left blank.

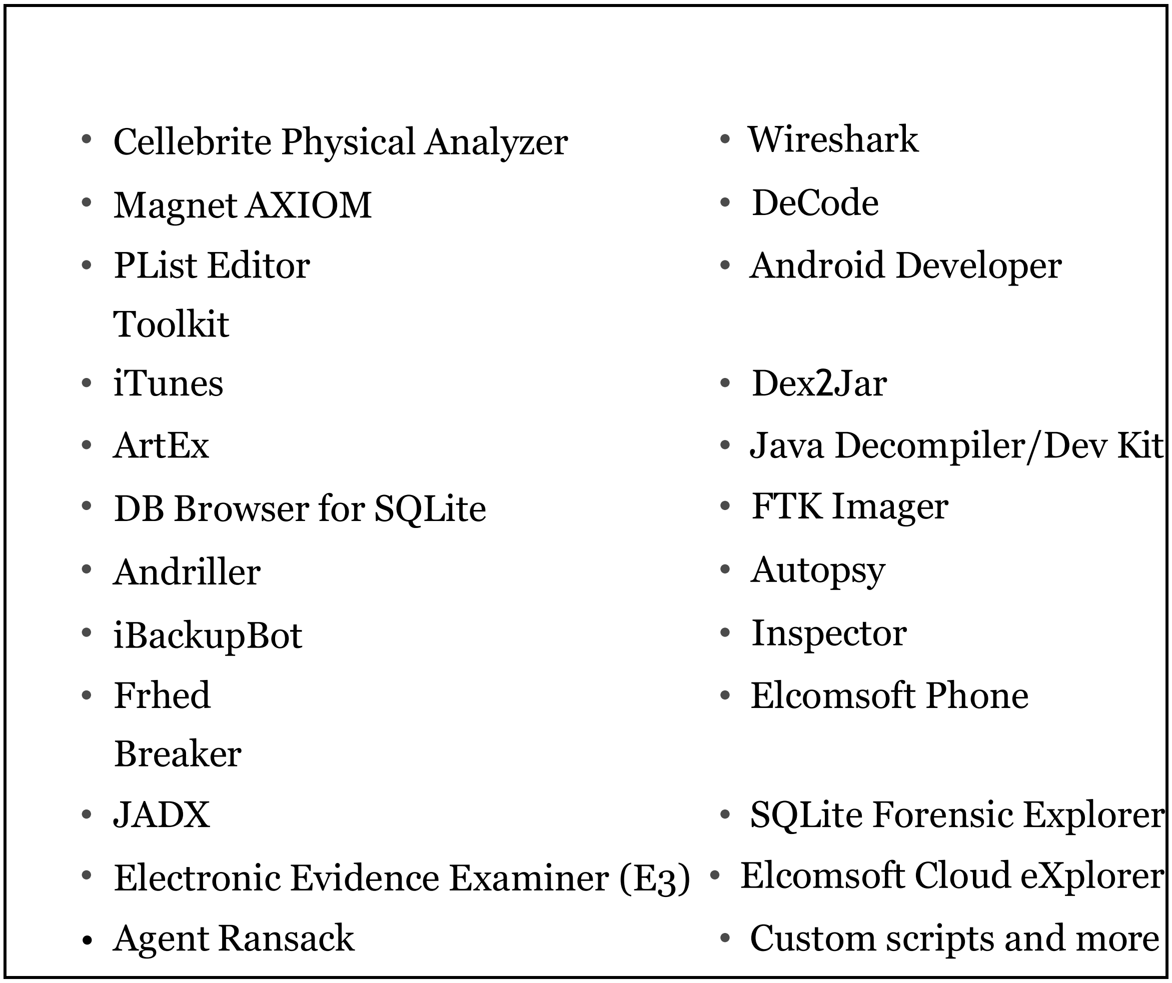
- Linux machines may require drivers for ExFAT

- Login: “**FOR585**”
- Password: “**forensics**”

SANS teaches the latest tools and techniques available. In addition, our aim is to create a laptop setup that is simple to use to accomplish the daily exercises utilizing forensic tools and scripts we deem most helpful in smartphone investigations. The media files include a preconfigured VMware Forensic Workstation, which is ready to tackle forensics as soon as your tools are licensed.

The tools may prompt you for updates, since updates are released far more frequently than we can create course media files. We know the versions in your FOR585 VM function as expected. If you choose to update, you may find that your lab results differ from the course workbook. These results may not be beneficial, as the tools sometimes lose capabilities once upgraded. On the flipside, the tools may parse additional information that was previously missed. The choice is yours, but we recommend you update any tool you wish after class as homework.

If you are having issues with Lab 0, refer to the end of the lab for troubleshooting tips. Alert your instructor, TA, or SANS Online SME immediately if your issue is not listed in the Lab 0 troubleshooting steps.



- Cellebrite Physical Analyzer
- Magnet AXIOM
- PList Editor
- iTunes
- ArtEx
- DB Browser for SQLite
- Andriller
- iBackupBot
- Frhed
- JADX
- Electronic Evidence Examiner (E3)
- Agent Ransack
- Wireshark
- DeCode
- Android Developer Toolkit
- Dex2Jar
- Java Decompiler/Dev Kit
- FTK Imager
- Autopsy
- Inspector
- Elcomsoft Phone Breaker
- SQLite Forensic Explorer
- Elcomsoft Cloud eXplorer
- Custom scripts and more

This page intentionally left blank.

- WARNING: It may not be a free tool!
- And most importantly when the tool isn't

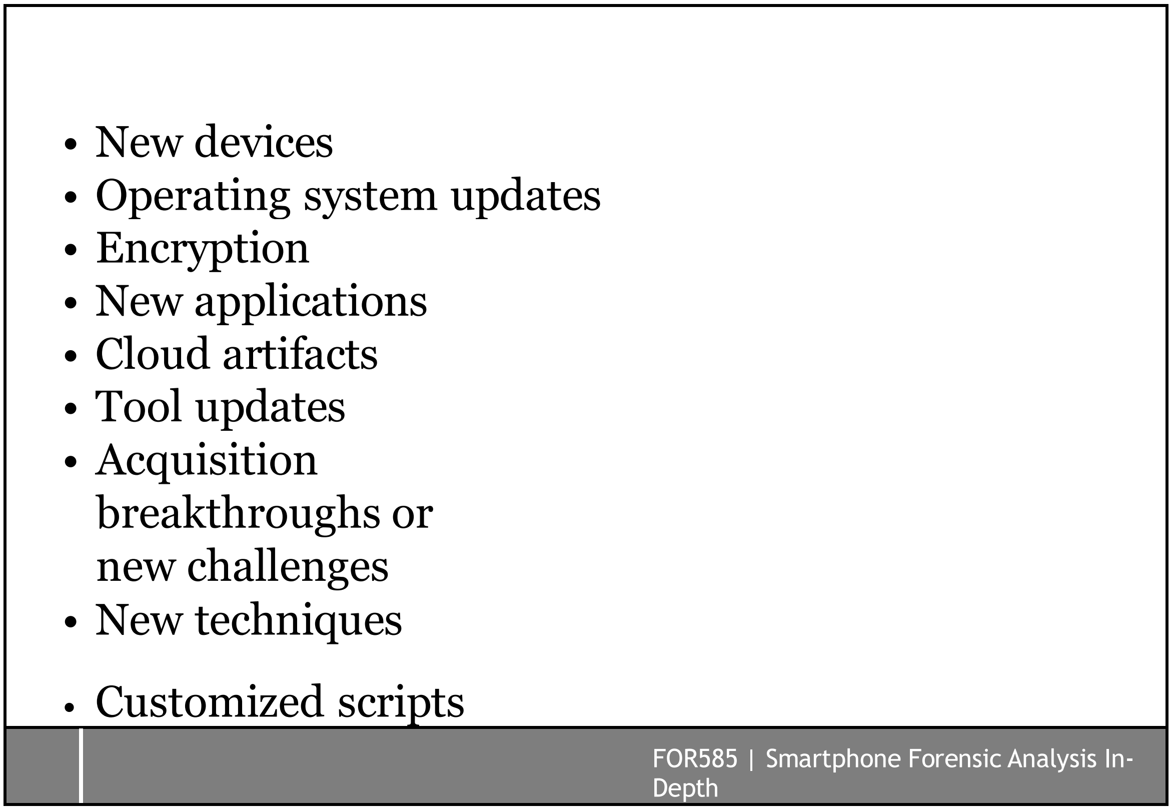
handling

the data correctly.

When it comes to mobile forensics, it is almost impossible to rely completely on open-source solutions. Often, for open-source tools to work, you need root access or a jailbreak to obtain file system information. In reality, this isn't always feasible. Because of this, we aim to teach you the best tools for both acquisition and analysis for each iOS and Android. We will cover both commercial and open-source solutions, as well as custom scripts and SQL queries that were developed for smartphone data.

This course will cover where the artifacts exist in the file system of iOS and Android devices and will include operating system updates. We will use our tools to get us to a certain point and then export the files of interest and use manual techniques to further our examination. We do not want you to become the examiner that uses the "find evidence" button, but we do want you to learn how far the tools can take you. Let's be honest— forensics is not cheap. Smartphone forensics is expensive, and the costs can quickly get out of hand when you are building your toolbox. For that reason, we want to highlight the tools that work most effectively for mobile devices of interest.

We encourage you to be creative and develop SQL queries and scripts that replicate what the commercial tools do. Please share those with us so we can include them in the alumni group or even in the course.



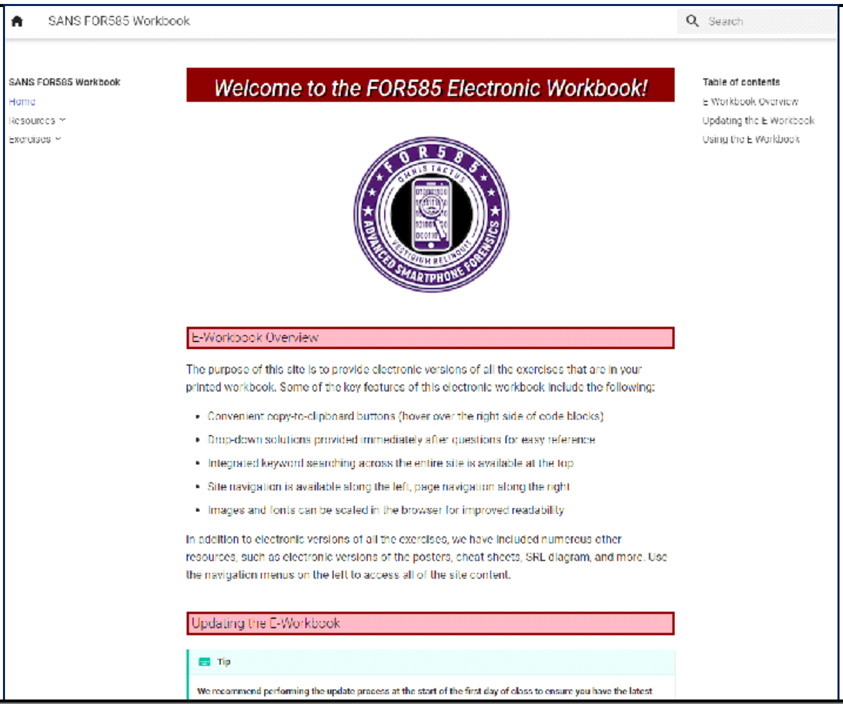
- New devices
- Operating system updates
- Encryption
- New applications
- Cloud artifacts
- Tool updates
- Acquisition breakthroughs or new challenges
- New techniques
- Customized scripts

This page intentionally left blank.

The labs in this course are designed to enhance your learning experience and to add a hands-on component to help you develop and hone your mobile forensics skill set. The course labs are not intended to be a test of what you learned in the lecture portion of class. They will go beyond the lecture and push you further as well as reinforce concepts touched upon in class.

The eWorkbook will have objectives, questions, answers, and walkthroughs. We have included one potential way to solve the problem—you may find other solutions. It's up to you whether you'd like to try the lab without instructions or work through the lab based on the instructions. Included in this course are 35 labs with data from over 32 devices (all unique datasets), plus SIM, SD, and cloud data. Be sure to check your course media files for bonus labs!

Make sure you download public images for more data to practice with. These are available via Magnet, Cellebrite and Josh Hickman. Josh blogs at https://thebinaryhick.blog where you will find his shared iOS and Android extractions.



Welcome to the FOR585 Electronic Workbook!

E-Workbook Overview

The purpose of this site is to provide electronic versions of all the exercises that are in your printed workbook. Some of the key features of this electronic workbook include the following:

- Convenient copy-to-clipboard buttons (hover over the right side of code blocks)
- Drop-down solutions provided immediately after questions for easy reference
- Integrated keyword searching across the entire site is available at the top
- Site navigation is available along the left, page navigation along the right
- Images and fonts can be scaled in the browser for improved readability

In addition to electronic versions of all the exercises, we have included numerous other resources, such as electronic versions of the posters, cheat sheets, SRL diagram, and more. Use the navigation menus on the left to access all of the site content.

Updating the E-Workbook

Tip

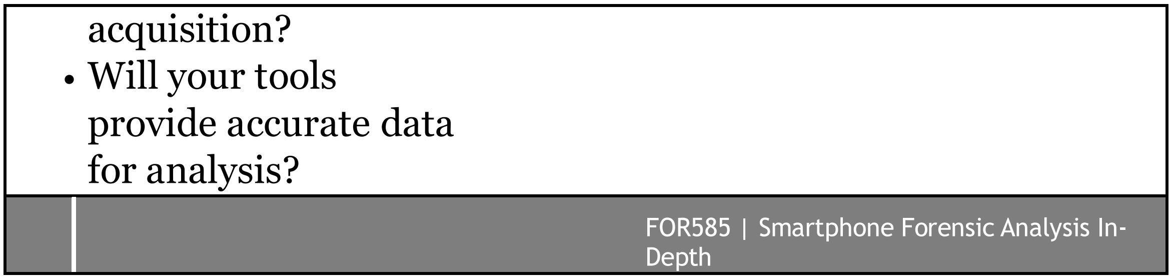
We recommend performing the update process at the start of the first day of class to ensure you have the latest

Your instructor will provide an overview of the eWorkbook. This workbook is designed in a way to help you learn, yet not show the answers unless you elect to see them. We aim to provide hints and provide a thorough walkthrough of the best methods for uncovering the answers to the questions. One thing you will learn in this course is that there isn't just one way to get to an answer. The important thing is that you learn various methods and learn to rely on the files that store the correct information.

Fundamentals of Smartphone Analysis and Android Forensic Overview

This page intentionally left blank.

- How many people do *not* use a mobile device?
- How many of those devices ar smart devices?
- Are smartphones replacing th need for laptops and desktops
- Will the data on the devices be similar? Where does it exist?
- Will your tools support



acquisition?

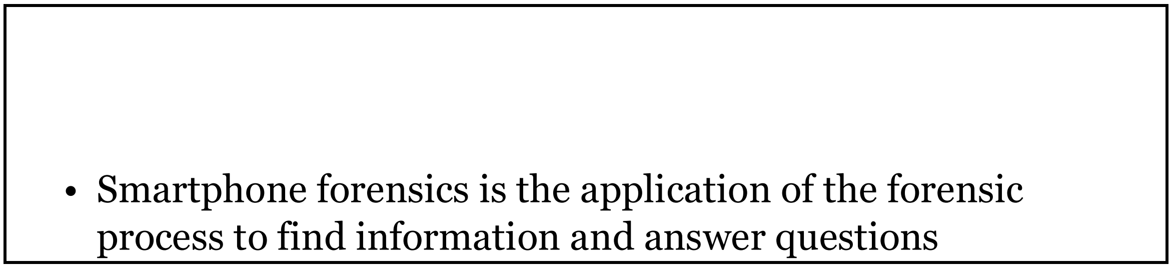
- Will your tools provide accurate data for analysis?

Smartphones are some of the most prevalent types of electronic media that exist today. Most people have at least one mobile device, and, commonly, that device will be a smartphone. Will you know how to handle these devices? Do you know what the data looks like and, more importantly, do you know where the data will be stored? Smartphones can have a variety of data storage capabilities and contain (and/or may be connected to) other storage components that may need to be addressed during the forensics process. These include:

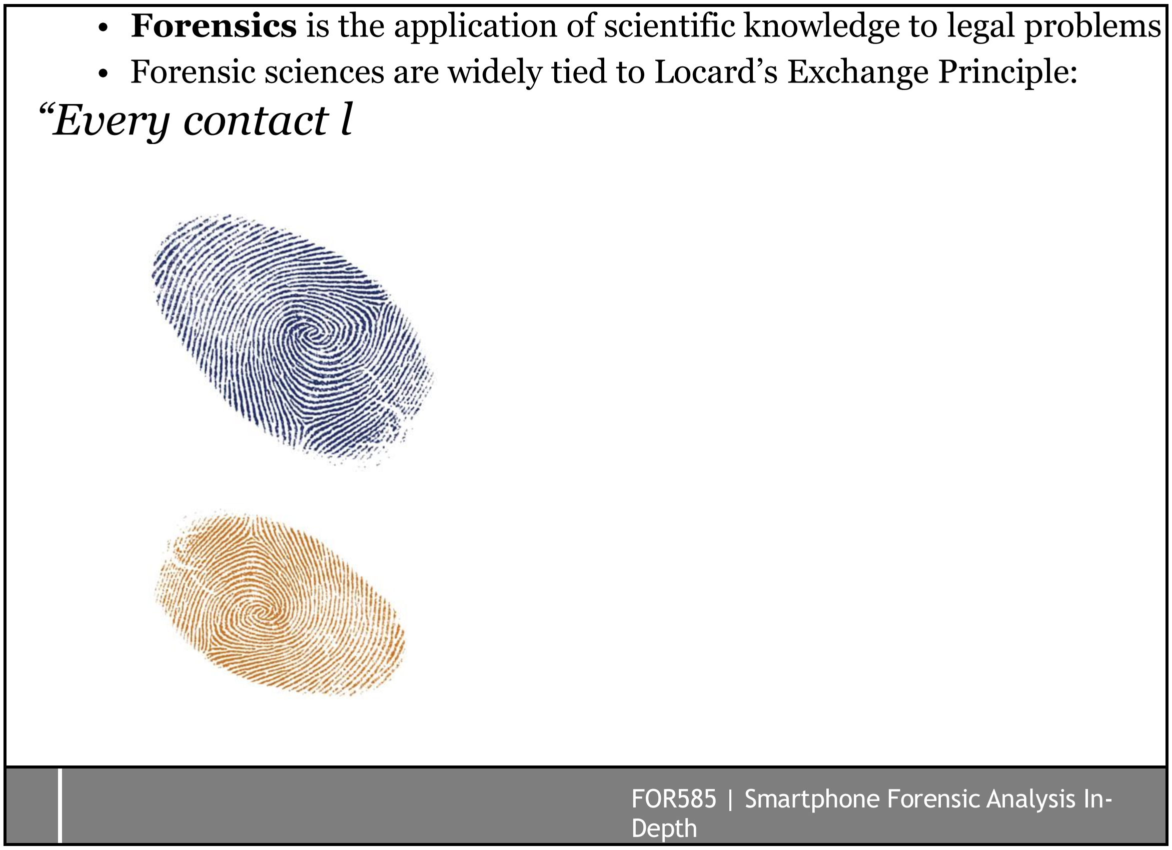
- Read-only memory for firmware
- RAM
- Flash memory for data storage
- SIM card
- Storage expansion cards (microSD)
- Other cloud-based storage
- Data on synced computer(s)
- Servers hosting backup data

Evidentiary data may exist on the smartphone, on a data-expansion card (SD type card) installed in the smartphone, in a cloud-based account that is synced or connected with the phone, or on computers that the smartphone has been synced with. Also, remember that any other phones the smartphone has communicated with may also contain evidence that may be useful to your investigation, and if what you are looking for no longer exists on the smartphone itself, that data may have been transferred to another phone it communicated with.

Smartphones also use RAM, and there are recent developments in methodology for the collection and examination of RAM from smartphones. Most devices will require additional accesses in order to extract data from RAM. This is commonly done by jailbreaking or rooting the device. More on accessing specific data categories will be covered in sections 2 and 3 of this course.



- Smartphone forensics is the application of the forensic process to find information and answer questions



- **Forensics** is the application of scientific knowledge to legal problems
- Forensic sciences are widely tied to Locard’s Exchange Principle:

“Every contact l

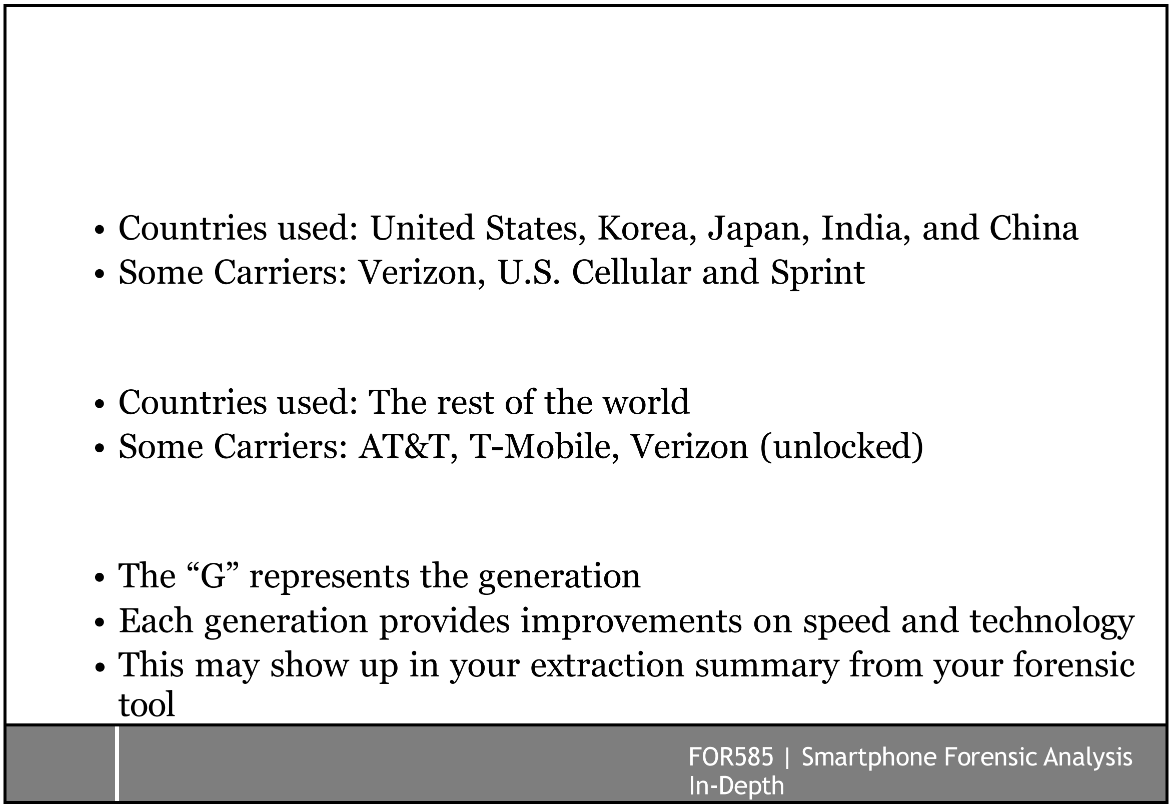
Smartphone forensics is the application of the forensic process to find information and answer questions about how a smartphone was (or was not) involved in a particular incident. Smartphone forensics is a sub-field of digital forensics, and it follows the same principles.

Forensics is the application of scientific knowledge to legal problems.[1] Forensic sciences are widely tied to Locard’s Exchange Principle: “Every contact leaves a trace.”[2] The process of digital forensics is the same as other forensics sciences, and the process in mobile device forensics closely follows the same path. In mobile device forensics, however, we also have the extra challenge of isolation of devices we are working with because they are designed to communicate with the surrounding mobile network, satellites, surrounding wireless networks, and via Bluetooth or other means.

Not all applications of digital or mobile device forensics are designed to produce evidence, but all require reliability, integrity, and veracity, whether the forensics is being performed for information security, incident response, intelligence gathering, policy compliance, remediation, research, or for criminal or civil investigations.

References:

- Webster’s Dictionary
- Prof. Edmond Locard, c. 1910



- Countries used: United States, Korea, Japan, India, and China
- Some Carriers: Verizon, U.S. Cellular and Sprint

- Countries used: The rest of the world
- Some Carriers: AT&T, T-Mobile, Verizon (unlocked)

- The “G” represents the generation
- Each generation provides improvements on speed and technology
- This may show up in your extraction summary from your forensic tool

FOR585 | Smartphone Forensic Analysis In-Depth

Smartphones generally operate using one of two mobile network technologies, which determines how the device communicates with its cellular network:

- **GSM**: Global System for
  Mobile Communications
  Some Carriers: Verizon,
  U.S. Cellular and Sprint

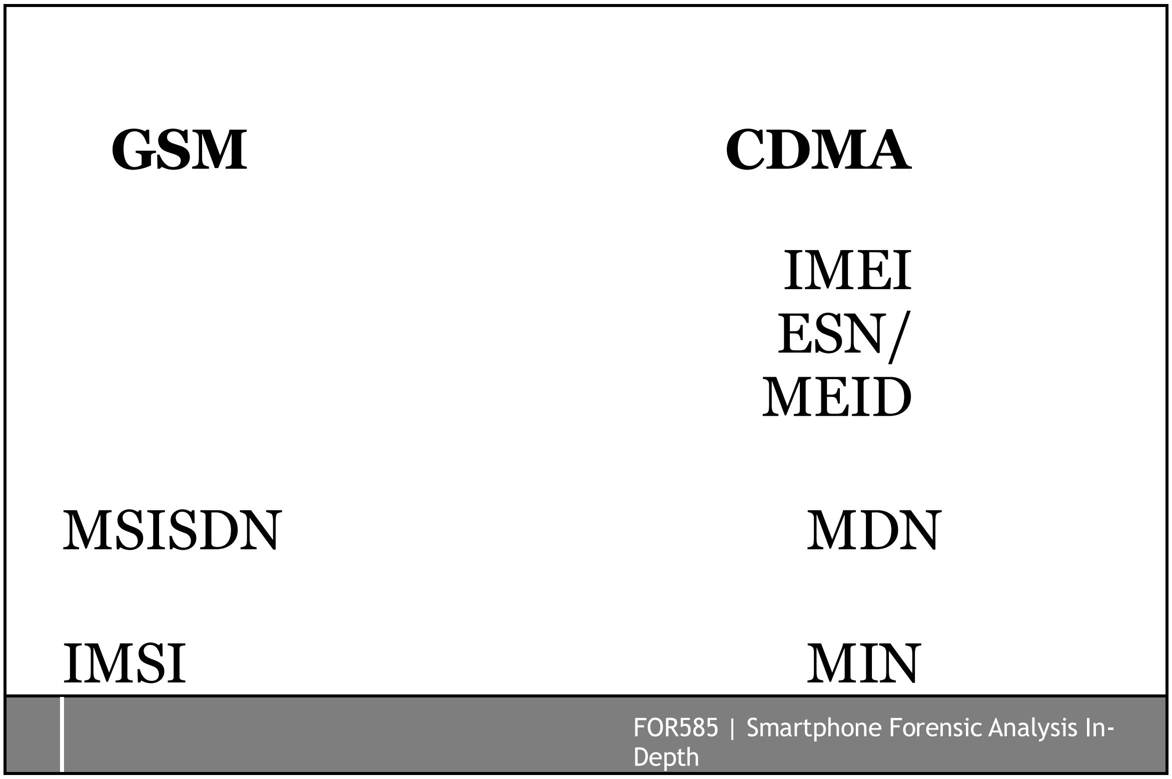
- **CDMA**: Code Division Multiple Access
  Some Carriers: AT&T, T-Mobile, Verizon (unlocked)

Although GSM and CDMA provide similar basic features and services to end users (voice calling, text messaging, and data services), they operate differently at many technical levels, making GSM phones completely incompatible with CDMA networks, and vice versa.

There are several generations of each technology, but for the purposes of smartphone handset forensics, it is primarily important to simply know whether you are working with a CDMA or GSM phone. In the past, it was easy to determine whether a phone was CDMA or GSM based on whether it contained a SIM card because only GSM phones contained SIM cards. Now, however, newer-technology (many) CDMA smartphones can have the capability to work on both CDMA and GSM networks and, therefore, may contain a SIM card or a R- UIM (Removable User Identity Module) card. Therefore, you may have to do some investigation to see whether the phone you are working with is a GSM or CDMA phone.

You can determine what network technology a phone is using by checking that phone by make and model at www.phonescoop.com, or by determining what service provider that phone uses, because the carrier either uses a CDMA or a GSM network. Knowing what general technology, the handset you're trying to examine is using is an important first step in the examination process.

The get to these settings on and Android and iOS follow the steps below. These steps will also help you with the identifiers on the next slide.
iOS – Settings > General > About
Android – Settings > About phone > Status



GSM	**CDMA**
	IMEI ESN/ MEID
MSISDN	MDN
IMSI	MIN

GSM Smartphones:
The International Mobile Equipment Identifier (IMEI) is a unique 15-digit number that identifies a GSM cellular phone handset on its network. This number is generally found under the battery of the cellular phone. The first eight digits are a Type Allocation Code (TAC), and the next six digits are the Device Serial Number (DSN). The final digit is a check digit, usually set to 0.

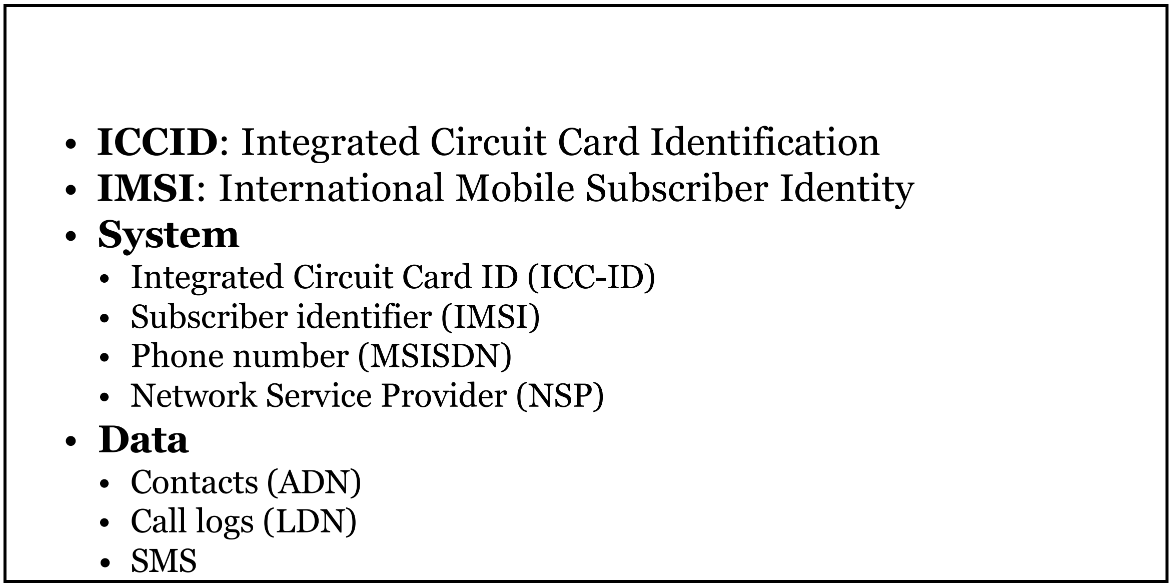
The Mobile Station International Subscriber Directory Number (MSISDN) is the phone's 15-digit, globally unique number. The MSISDN follows the International Telecommunication Union (ITU) Recommendation E.164 telephone numbering plan, composed of a 1–3-digit country code, followed by a country-specific number. In North America, the first digit is a 1, followed by a 3-digit area code.

The following links are good resources for researching IMEIs:
https://ww
w.imei.info
https://ww
w.numberi
ngplans.co
m/

CDMA Smartphones:
The **Electronic Serial Number** (ESN) is located under the battery of the cellular phone. This is a unique 32-bit number assigned to each mobile phone on the network. The ESN may be listed in decimal (11 digits) and/or hexadecimal (8 Hex digits). Be aware that the Hex version of the ESN is not a direct numeric conversion of the decimal value. An ESN converter can be found at https://www.elfqrin.com/esndhconv.php.

The **Mobile Equipment ID** (MEID), also found under the battery cover, is a 56-bit number that replaced the ESN due to the limited number of 32-bit ESN numbers. The MEID is listed in Hex, where the first byte is a regional code, the next three bytes are a manufacturer code, and the remaining three bytes are a manufacturer-assigned serial number. CDMA phones also have two other identifying numbers, namely the **Mobile Identification Number (MIN)** and **Mobile Directory Number (MDN)**. The MIN is a carrier-assigned, carrier-unique 24-bit (10-digit) telephone number. When a call is placed, the phone sends the ESN and MIN to the local tower. The MDN is the globally unique telephone number of the phone. Prior to Wireless Number Portability, the MIN and MDN were the same, but in today's environment, the customer can keep her phone number (MDN) even if she changes carriers.

Reference:
https://for585.com/3jas7 (Phonescoop.com network definitions)



- **ICCID**: Integrated Circuit Card Identification
- **IMSI**: International Mobile Subscriber Identity
- **System**
  - Integrated Circuit Card ID (ICC-ID)
  - Subscriber identifier (IMSI)
  - Phone number (MSISDN)
  - Network Service Provider (NSP)
- **Data**
  - Contacts (ADN)
  - Call logs (LDN)
  - SMS

On GSM, LTE and 5G phones, there is at least a Subscriber Identity Module (SIM) card slot, which may be located under the battery. The SIM card may be branded with the name of the network to which the SIM is registered.

Also located on the SIM card is the Integrated Circuit Card Identification (ICCID), which is an 18- to 20-digit number (10 bytes) that uniquely identifies each SIM card. The ICCID number is tied to the International Mobile Subscriber Identity (IMSI), which is typically a 15-digit number (56 bits) consisting of three parts, including the Mobile Country Code (MCC; 3 digits), Mobile Network Code (MNC; generally 3 digits in India, Malaysia, a lot of the Caribbean and South America, and several other countries; may be 2 digits as well),[1] and Mobile Station Identification Number (MSIN; 9 digits in the U.S. and Canada, and 10 digits elsewhere), which are stored electronically within the SIM. The IMSI can be obtained either through analysis of the SIM or from the carrier.[2, 3]

References:

- https://for585.com/b-imd (Wikipedia entry for Mobile Country Code)
- https://for585.com/shafik
- https://for585.com/c3jki (Images)

The Universal Integrated Circuit Card (UICC) is the smart card that runs applications for devices to function. For example, the SIM is simply an application running on a UICC. As you know, multiple applications can run on a single device or SIM. It is not common, but more than one SIM can reside on a UICC. These are often referred to as SIM stackers. Most individuals never use the term UICC, but instead simply refer to the card as a SIM. SIM cards can range in size from 16 KB to 1+ GB.

SIM (Subscriber Identity Module) cards are smart cards used by Network Service Providers (NSPs) to authenticate the user to a network. A variety of mobile device technologies use SIM cards in different manners. For example, GSM phones won't power on for specific handsets unless the device can function due to the use of Wi-Fi when the device is missing the SIM card.

Universal Mobile Telecommunications System (UMTS), otherwise nicknamed "world phones", featuring 3G, 4G, and LTE requires a SIM card to authenticate users to the network. A good example of a UMTS device is a Verizon iPhone 4s. The Verizon iPhone 4s contains a micro-SIM. This differs from the Verizon iPhone4, which is strictly CDMA and does not contain a SIM card.

The authentication key (Ki) is not copied during a forensic acquisition. Ki is used to authenticate the SIM on the GSM network. This key is unique for each SIM card.

NOTE: Ki has been known to be decrypted by certain applications and groups. Once a network senses that two devices are using two SIM cards with the same Ki, both are disabled. Although you won't learn how to decrypt and capture Ki in this course, it doesn't mean it hasn't been done before.

The Removable User Identity Module (R-UIM) is a SIM card that was designed for CDMA handsets. This is commonly seen in China and is used by China Unicom.

Remember, the SIM card is not an SD card. They are completely different components of smart devices. For UICCs, the easiest thing to remember is that GSM devices use SIM cards, and CDMA devices use R-UIM. Forensic acquisition of R-UIM devices is not supported by all the commercial tools supporting SIM cards. A great tool for R-UIM devices is the SIMIS 3G kit from Teel Technologies.

SIM cards essentially have two sections of information. The first is the system area, which cannot be locked by a user PIN. The second is the data area, which contains the user data, including the contacts, dialed calls, and SMS messages. Deleted SMS messages are commonly recovered from SIM cards.

The system area contains the following equipment identifiers of interest:

- **Integrated Circuit Card ID (ICCID)**: An 18-digit number with a single check digit, which assists in error detection. This number is essentially the serial number of the SIM card. The ICCID is sometimes printed on the external side of the SIM card and is always stored internally on the media. Most forensic kits acquire this data, even on locked SIM cards.

Here is an example of an ICCID with a detailed

explanation of the numbering:

8996430671601871055 – 19-digit ICCID (18 digits +

one single check digit) 89 96 43 0671 60 187105 5

89:
Represe
nts the
Telecom
ID 96:
Represe
nts the
country
code 43:
Represe
nts the
network
code
0671: Represents the month and
year of manufacturing 60:
Represents the switch
configuration code
187105:
Represents

the SIM
number 5:
Represents
the check
digit

A tool such as https://www.numberingplans.com/ offers SIM analysis tools, which can assist the examiner in identifying the components listed. Understanding each chunk of the ICCID makes verifying the results from www.numberingplans.com, or your forensic tool, much easier.

- **International Mobile Subscriber Identity (IMSI)**: Used by the NSP to identify the device and the subscriber on handsets. The IMSI is normally 15–16 digits but can be shorter, depending on the NSP.

Here is an example of an IMSI with a detailed breakdown:
3101501234567892 – IMSI

310 150 1234567892

310: Represents the Mobile Country Code (MCC)
150: Represents the Mobile Network Code (MNC) (two digits used for European Standard and other countries who don't use three digits, three digits used for North American Standard, India, the Caribbean, and South America); this number is the SIM card's home network
1234567892: Represents the 10-digit Mobile Subscriber Identification Number (MSIN), which is the unique ID for the subscriber

- **Mobile Subscriber Integrated Services Digital Network Number or Mobile Station International Subscriber Directory Number (MSISDN)** (may be interpreted in other manners): Phone number for the handset. This number is not always easy to recover because users often are required to save their own phone numbers to the SIM card or the handset. The MSISDN is not hard-coded to the device by the NSP.
- **Network Service Provider (NSP)**

The file structure of a SIM card contains master files (MFs), dedicated files (DFs), and elementary files (EFs). Every SIM card must contain at least one MF, which is the root directory of the file system. The MF folder contains the DF and EF. The DFs are directories that contain elementary files. The EFs are containers storing user data. As shown above, an EF can fall directly under a MF.

The hexadecimal representation of the first byte of the SIM directories is:
3
F

=

M
a

ster file 7F = Dedicated file
2F = Elementary file under the master file 6F = Elementary file under a dedicated file

For example, the DFTelecom contains several EFs. These EFs are comprised of user data, such as the phone number, contacts, SMS, and dialed calls. Each category of data is assigned to an individual EF. A further explanation of this is

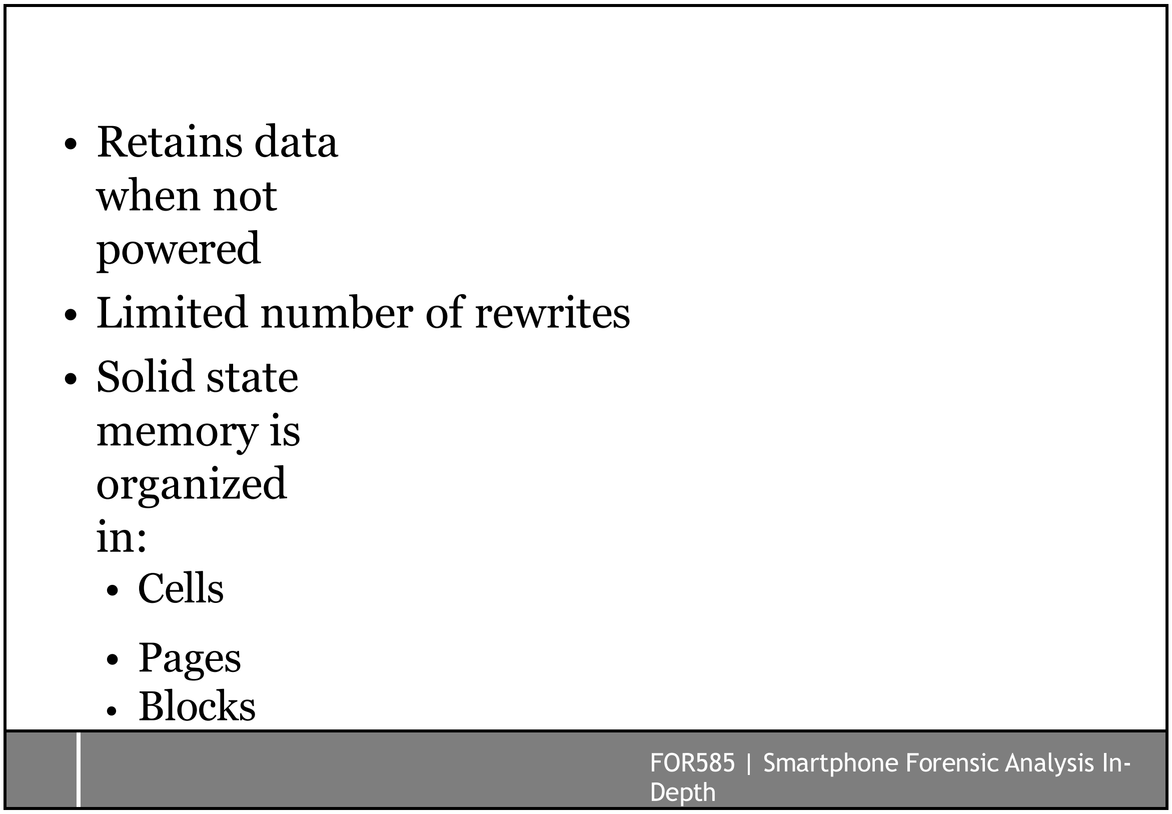
3F00 = Master f

ile

7F10 = DFTelecom

6F40 = EF MSISDN
6F3A = EF ADN
6F3C = EF SMS
6F44 = EF LDN

To further explain this example, the MF containing the Telecom DF contains four EFs. The EF MSISDN (Mobile Station International Subscriber Directory Number) contains the phone number for the SIM card. The EF ADN (Abbreviated Dialing Number) contains the contacts for the SIM card. The EF SMS contains the SMS message communications. Deleted SMS messages may reside here and should be recovered for forensic examination. Finally, the EF LDN (Last Dialed Number) is comprised of the most recent outgoing calls. Only outgoing calls are tracked on SIM cards.



- Retains data when not powered
- Limited number of rewrites
- Solid state memory is organized in:
  - Cells
  - Pages
  - Blocks

FOR585 | Smartphone Forensic Analysis In-Depth

Solid state memory retains data even when it is not powered on. It can accept only a limited number of writes in its lifetime, however, and eventually wears out.

Wear leveling and garbage collection, which are discussed in detail in upcoming slides, are techniques used to extend the life of the memory and increase storage efficiency.

There are two types of solid-state memory used in smartphones, including NAND and NOR. NOR is commonly used for firmware, while NAND is generally used for storage of user data.

NAND versus NOR flash

- NOR commonly used for firmware
- NAND commonly used for user data
- NAND blocks are subdivided into pages
- NAND is more commonly used today

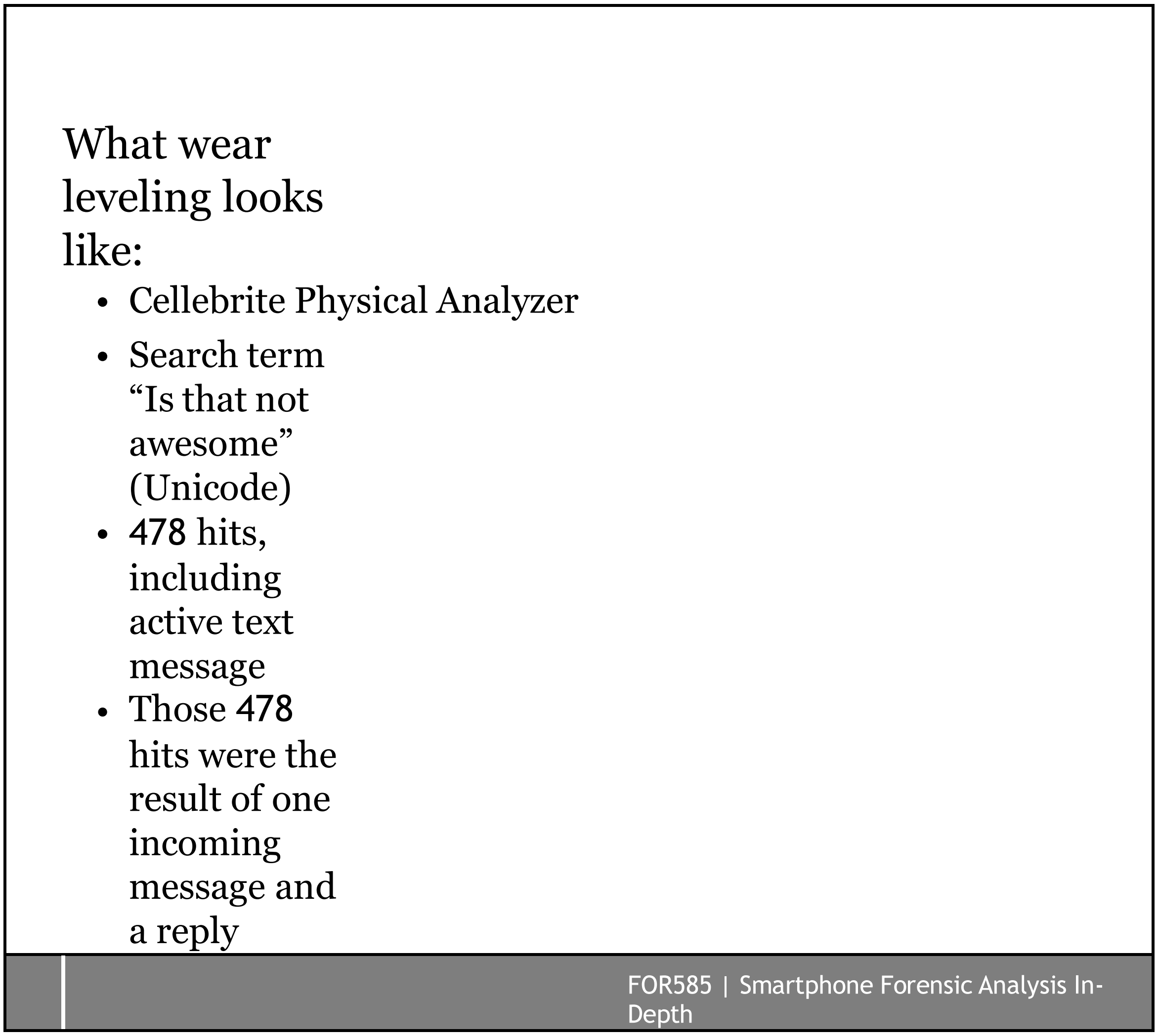
Smartphones store data on one or more chips within the phone that use flash memory technology. They may also store additional data on removable storage cards (microSD) and/or SIM cards, which are discussed later in this section.

For solid state memory, data is written to flash memory in units called **pages**. **Pages** are made up of multiple
cells. **Blocks** are made up of multiple pages.[1]

Flash memory can only be erased in larger units called **blocks**.

An empty unit of flash memory is filled with 1s rather than with 0s as with traditional hard disk drive type media. Erased blocks are overwritten with 0x1111111.

Reference:
[1] https://for585.com/9ahez (Image)



What wear leveling looks like:

- Cellebrite Physical Analyzer
- Search term "Is that not awesome" (Unicode)
- 478 hits, including active text message
- Those 478 hits were the result of one incoming message and a reply

As previously mentioned, data is written to flash memory in units called **pages**. **Pages** are made up of multiple **cells**. However, flash memory can only be erased in larger units called **blocks**. **Blocks** are made up of multiple pages.

If the data in some of the pages of a particular block are no longer needed (**stale pages**), only the pages with good data in that block are read and are then rewritten into another previously erased empty block. Then the free pages left by not moving the stale data are available for new data. This process is called *garbage collection*, and it happens automatically in the background as a mobile device operates. All solid-state storage includes some level of garbage collection, but they may differ in when and how fast they perform the process.

Unlike hard disk drives, flash memory-based solid-state devices (SSDs) don't contain spinning parts. In flash- based storage devices, data is written to pages (sometimes called cells) in an SSD that is made up of multiple flash memory chips. Cells have a finite life cycle, meaning that each one can only be written to so many times before it wears out and eventually becomes unprogrammable.

Wear leveling is a process that is designed to extend the life of solid-state storage devices. Solid state storage is made up of multiple microchips that store data in blocks. Each block can tolerate only a finite number of write/erase cycles before becoming unreliable. For example, NAND flash is typically rated at about 100,000 program/erase cycles. Wear leveling arranges data so that write/erase cycles are distributed evenly among all the blocks in the device, thereby allowing the device to last longer.[1]

In practical terms, the wear leveling process can result in many search hits for any search term during the examination of data from a physical image of a smartphone or other device using flash-based memory for data storage.

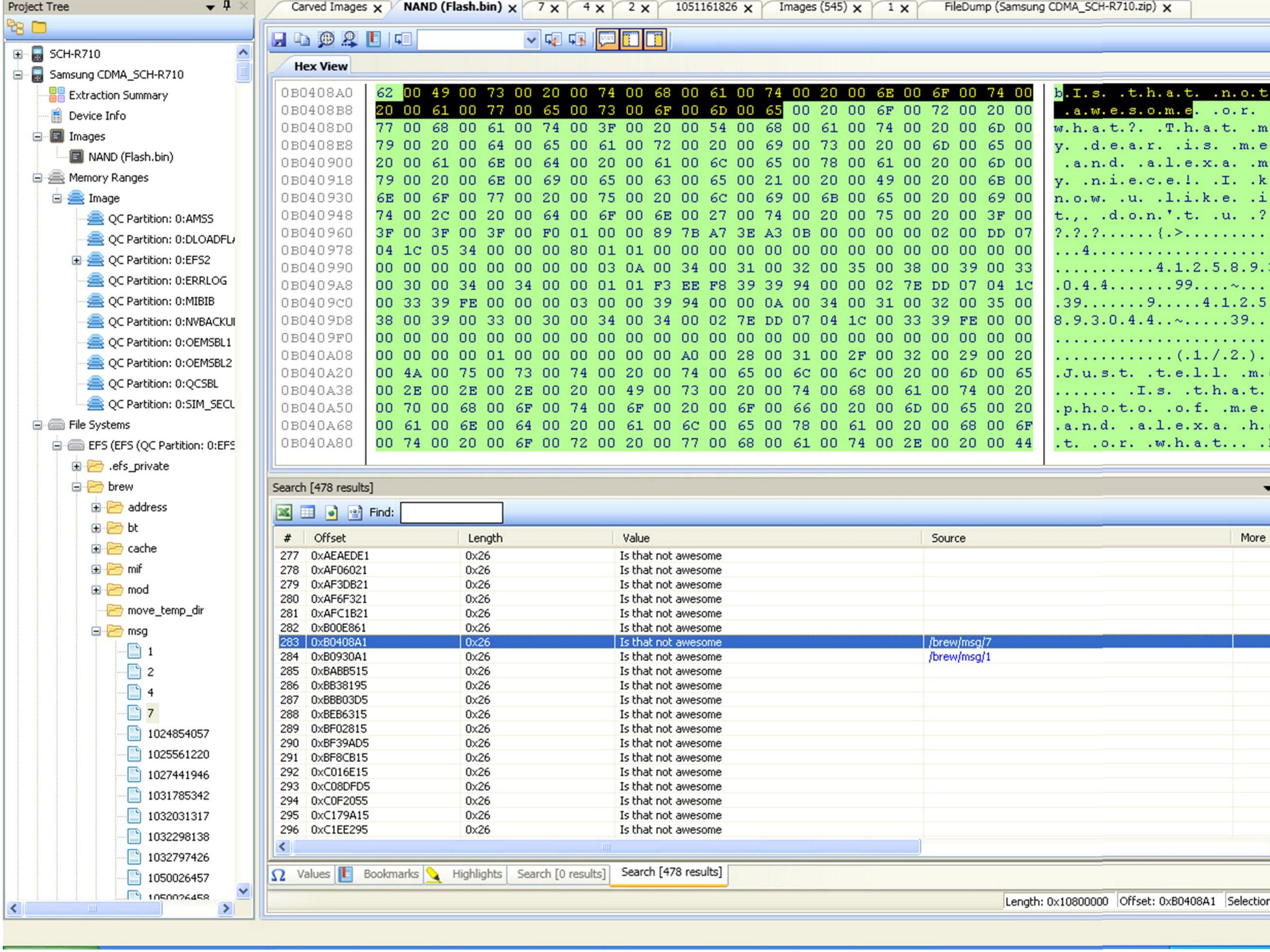
In the slide's example, the search term "Is that not awesome" was conducted across the full physical image of the NAND chip on a Samsung SCH R710 cell phone.

This search term resulted in 478 search hits across all the data on the physical image, although there were only two active files with a keyword match on the phone.

The wear leveling process can result in the content of deleted data of all types being written multiple times in multiple locations on a phone, leaving behind a plethora of potential evidentiary data.

Reference:

- https://for585.com/se1mv (Cypress white paper on wear leveling)



```
0B0408A0  62 00 49 00 73 00 20 00 74 00 68 00 61 00 74 00 20 00 6E 00 6F 00 74 00  b.I.s. .t.h.a.t. .n.o.t
0B0408B8  20 00 61 00 77 00 65 00 73 00 6F 00 6D 00 65 00 20 00 6F 00 72 00 20 00   .a.w.e.s.o.m.e. .o.r.
0B0408D0  77 00 68 00 61 00 74 00 3F 00 20 00 54 00 68 00 61 00 74 00 20 00 6D 00  w.h.a.t.?. .T.h.a.t. .m
0B0408E8  79 00 20 00 64 00 65 00 61 00 72 00 20 00 69 00 73 00 20 00 6D 00 65 00  y. .d.e.a.r. .i.s. .m.e
0B040900  20 00 61 00 6E 00 64 00 20 00 61 00 6C 00 65 00 78 00 61 00 20 00 6D 00   .a.n.d. .a.l.e.x.a. .m
0B040918  79 00 20 00 6E 00 69 00 65 00 63 00 65 00 21 00 20 00 49 00 20 00 6B 00  y. .n.i.e.c.e.!. .I. .k
0B040930  6E 00 6F 00 77 00 20 00 75 00 20 00 6C 00 69 00 6B 00 65 00 20 00 69 00  n.o.w. .u. .l.i.k.e. .i
0B040948  74 00 2C 00 20 00 64 00 6F 00 6E 00 27 00 74 00 20 00 75 00 20 00 3F 00  t.,. .d.o.n.'.t. .u. .?
0B040960  3F 00 3F 00 3F 00 F0 01 00 00 89 7B A7 3E A3 0B 00 00 00 00 02 00 DD 07  ?.?.?.....{.>.........
0B040978  04 1C 05 34 00 00 00 80 01 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ...4....................
0B040990  00 00 00 00 00 00 00 00 03 0A 00 34 00 31 00 32 00 35 00 38 00 39 00 33  ...........4.1.2.5.8.9.3
0B0409A8  00 30 00 34 00 34 00 00 01 01 F3 EE F8 39 39 94 00 00 02 7E DD 07 04 1C  .0.4.4......99...~....
0B0409C0  00 33 39 FE 00 00 00 03 00 00 39 94 00 00 0A 00 34 00 31 00 32 00 35 00  .39......9....4.1.2.5
0B0409D8  38 00 39 00 33 00 30 00 34 00 34 00 02 7E DD 07 04 1C 00 33 39 FE 00 00  8.9.3.0.4.4..~.....39..
0B0409F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ........................
0B040A08  00 00 00 00 01 00 00 00 00 00 00 A0 00 28 00 31 00 2F 00 32 00 29 00 20  .............(.1./.2.).
0B040A20  00 4A 00 75 00 73 00 74 00 20 00 74 00 65 00 6C 00 6C 00 20 00 6D 00 65  .J.u.s.t. .t.e.l.l. .m.
0B040A38  00 2E 00 2E 00 2E 00 20 00 49 00 73 00 20 00 74 00 68 00 61 00 74 00 20  ....... .I.s. .t.h.a.t.
0B040A50  00 70 00 68 00 6F 00 74 00 6F 00 20 00 6F 00 66 00 20 00 6D 00 65 00 20  .p.h.o.t.o. .o.f. .m.e.
0B040A68  00 61 00 6E 00 64 00 20 00 61 00 6C 00 65 00 78 00 61 00 20 00 68 00 6F  .a.n.d. .a.l.e.x.a. .h.
0B040A80  00 74 00 20 00 6F 00 72 00 20 00 77 00 68 00 61 00 74 00 2E 00 20 00 44  .t. .o.r. .w.h.a.t... .
```

#	Offset	Length	Value	Source	More
277	0xAEAEDE1	0x26	Is that not awesome		
278	0xAF06021	0x26	Is that not awesome		
279	0xAF3DB21	0x26	Is that not awesome		
280	0xAF6F321	0x26	Is that not awesome		
281	0xAFC1B21	0x26	Is that not awesome		
282	0xB00E861	0x26	Is that not awesome		
283	0xB0408A1	0x26	Is that not awesome	/brew/msg/7	
284	0xB0930A1	0x26	Is that not awesome	/brew/msg/1	
285	0xBABB515	0x26	Is that not awesome		
286	0xBB38195	0x26	Is that not awesome		
287	0xBBB03D5	0x26	Is that not awesome		
288	0xBEB6315	0x26	Is that not awesome		
289	0xBF02815	0x26	Is that not awesome		
290	0xBF39AD5	0x26	Is that not awesome		
291	0xBF8CB15	0x26	Is that not awesome		
292	0xC016E15	0x26	Is that not awesome		
293	0xC08DFD5	0x26	Is that not awesome		
294	0xC0F2055	0x26	Is that not awesome		
295	0xC179A15	0x26	Is that not awesome		
296	0xC1EE295	0x26	Is that not awesome		

Length: 0x10800000 Offset: 0xB0408A1 Selection

Garbage collection is the process by which solid state memory improves write performance, proactively eliminating the need for whole block erasures prior to every write operation.

As we just learned, data is written to flash memory in units called pages (made up of multiple cells). However, the memory can only be erased in larger units called blocks (made up of multiple pages). If the data in some of the pages of the block are no longer needed (also called stale pages), only the pages with good data in that block are read and rewritten into another previously erased empty block. Then the free pages left by not moving the stale data are available for new data. This process is called *garbage collection*. All SSDs include some level of garbage collection, but they may differ in when and how fast they perform the process.[1]

Traditional hard disk drives simply overwrite unneeded data blocks with new data. Flash memory, however, must erase the unneeded data blocks before new data can be written. The garbage collection process works in the background and systematically identifies which memory cells contain unneeded data. Those blocks of unneeded data are cleared during off-peak times to maintain optimal write speeds during normal operation of the device.[2]

There is a great deal of potential complexity to the garbage collection process, which can occur at any of the following levels:

- Operating system
- Application code/programming language
- Hardware controllers
- How the programmer chooses to implement garbage collection

Each of the programming languages (Java, .NET, C#, and so on) have their own Garbage Collection (GC) functions and methods. The developer can choose to implement them (or not) in his code. Some developer platforms have their own garbage collection implementations outside of the standard language-based methods.

Apple doesn't use any sort of garbage collection process across the entire device. It is not needed as the Automatic Reference Counting (ARC) handles the regular controls of memory and data allocation. A discussion of this issue can be found at: https://www.macobserver.com/tmo/article/apple-warns-developers-garbage-collection-is-dead-move-to- HYPERLINK "http://www.macobserver.com/tmo/article/apple-warns-developers-garbage-collection-is-dead-move-to-arc/"arc/

More information about how Android Dalvik uses and configures garbage collection can be found here:
https://developer.android.com/studio/profile/memory-profiler#LogMessages

References:

- https://for585.com/b0hog (Wikipedia entry on garbage collection)
- https://for585.com/b7rov (TechTarget article on garbage collection)

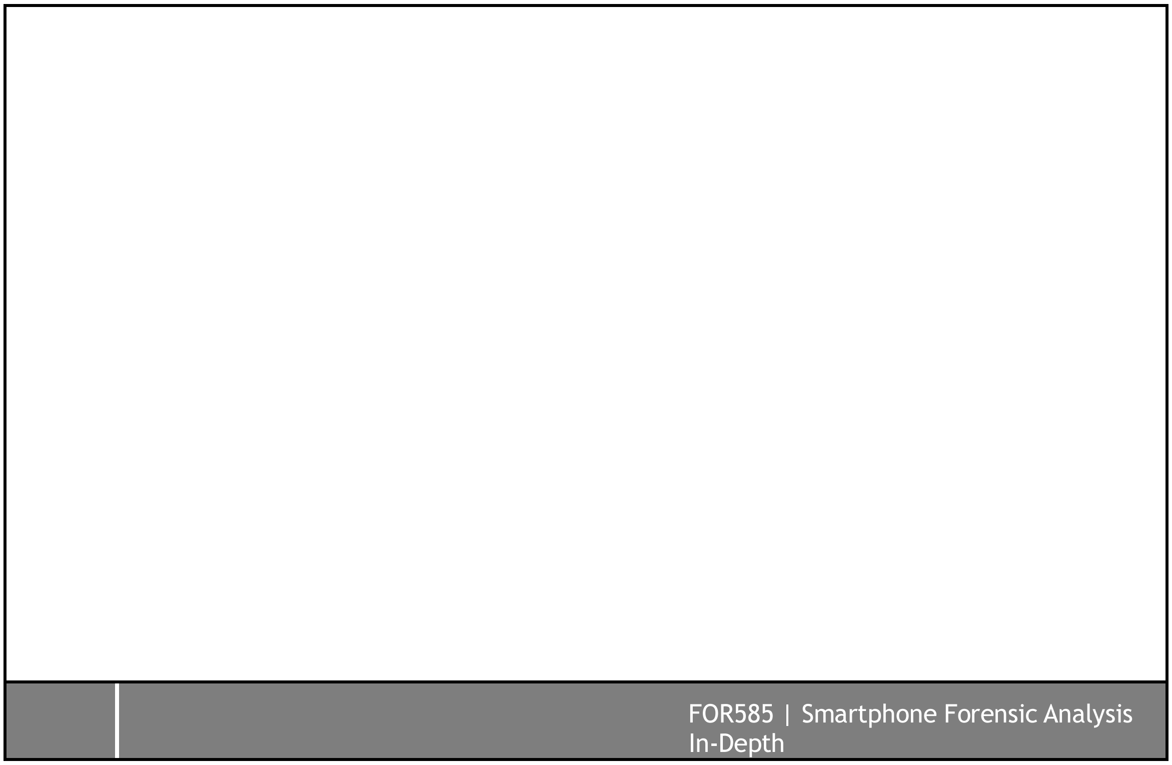
FOR585 | Smartphone Forensic Analysis In-Depth

Smartphones commonly have SD cards and SIM cards, which contain unique data that may not be captured during forensic acquisition of the device. It is best to assume that the data saved on these devices (the SIMs and SD cards) will not be acquired during the smartphone acquisition and to acquire each individually.

SD cards can be acquired using most smartphone forensic tools; however, a tool like FTK imager is free and may work faster. When acquiring an SD card, ensure that your card reader is write protected to ensure changes are not made to the media. Always test your write protection method prior to acquiring evidence.

Prior to pulling a SIM card from a device, make sure you have already acquired the data. Some SIM cards will lock the device and possibly put it in a BFU state. SIM cards can be acquired using most smartphone forensic tools. With modern devices, the SIM card is often not acquired during forensic extraction, and data can be saved

to these devices. We will cover more on SIM card data in Section 2 of this course and how Android saves user data to the SIM. Best practices state to acquire the SIM card and review it for user-created data.
Methods for SIM extraction and bonus Lab 1 are included in the Bonus Section within your media files. Try this lab if you are curious as to what a file system extraction of a SIM card reveals.



FOR585 | Smartphone Forensic Analysis In-Depth

SD cards are the most compact data storage available today. SD cards are often available for use in smartphones and sometimes improve the performance of the device since they prevent the internal memory on the device from being overused. Some devices, such as iOS devices, do not allow for the use of an SD card.
This forces the consumer to purchase a device with a greater storage capacity.

There are three physical sizes of SD cards: original, mini, and micro. Most of the smaller SD cards come with an adapter that can be used to acquire the media. Most SD card readers have slots for each physical size.

Storage size differs for SD cards. Standard SD cards are 2 GB maximum in size. Micro Secure Digital High Capacity (SDHC) stores between 4 GB and 32 GB and are the most common SD cards found in smartphones. A 32 GB SD card allows for a lot more storage than what we are used to seeing on these devices. Secure Digital Extended Capacity (SDXC) can store between 32 GB and 2 TB of data. Most modern Samsung (and not only Samsung) phones supporting SDXC can reach at least 512 GB. We see them quite often.

Remember, these devices are transient. Consumers may reuse these SD cards, and you might be surprised to find information relating to other smartphones, cameras, and computers during your investigation.

Although it may take longer, the best method for acquiring SD cards found in smartphones includes:

- Acquire the smartphone with the SD card inserted in the handset.
- Remove the SD card and acquire using a tool such as FTK Imager.
- Examine with both traditional forensic tool (EnCase, XWAYS, Autopsy, FTK, etc.) and mobile device forensic tools.

FOR585 | Smartphone Forensic Analysis In-Depth

SD cards can store data in any format. Examiners must take this into consideration when conducting a forensic examination of an SD card. Smartphones, however, use the SD cards to store application data, user-created files, backup files, and possibly encrypted volumes. Depending on the smartphone, the data stored and acquired from the SD card may vary.

A full file system examination of the SD card should be conducted in your forensic tool of choice. XWAYS, Autopsy, Inspector, EnCase, or FTK provide access to all the files (deleted and active) on the media to include encrypted volumes. Mobile device forensic tools may be required for parsing third-party applications and phone backup files. Andrew Hoog, from viaForensics, references the use of SD cards in Android devices in his book *Android Forensics: Investigation, Analysis and Mobile Security for Google Android*.

The examiner should use tools that allow him/her to search by file extension, keyword search, mount files, and carve data that may reside on the media, but isn't being interpreted. Analysis for the Android portion of this SD card are covered in the corresponding sections of this course.

Deleted images may be recovered from the SD card. Depending on the SD card and the process for deleting the image, EXIF data may be recoverable. FTK Imager shows the deleted images from the SD card but may not display the EXIF data. Should you not find any EXIF data, it is best to export the image and attempt to recover the data using another tool or a hex editor.

Many smartphones contain microSD data storage expansion cards that are used for increased storage and functionality. These storage cards can contain data that has great evidentiary value.

Most MicroSD data storage cards are generally formatted with FAT file system; however, some are formatted NTFS, and some contain proprietary formatting. Many phones allow the user to encrypt data stored on the microSD card, and some of these cards are locked to the device they are installed in and are not readable outside the device. You may find that a microSD card has been used in several smartphones or other devices.

If possible, microSD expansion cards should be processed as you would traditional media, including the use of write protection devices. Processing data expansion cards while installed in a device may result in changes to the date and timestamps on the card, and so the cards should first be processed outside the phone, and then with the phone if you choose to see the data in its native context. Some expandable storage cards are locked to the device and are not readable outside the device and, therefore, need to be processed while installed in the device.

Some Android devices allow for external and additional internal storage. The standard SD cards range in size and can be added and removed from the device by the user. Once inserted, the SD card must be mounted for the Android to access the media. The Android YAFFS2 file system relies on external SD cards for additional storage. eMMC (embedded multimedia cards) were introduced with the EXT file system. The reasoning was to allow more internal storage that didn't rely on or use up the NAND flash memory. Keep in mind that the trend is no longer to use an SD card that can be removed from the device. ISP supports many eMMC devices. An alternative to ISP is to acquire the device as a USB mass storage device, similar to what your instructor will demo in the next few slides using FTK Imager.

Android devices may have removable batteries, but this is no longer the standard. Removable SD cards make the Android and iOS devices very different smartphones. This used to be one of the biggest differences in the two smartphones. However, Android phones don’t always allow for SD cards to be inserted as they have in the past. SD cards can either be physical (removable) or emulated, which makes forensic acquisition and analysis more challenging for examiners when trying to put together the puzzle pieces. Third-party applications can store unique data on the SD card and, if separated, the application cannot be fully parsed, depending on how the data is stored by the application. Acquiring Android devices with SD cards is a lengthy process. When time allows, we recommended the following steps:

If the phone is powered off, acquire the SD card first, using methods we will demo in the next few slides or by referring to the bonus material included within the media files provided with the course.

- If the phone is powered on, acquire the phone with the SD card in it, even if the SD card is not mounted by the Android file system. You do not want to power off the device to remove external media if you made the choice to keep the device powered (i.e., preventing the device from locking or avoiding encryption).
- Ensure that you have acquired the SD card separate from the phone as well as inside the phone, where time allows.

Make sure to document all steps to ensure that nothing is overlooked, and your methods are as forensically sound as possible. (For example, if the device was rooted during acquisition, this should be noted.) Keep in mind that a phone can only pull data from an SD card that the phone “understands.” This means if the user has tax documents or encrypted containers stored on the SD card prior to using it with an Android, this data could be overlooked during acquisition because the phone doesn’t understand that type of data. You, the examiner, need to make sure you make every effort to capture all data from the smartphone media. This means you should acquire the SD card as traditional digital media, in a write-protected manner, and not just through the smartphone.

The pictures above depict the contents of a file on an encrypted and decrypted media when removed from an Android.

First, check the device to see if the media card is encrypted. Encrypted media cards can only be decrypted using the device that applied the encryption. This can be done in the settings of the Android.

In order to successfully acquire meaningful data from the associated media cards, encryption settings should be turned off before removing the media card. Make sure to double-check that your acquisition contains readable data!

In modern Android devices, this can be accessed through
Settings > Biometrics and Security > Encrypt/Decrypt SD card > Enter your device pin/passcode/password and the data will be encrypted/decrypted.

HOT

COLD

FOR585 | Smartphone Forensic Analysis In-Depth

Mobile devices, including smartphones, present a variety of unique challenges to investigators involved in forensic examinations of data from these devices. Part of the challenge arises from the fact that there are so many different makes and models of phones using a variety of underlying operating systems. Because of this, there is no one-size-fits-all forensic solution for mobile devices, and in many cases, numerous tools may need to be used to obtain the data needed for a particular investigation. Additionally, traditional digital forensics concepts may not apply due to the way flash memory functions.

Mobile devices are constantly changing when powered on, and there is no way to write block a mobile device because they communicate using modem protocols, such as AT commands and others. Because of these factors, the goal in forensics is to make as little change as possible and to document those changes that were made to the device during the forensic process.1

Just powering on a smartphone generates changes to the data and traces that can be observed during a forensic examination of the device. This is true even if the user does not operate the device other than just powering it on. The device, once powered on, reaches out to the mobile network to authenticate on the system and may store last-location data related to mobile towers or GPS if it is enabled.

At some point, however, with few exceptions, the phone most likely must be turned on in order to work with it, unless the capability for data extraction through physical, JTAG, ISP (In System Programming), or chip-off methods exists. Therefore, we need to take steps to make as few changes to the device as possible and to control and document the changes we do make.

Another consideration is the current state of the device when you receive it. Is the device HOT or COLD? A HOT device would be one that was recently unlocked with the passcode by the user. A COLD device would be one that was freshly restarted and has yet to be unlocked. These concepts make a difference in handling and how the tools can access the data. For more information on HOT and COLD devices, refer to the Cellebrite webinar provided by Heather Mahalik and Shahar Tal on Android Encryption (https://www.cellebrite.com/en/resources/webinars/).

For more information on the worse mistakes to make on iOS devices, refer to the blog by Elcomsoft: https://blog.elcomsoft.com/2020/01/the-worst-mistakes-in-ios-forensics/. It's crazy to think that simply looking at the phone could hurt your chances of accessing it if you reset the biometric lock.

As stated earlier, smartphones are specifically designed to communicate with other devices, whether through the mobile network, a data connection, Bluetooth, or similar wireless technologies.

To prevent incoming calls, texts, and other data from affecting the integrity of data of potential evidentiary value on the smartphone, as well as to prevent overreaching legal authority, attempts should be made to block incoming and outgoing signals to the device.

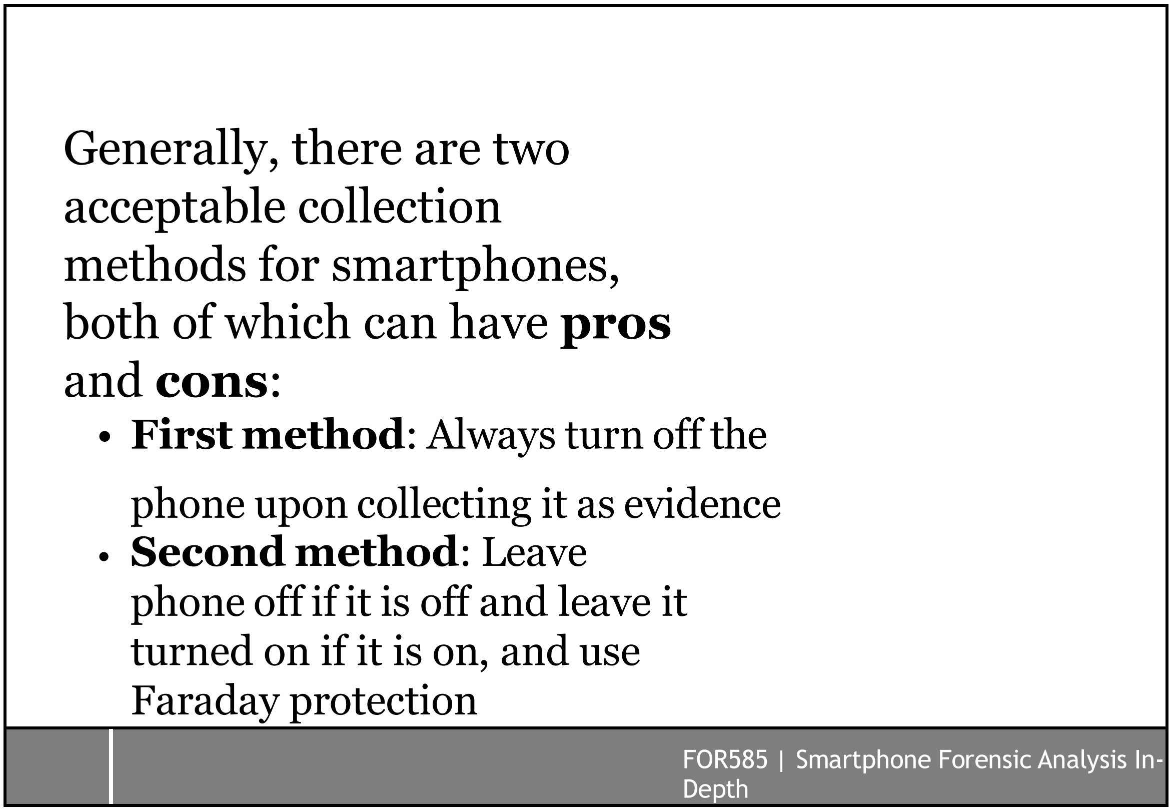
Additionally, it is possible to remotely wipe a smartphone or other mobile device either by use of a variety of mobile applications or by having the carrier remotely wipe the device by reporting it stolen and requesting that it be wiped.

Common methods for isolating smartphones include Radio Frequency (RF) blocking container or jamming appliances.* Be aware that blocking radio frequency signals will drain the battery as the smartphone tries to connect to the mobile network. Faraday devices can be expensive, and use of the devices may not prevent the alteration of mobile phone data if the device fails.[2] It is also difficult to operate touchscreen devices through Faraday materials, and not all Faraday technologies are see-through or flexible enough to allow manipulation of a smartphone. Additionally, Faraday technologies have a fairly high failure rate and cannot be completely relied on to block wireless signals.[3] If a Faraday failure occurs, document any known changes that were made to the phone, including incoming text messages, calls, or other communications.

*Be sure that it is legal for you to use signal jamming equipment. In the United States, most private and public organizations do not have legal authority to do so.

References:

- https://for585.com/byg0f ("Cellular Phone Evidence Data Extraction and Documentation", V.3, Cindy Murphy)
- https://for585.com/0q38a (SWGDE Best Practices for Mobile Phone Forensics)
- https://for585.com/t5ugz (Eric Katz's dissertation on Faraday solutions, Purdue)



Generally, there are two acceptable collection methods for smartphones, both of which can have **pros** and **cons**:

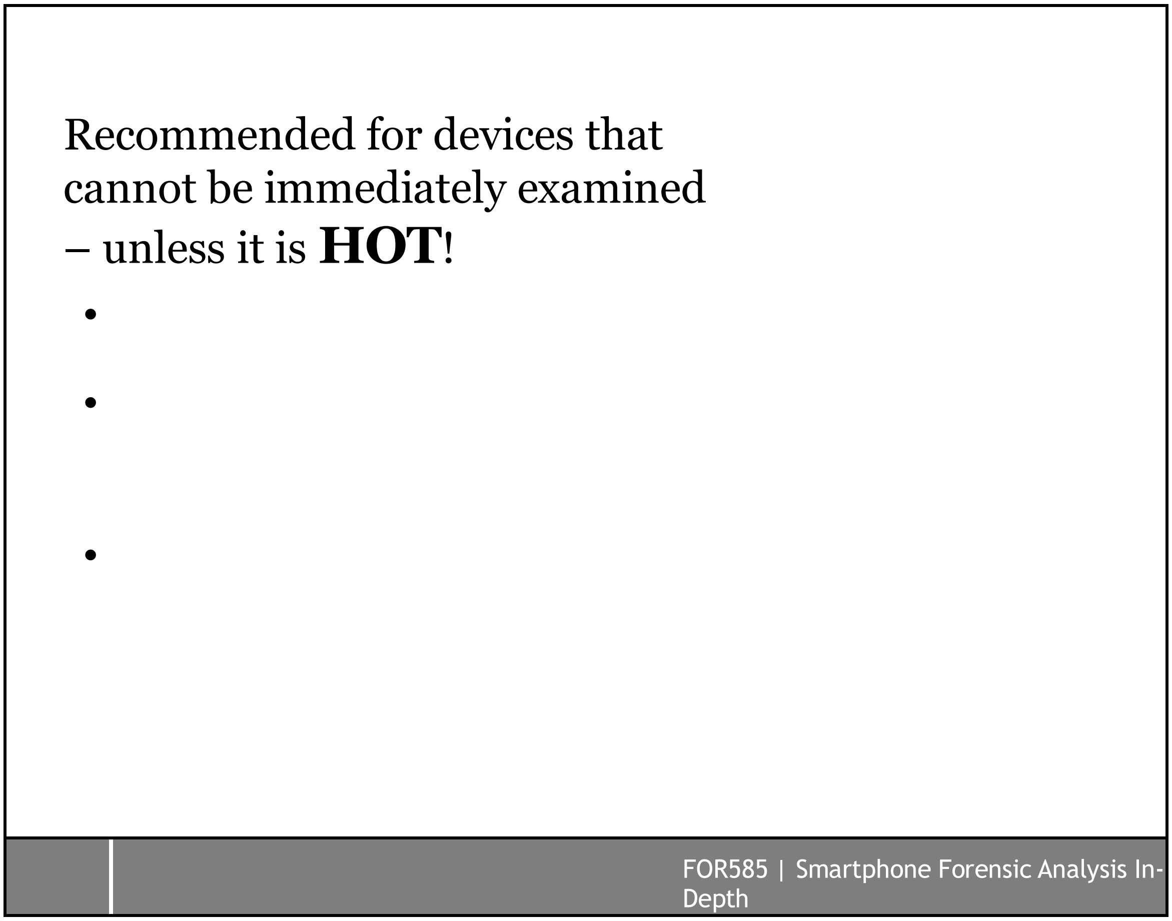
- **First method**: Always turn off the phone upon collecting it as evidence
- **Second method**: Leave phone off if it is off and leave it turned on if it is on, and use Faraday protection

Generally, there are two accepted collection methods for smartphones, both of which can have pros and cons. Knowing each method, you can decide which is most appropriate given the circumstances of your investigation.

The first collection method is to always turn off the phone upon collecting it as evidence.

The second is to leave the phone off if it is off and leave it turned on if it is on. (Use Faraday protection if you collect a phone that is powered on.)

Faraday bags are fallible, and a device inside the Faraday bag may receive signals from the surrounding mobile network. NOTE: If a charger is attached to a phone inside a Faraday bag, the charger cable may act as an antenna and can allow the phone to receive incoming data.



Recommended for devices that cannot be immediately examined – unless it is **HOT**!

FOR585 | Smartphone Forensic Analysis In-Depth

Method 1: Always Off Rule for Cell Phones[1]

This method works well if there will be a time delay in examination of the phone that could potentially result in battery discharge and data loss:

- Seize the phone and turn it off for evidence preservation
- May remove the battery to prevent phone from accidentally being turned on
- Only turn the phone on when a

trusted tool directs you to do so The benefits of turning off the phone during collection

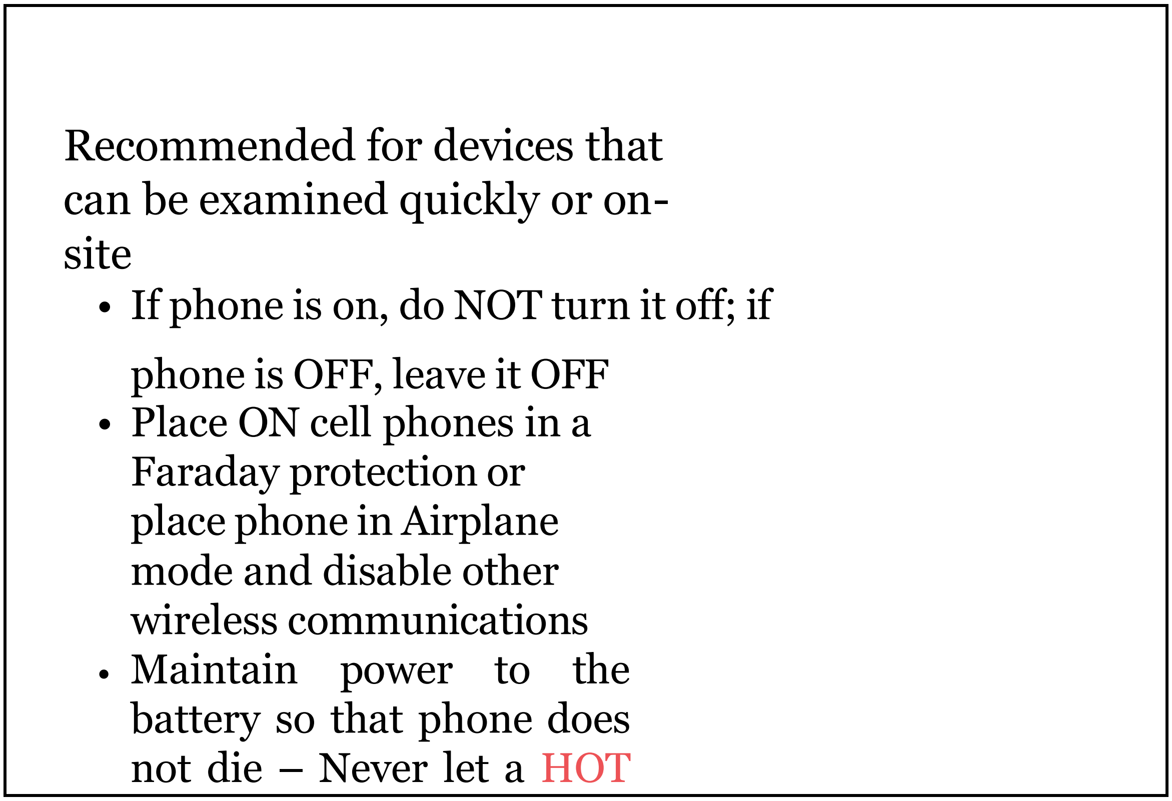
include:

- Preserving call logs and last cell tower location information (LOCI)
- Preventing overwriting deleted data
- Preventing data destruction signals from reaching the mobile phone
- Preventing improper mobile phone handling (for example placing calls, sending messages, taking photos, or deleting files)

The risks of turning off the mobile phone include possibly engaging protection mechanisms, such as encryption, passwords, PIN codes, and more.[2] Turning off the phone also results in loss of data from RAM. However, it is unlikely you will capture that data due to constraints that will be discussed later in this course. Make sure to use caution when turning off a device that was HOT (aka. Recently unlocked with a passcode) as this may be your only chance to get the most thorough dump of data from that phone.

References:

- Det. Cynthia Murphy, *Collection of Cell Phones as Evidence: Suggested Practices* (Madison, WI: Madison Police Department, 2011).
- https://for585.com/0q38a (SWGDE Best Practices for Mobile Phone Forensics)



Recommended for devices that can be examined quickly or on-site

- If phone is on, do NOT turn it off; if phone is OFF, leave it OFF
- Place ON cell phones in a Faraday protection or place phone in Airplane mode and disable other wireless communications
- Maintain power to the battery so that phone does not die – Never let a HOT

device go COLD!

The ON/OFF method works well if the phone can be examined in a timely manner. The ON/OFF rule is explained here.

If the phone is on, do NOT turn it off. If the phone is OFF, leave it OFF. Place ON cell phones in a Faraday bag, arson can, or four to five layers of aluminum foil, or place the phone in Airplane mode (if equipped) to prevent the phone from communicating wirelessly, which could potentially destroy evidence. Maintain power to the battery so that the phone doesn't die, causing potential loss of some or all data, depending on the phone model. Gather any associated cables, accessories, or documentation for the device.

Remember that turning off a password-protected cell phone could render the data on the phone inaccessible if the phone is turned off. Also, turning on a phone to go through it looking for numbers, contacts, text messages, pictures, or anything else may destroy valuable evidence.[1]

Exigency may dictate that the mobile phone remains on for immediate processing. If the mobile phone must be left on, isolate it from its network while maintaining power.[2]

- Radio Frequency (RF) shielding: Mobile phones communicate with cell towers. Allowing this communication changes data on the phone.
- Many mobile phones can be placed in Airplane mode, limiting access to cell towers (911 calls still available). This requires user input on the handset.
- Disable Wi-Fi, Bluetooth, RFID, and IrDA communications if practical.

Before beginning examination of a mobile device, to avoid legal penalty or exclusion of evidence, ensure that you have proper legal authority to perform the forensic examination.

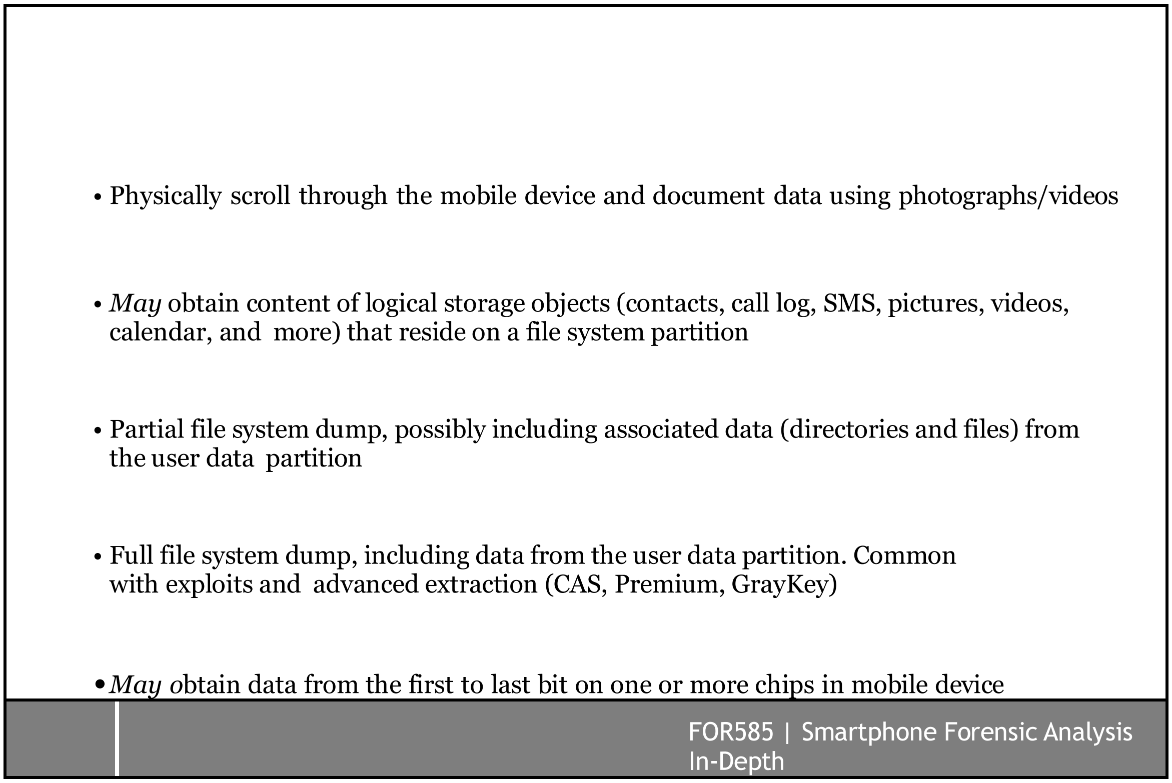
Legal authority varies by jurisdiction and can be significantly different depending on where you live and what sort of examination you are performing.

In the U.S., case law regarding mobile devices is in a state of flux. If you have any questions about whether what you are doing is legally acceptable, consult legal counsel. Search of cell phones incident to arrest has been determined to be unlawful in most circumstances by the U.S. Supreme Court (*Riley v. California*).[3]

Be particularly conscious of the fact that a mobile device is a "window" to access other user accounts where the data exists not on the device itself, but in the cloud or in an account on the internet. Legal authority to perform a search of data contained within the mobile device does NOT normally extend to data stored on the internet but accessed by the device.

References:

- Det. Cynthia Murphy, *Collection of Cell Phones as Evidence: Suggested Practices* (Madison, WI: Madison Police Department, 2011).
- https://for585.com/0q38a (SWGDE Guide to Mobile Phone Forensics)
- https://for585.com/mez8q (*Riley v. California*)



- Physically scroll through the mobile device and document data using photographs/videos
- *May* obtain content of logical storage objects (contacts, call log, SMS, pictures, videos, calendar, and more) that reside on a file system partition
- Partial file system dump, possibly including associated data (directories and files) from the user data partition
- Full file system dump, including data from the user data partition. Common with exploits and advanced extraction (CAS, Premium, GrayKey)
- *May o*btain data from the first to last bit on one or more chips in mobile device

FOR585 | Smartphone Forensic Analysis In-Depth

There are five basic levels of acquisition for smartphones, including:

- **Manual examination**: Process undertaken when the examiner physically scrolls through the mobile device and documents data using photographs or written notes regarding the contents of the device.
- **Logical acquisition**: Obtaining specific contents of logical storage objects that reside on a file system partition within a mobile device. A tool that communicates with the phone and obtains and reports only existing non-deleted contacts, call history records, SMS text messages, pictures, videos, or one or more items from the previously listed categories is conducting a logical acquisition of the data from the device.
- **File system acquisition/Advanced Logical**: Partial file system dump. Physical analyzer uses the term "Advanced Logical" for some smart devices, more commonly iOS devices. If we think about "iOS Advanced Logical" it is a partial file

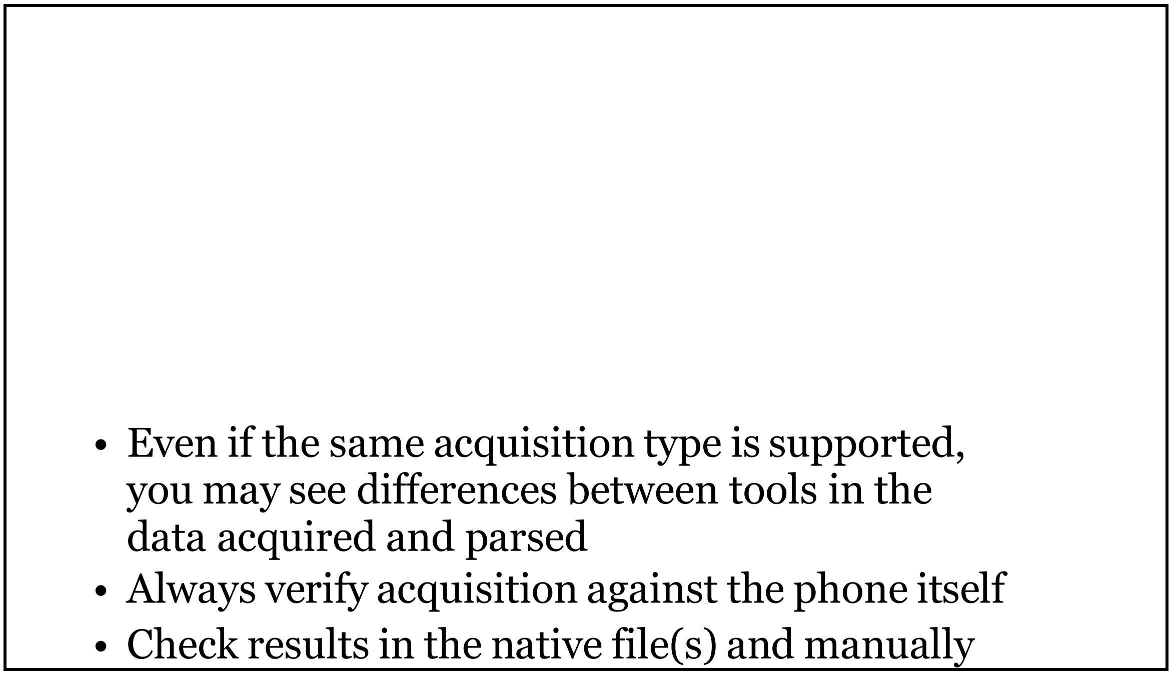
system obtained through backup + AFC, if we think about “Android Advanced Logical” it is again a partial file system obtained through backup + MTP + ADB. This will be covered in detail in each phone-specific section of the course.

- **Full file system**: Full file system dump, including data from the user data partition. Common with exploits and advanced extraction (CAS, Premium, GrayKey). This is the next best thing to a full physical acquisition.
- **Physical acquisition**: Use of the term *physical acquisition* in mobile-device forensics refers to the process of obtaining all the data from first to last bit from one or more physical stores (memory chips) in the mobile device. JTAG, ISP, and chip-off methods fall under physical acquisition. The ability to obtain this type of extraction is becoming rare as encryption mechanisms are constantly growing in strength.

When performing smartphone forensics, it is ideal to first obtain the deepest level of acquisition supported for the make and model of phone you’re working on that the tools you are using will support. Then go back and complete other supported extraction methods that may do a better job of parsing and displaying data from the phone.

For example, if you are able to obtain a physical memory dump from the phone, you get deleted information and files that are locked by the OS when they are in use. You may also be able to obtain security and unlock codes with a physical acquisition. The tool you are using, however, may or may not parse user data in an easily readable format with a physical acquisition.

Therefore, if you next perform a file system and/or logical acquisition of the data, you may obtain better results for reporting purposes. Finally, taking pictures of particularly important pieces of data from the device.



- Even if the same acquisition type is supported, you may see differences between tools in the data acquired and parsed
- Always verify acquisition against the phone itself
- Check results in the native file(s) and manually

decode to verify results if necessary

- Use more than one tool—ALWAYS!
- Know what your tools are doing – bootloaders, jailbreaks, roots

FOR585 | Smartphone Forensic Analysis In-Depth

One important thing to keep in mind when conducting mobile device forensics is that not all tools are equal, and not all smartphones are equal. Different mobile device forensic tools support various makes and models of phones to different levels of acquisition, and the way they report data back to the examiner may not be consistent. Even if the same acquisition level is supported by different tools, there may be differences between tools in the data acquired and in how it is reported. There are also sometimes errors in data translation, especially when it comes to date and timestamps because there are so many different data formats utilized by mobile devices.

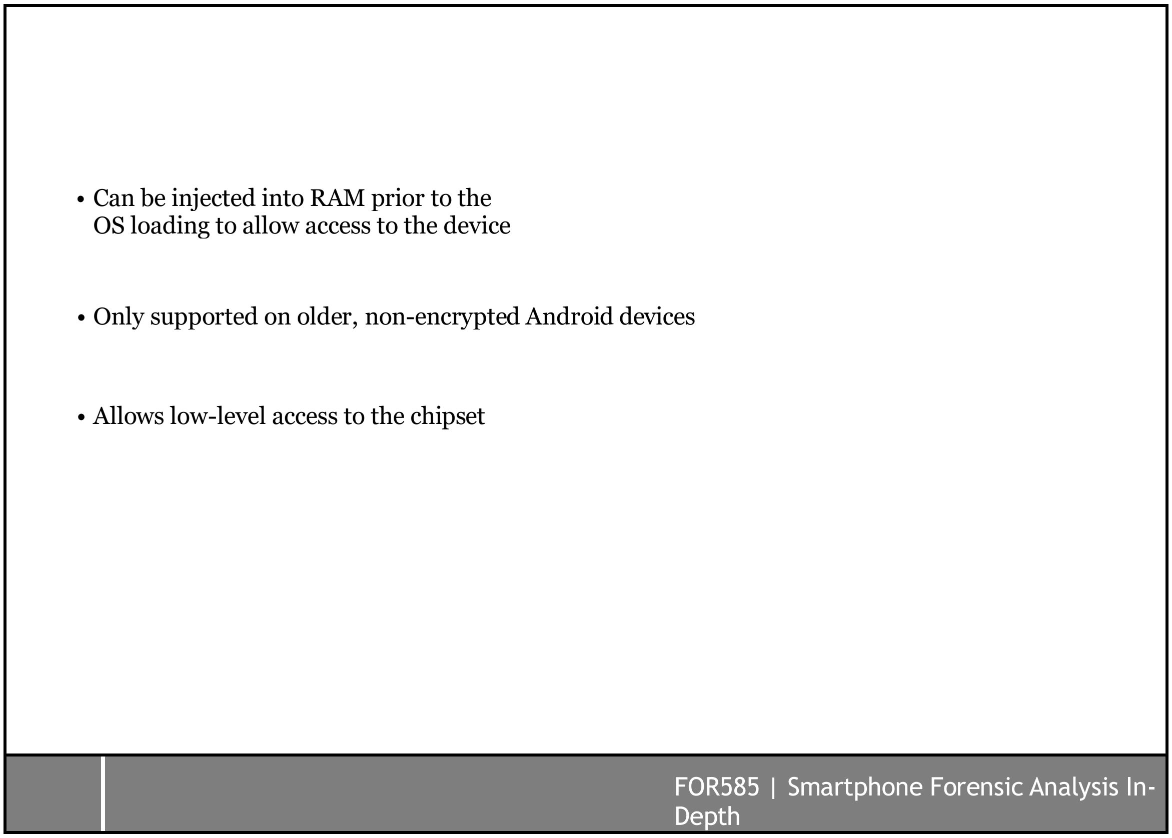
For these reasons and others, it is important that you make a habit of always verifying acquired data against the phone itself and/or manually verify decoded data by checking the underlying hex yourself.

It is important not only to know how to use the mobile forensic tools you have access to, but also to know what they are doing when you process a phone. Different mobile forensic tools use different methods to access and acquire the data from different makes and models of phones. The tool itself often makes changes to the data on the phone, including the installation of bootloaders or applications, or use of jailbreaking and rooting methods. These methods are acceptable and necessary to obtain data from the devices, but it is advisable to know what your tools are doing for each phone so you can document and explain any changes made to the device and why they were made. In this course, we will discuss new methods for access like checkra1n and checkm8. It’s important for you to understand how these exploits work and how the tools leverage them.

Another reason to know what your tools are doing is that you may find different methods to use the tool to obtain better results. For example, with Cellebrite Physical Analyzer, on many Android-based devices, using Generic Android Methods under the File System acquisition option may parse more data into usable format than a successfully supported physical acquisition of the same model of phone.

Sometimes, tools may make changes to the original device during acquisition that affect subsequent acquisition attempts. For example, when performing acquisitions on newer iPhones, Oxygen applies an

encryption password to devices that have not had a previously encrypted backup, without notification to the examiner. If the examiner attempts a subsequent acquisition using Cellebrite or another tool, they will be asked to supply the encryption password applied by Oxygen during the parsing process, though that password is not well documented. The password is “oxygen”. Cellebrite uses “1234” or “12345” and Magnet uses “mag123” when backup encryption is applied to the extraction. This information isn’t readily available in the documentation, and Elcomsoft Phone Breaker was used to solve the puzzle of what the password was; keep in mind this may happen to you and you may find yourself having to crack a password you and the user did not set on the device.



- Can be injected into RAM prior to the OS loading to allow access to the device
- Only supported on older, non-encrypted Android devices
- Allows low-level access to the chipset

FOR585 | Smartphone Forensic Analysis In-Depth

For older Android devices, we can perform full physical extractions of data using a small program or set of instructions called a “bootloader”.[1], The bootloader is injected into the RAM of the device and executes before the operating system boots. Bootloaders are usually a generic solution, based upon a family of devices, often defined by the chipset in the phone. They are designed to be read-only and are therefore considered safe and forensically sound. Bootloaders allow for complete acquisition of flash memory of a mobile device, including spare area of the flash memory.[1]

Manufacturers of cell phone forensic tools will often create custom bootloaders for full physical acquisition purposes. In order to use a bootloader for acquisition, sometimes the phone must be put into rescue or recovery mode, or a specialized cable might be necessary.[2] Sometimes firmware update protocols are used to insert a bootloader.

When a device is locked, it may be possible to use Android's recovery partition to access the data on the device. The Android device needs to be placed into recovery mode so the custom recovery image can be pushed to the device. A custom recovery is a third-party image that replaces the stock Android recovery partition and allows the user to gain access to the data or root the device. Again, this is only supported for older, non-encrypted Android devices. A common custom recovery partition is Team Win Recovery Project (TWRP). The Magnet Recovery Images are a great reference for researching custom recovery partitions: https://www.magnetforensics.com/resources/advancedmobile/.

EDL mode is available with some Qualcomm chips and can allow low-level access to the chipset. This is designed to allow for device analysis, repair, or re-flashing, but can present the opportunity to achieve a full physical extraction of the data from the chip.[3] EDL mode can be accessed by several software methods, including:

- Special key combinations: Depending on the manufacturer, the key combination may be different. The most common combination from a powered-off state is: Hold Vol Up + Vol Down while connecting USB. Other combinations may include Vol Up, Vol Up + Vol Down + Power, etc. Some vendors have early boot menus that offer the choice of entering the mode (recovery, fastboot, download).
- ADB: Most phones with EDL available allow entry to EDL mode via a command available from an authorized ADB session. (Try: adb reboot edl.) This is useful for obtaining a physical extraction of an *unlocked* device. This is what Cellebrite UFED attempts when trying "Generic Qualcomm ADB".
- Fastboot: An alternative vendor-specific method exists from the fastboot mode, which is sometimes reachable by other key combinations (usually Vol Down + Power).
- FTM: Some vendors have implemented FTM mode (hold Vol Down while connecting the USB), which exposes an ADB interface. Cellebrite UFED can detect this mode and continue to extract normally using the "Generic Qualcomm ADB" method.

Hardware methods may include:

- EDL cable: Some devices will detect a special cable that will signal the device to enter EDL. Specialized cables may be obtained from various stores and will be supplied by Cellebrite to customers when the phone is supported.
- Test points: Some devices have test points that, when shorted to ground, will cause the device to enter into EDL mode. Depending on the board, they may be easily accessible, even without significant disassembly.
- eMMC faults: Electrical faults are introduced to the eMMC chip as it boots. With a pinout chart for the specific board in the device, you may short the CMD, CLK, or D0 lines to ground temporarily during power on.

References:

- https://for585.com/e0d7p (Cellebrite white paper: "What Happens When You Press That Button?")
- https://for585.com/cnosp (Magnet Forensics: "White Paper: Android Acquisition Methods from Root to Recovery")
- https://for585.com/a4p61 (Cellebrite Practical Guide: Qualcomm EDL Extractions)

Cellebrite offers additional methods to acquire Android devices that are newer and otherwise inaccessible. These include Qualcomm Live, MTK Live and the latest capability referred to as Huawei (Kirin Live). These extraction methods are linked to the chip inside of the Android and are accessible via UFED.

The Joint Test Action Group (JTAG) method used to be heavily relied upon when forensic and open-source tools did not support data acquisition of a smartphone. Currently, many devices do not support this method of acquisition. Encryption and lack of TAPs are to blame for this lost craft. This type of acquisition can damage a device if done by an untrained examiner. The JTAG method acquires data via the Test Access Ports (TAPs), which requires the device be taken apart, yet remain functional. (For example, the device must be able to be powered and recognized by the forensic workstation). Thus, if an examiner is untrained, the device may become damaged during disassembly or soldering to the TAPs, which renders the evidence unusable. Use caution when exploring this method and make sure you have proper training, and practice on test devices prior to attempting JTAG on real evidence.

When done correctly, JTAG is a great way to recover data from devices. Prepaid/throw-away/burner phones present challenges during examinations because they are usually difficult to acquire. The data ports have often been disabled to prevent individuals from re-flashing the firmware on these inexpensive phones and reselling them for use with a different carrier for a much higher price. These devices are generally good use cases for JTAG, ISP, or chip-off techniques.

ISP, or In-System Programming, is a method that enables examiners to extract data physically by connecting to the chip on the Printed Circuit Board containing phone data to obtain a full physical extraction. This method does not require the removal of the chip and is comparable to JTAG in that aspect.

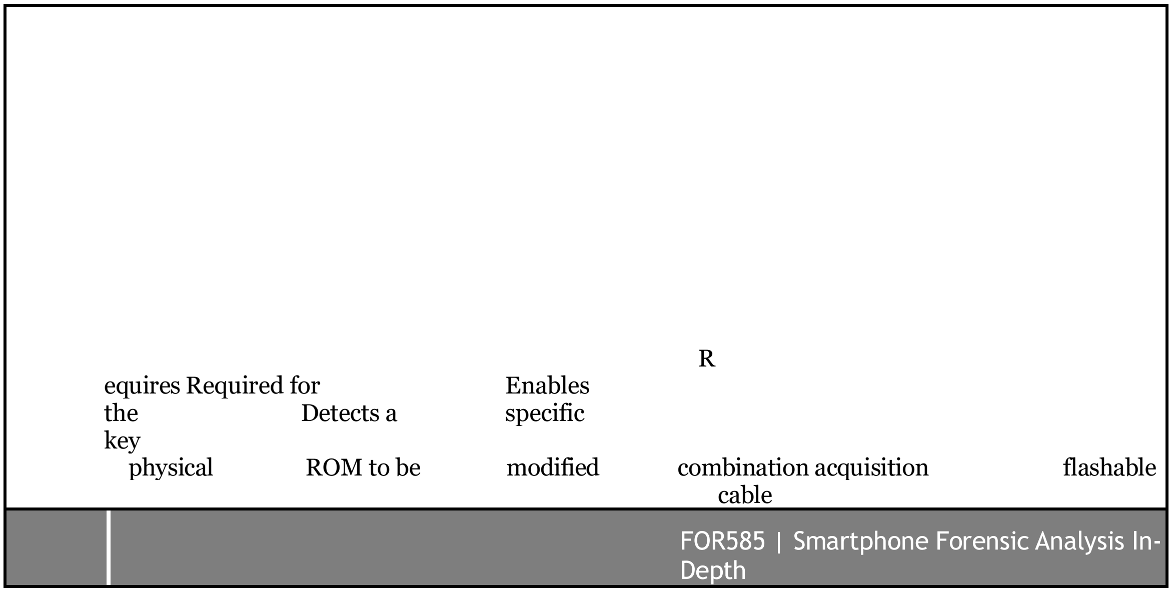
Chip-off forensics is a data acquisition procedure that involves physically removing the non-volatile integrated circuit (IC) chip from a device and reading it directly on a specialized external reader designed specifically to read the make or model of chip being worked with. A complete bit-for-bit forensic image file is made from the data contained in the removed chip, which is read directly from the chip, and the data is not changed during the imaging process.[2] Use caution when exploring this method and make sure you have proper training, equipment, and practice on equivalent test devices prior to attempting chip-off on real evidence.

Chip-off data extraction is a forensically sound way to recover data from devices and may be the only option in some circumstances when phones are not supported for data extraction due to locked data ports, unsupported operating systems, encryption, or physical damage to the phone. In cases where a mobile forensic tool may support a full physical image of a device for decoding and parsing but cannot obtain a full physical image due to passcode protection, chip-off can be used to extract data from the device, and scripts can then be used to deal with passcodes or swipe codes once the data is loaded into the forensic tool. Chip-off is usually considered the last option because the removal of the chip is generally a one-way operation, and the original hardware will be damaged during the process. However, in modern-day forensics, this may be the only option, so it is important that you are aware of the possibilities.

The chip is taken off by removing the epoxy that surrounds it and then heating up the solder, allowing the chip to be removed from the Printed Circuit Board (PCB). The other option is to grind the chip away from the board. After the chip is removed from the phone, it will need to be cleaned up to remove remaining solder and residue so that it can be read. The chip may also need to be “re-balled” in order to repair contacts damaged during removal. The chip will then be placed in a reader or adapter of the exact size and pin out. Many adapters are custom made to accommodate various proprietary chips. The NAND chips follow the specifications from the Open NAND Flash Interface (ONFI) group, and while the pin out will remain the same for most chips, there may be different package sizes. Once the correct reader or adapter is found, the chip can be mounted, and a full physical image of the chip can be made.[3]

References:

- Mahalik, Tamma, and Bommisetty, "JTAG defined", in *Practical Mobile Forensics*, Second Edition (Birmingham, UK: Packt, 2016).
- https://for585.com/o19f6 (Chip-Off and JTAG Analysis)
- https://for585.com/ox4u5 (Open NAND Flash Interface)



R
equires Required for Enables
the Detects a specific
key
physical ROM to be modified combination acquisition flashable
cable

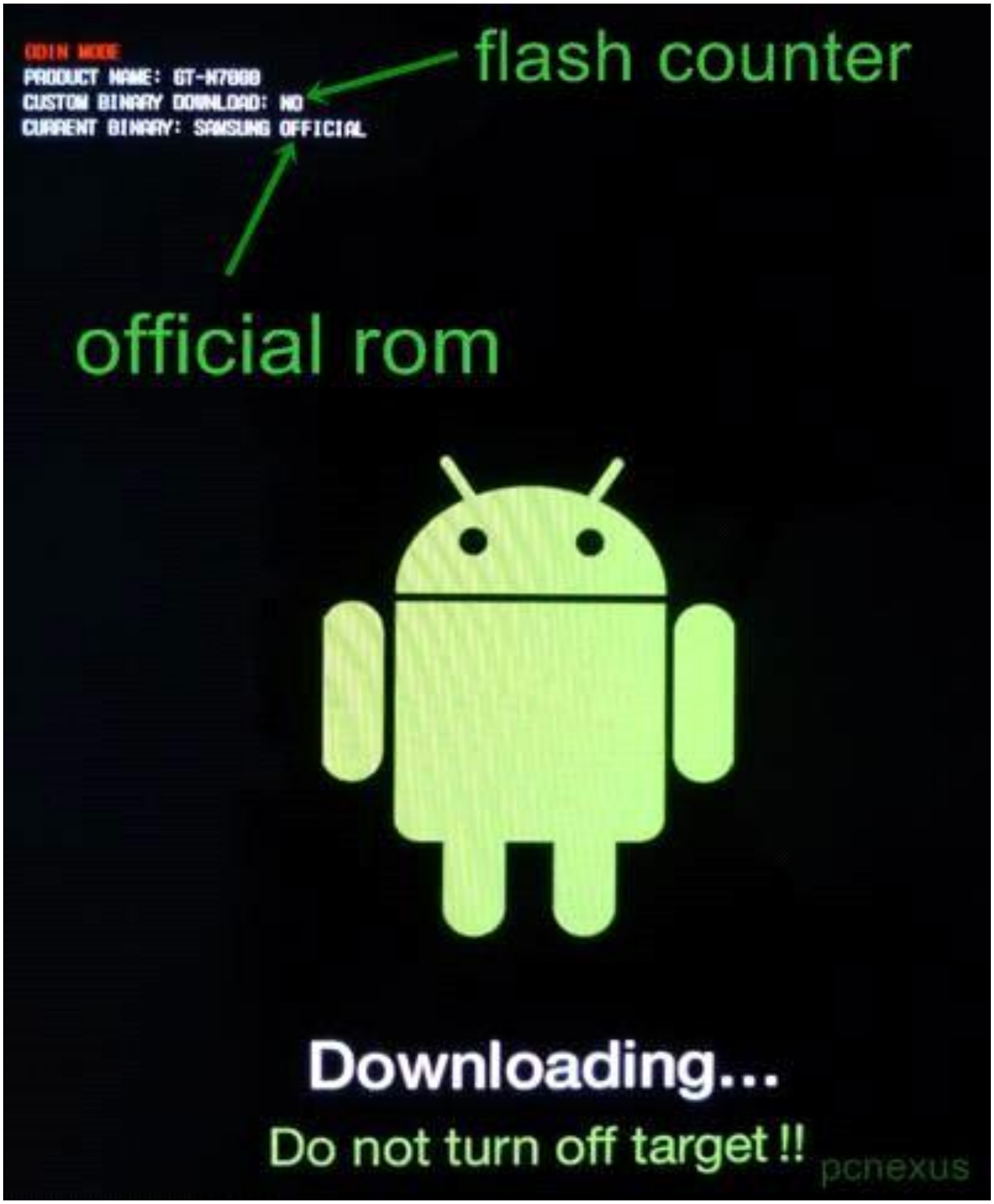
FOR585 | Smartphone Forensic Analysis In-Depth

Most commercial tools that attempt to physically acquire an Android device require that the device enter a mode referred to as Download Mode. Download Mode allows the user to flash the ROM of the device, which then allows the tool to create a forensic image. Flashing the ROM leaves traces on the device. When the custom ROM of the device is modified, it's tracked within the device. When the device enters Download Mode, the number of flash counts (ROM changes) are listed.

To enter Download Mode, a specific cable or key combination may be required. Each device may have a unique key combination required. The easiest way to get into Download Mode is to follow the instructions provided by the forensic tool.

The screenshot shows a device that has the original, custom ROM and has not been modified by a forensic tool or user. This state is represented by the picture below. If a device ROM had been modified, the CUSTOM BINARY DOWNLOAD indicator would say YES or retain the count for the number of times it was flashed, and the CURRENT BINARY may reference something like CUSTOM (or anything other than the OFFICIAL binary that shipped with the device). Keep in mind, your forensic tools make these modifications. If you are conducting covert operations, make sure that you pay attention to what you leave behind!

NOTE: Not all devices can enter Download Mode. We recommend researching your specific device if you are having a hard time entering Download Mode.

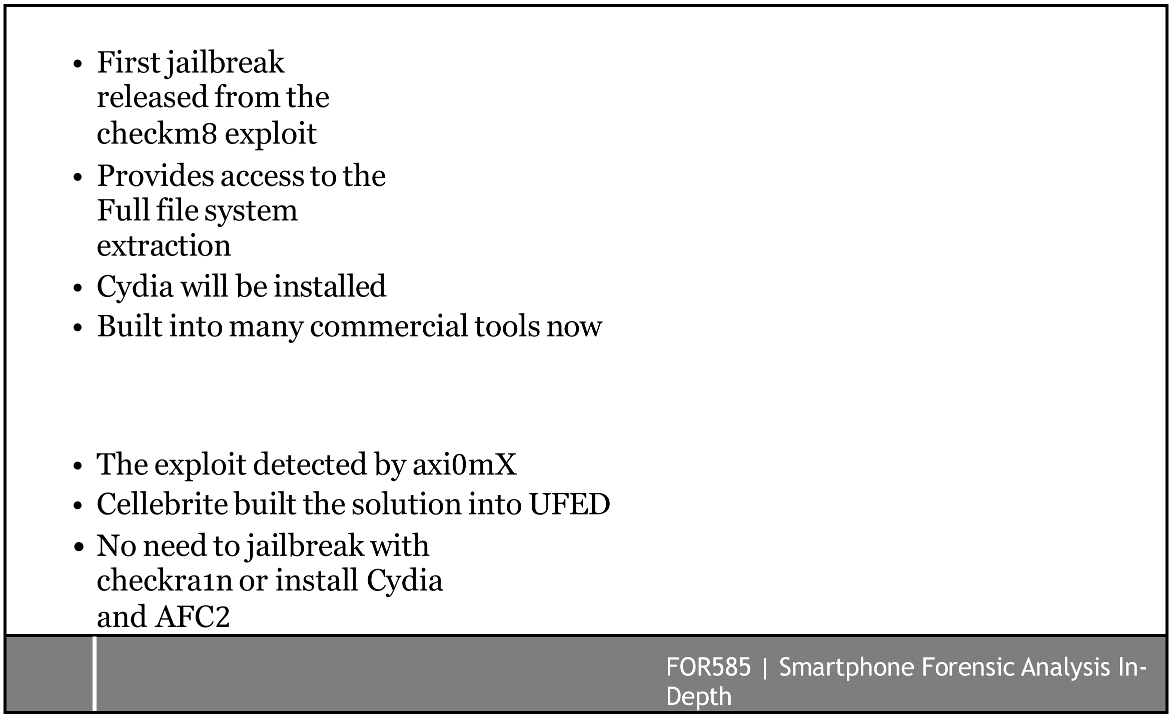


```
ODIN MODE
PRODUCT NAME: GT-N7000
CUSTOM BINARY DOWNLOAD: NO
CURRENT BINARY: SAMSUNG OFFICIAL
```

Downloading...

Do not turn off target !!

Graphic from https://for585.com/flash.

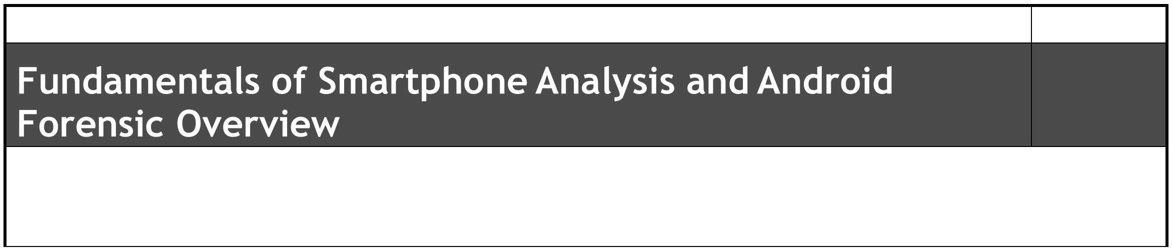


- First jailbreak released from the checkm8 exploit
- Provides access to the Full file system extraction
- Cydia will be installed
- Built into many commercial tools now

- The exploit detected by axi0mX
- Cellebrite built the solution into UFED
- No need to jailbreak with checkra1n or install Cydia and AFC2

The discovery of the checkm8 exploit on Apple devices (4S – X and several iPads) by axi0mX opened a door to access most haven't seen in almost a decade. We can now extract a Full file system extraction from iOS devices affected by this exploit. The best news for us in forensics: It cannot be patched as it's an exploit that lives on the chips inside of these iOS devices. At the time of this update, statistics show that 85% of devices currently in use fall into the realm of being susceptible to the checkm8 exploit. Now, all we needed was a jailbreak.

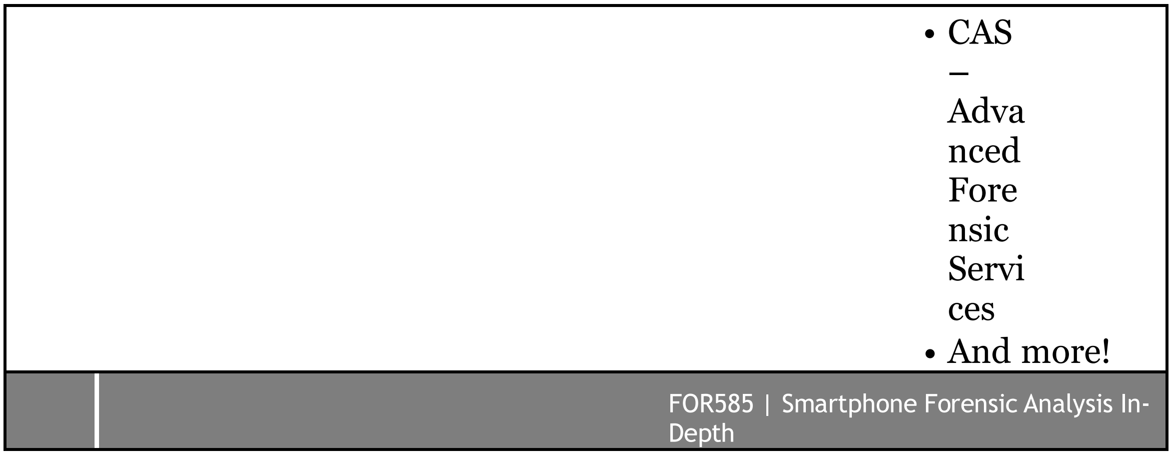
Enter checkra1n, the first public jailbreak that utilizes the checkm8 exploit. Cellebrite was the first to implement the capability into their tools. You can leverage UFED to obtain a Full file system extraction via checkm8 without manually jailbreaking the device. This method is deemed safer than manually using checkra1n because the exploit and jailbreak run in RAM and do not permanently change the file system. Since the release of checkm8 in UFED, many other vendors have implemented similar methods. More on checkm8 and checkra1n will be covered in section 3.



Fundamentals of Smartphone Analysis and Android Forensic Overview

This page intentionally left blank.

- Physical Analyzer
- Cloud Analyzer
- Pathfinder
- UFED
- In-Field
- Reader
- Premium/ES



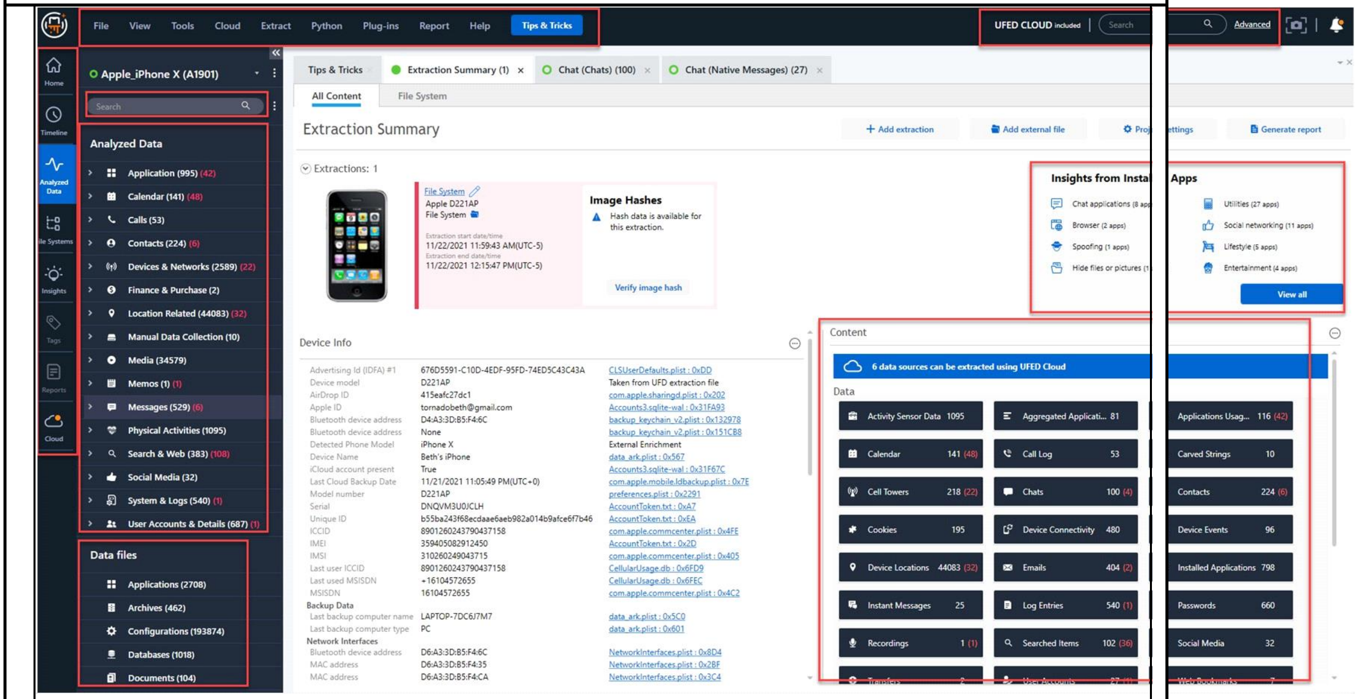
- CAS – Advanced Forensic Services
- And more!

FOR585 | Smartphone Forensic Analysis In-Depth

Cellebrite's mobile forensic solutions include a combination of hardware, software, and services designed for extracting and analyzing data from mobile devices. These tools are some of the most popular in mobile device forensics. While the views in Physical Analyzer may change as updates are released, the concepts covered in this course will be the same. Cellebrite is strong at application parsing, timelining and carving additional artifacts. Physical Analyzer is a great platform for conducting analysis as it provides support for almost all mobile platforms and shows you both the phone extraction image file and the file system for examiners to dig into data that isn’t parsed.

Cellebrite’s hardware and software package solutions are summarized below. While there may be more available, these are the ones we deemed relevant when writing the course.

- **UFED4PC** – Acquisition tool: The “Ultimate” version features both logical and physical support. This is where the checkm8 capability exists. The logical version features simply logical support. UFED4PC offers a version of UFED that is installed on your forensic workstation.
- **UFED Touch 2** – Acquisition tool: The portable version. This can be ruggedized for extreme conditions or in a traditional format. The features are the same as UFED4PC depending on if you have “Ultimate” or logical. This is also where the checkm8 capability exists.
- **Physical Analyzer** – The platform for examiners to conduct analysis. Features cloud and may feature Analytics in the future.
- **Pathfinder (Analytics)** – Platform for conducing OCR searches, redacting images/videos, link analysis, correlation, similar images, image recognition and more.
- **Reader** – The free platform for investigative review.
- **Premium** – LAW ENFORCEMENT ONLY (unlock access) acquisition tool: Additional access to Android and iOS devices for acquisition and unlocking – reach out to Cellebrite for more info, as we do not have it.
- **Guardian** – The case management system
- **Inspector** – Digital forensic platform for analysis – previously BlackLight
- **Digital Collector** – Remote collection tool for PC and Mac acquisitions – previously macquisition
- **CAS – Cellebrite Advanced Services** – The specialized services available to anyone with just cause and proper paperwork to unlock and extract data from smartphones.



Extraction Summary

Extractions: 1

File System
Apple D221AP
File System

Extraction start date/time
11/22/2021 11:59:43 AM(UTC-5)
Extraction end date/time
11/22/2021 12:15:47 PM(UTC-5)

Image Hashes

Hash data is available for this extraction.

Verify image hash

Device Info

Advertising Id (IDFA) #1	676D5591-C10D-4EDF-95FD-74ED5C43C43A	CLSUserDefaults.plist : 0xDD
Device model	D221AP	Taken from UFD extraction file
AirDrop ID	415eafc27dc1	com.apple.sharingd.plist : 0x202
Apple ID	tornadobeth@gmail.com	Accounts3.sqlite-wal : 0x31FA93
Bluetooth device address	D4:A3:3D:B5:F4:6C	backup_keychain_v2.plist : 0x132978
Bluetooth device address	None	backup_keychain_v2.plist : 0x151CB8
Detected Phone Model	iPhone X	External Enrichment
Device Name	Beth's iPhone	data_ark.plist : 0x567
iCloud account present	True	Accounts3.sqlite-wal : 0x31F67C
Last Cloud Backup Date	11/21/2021 11:05:49 PM(UTC+0)	com.apple.mobile.ldbackup.plist : 0x7E
Model number	D221AP	preferences.plist : 0x2291
Serial	DNQVM3U0JCLH	AccountToken.txt : 0xA7
Unique ID	b55ba243f68ecdaae6aeb982a014b9afce6f7b46	AccountToken.txt : 0xEA
ICCID	8901260243790437158	com.apple.commcenter.plist : 0x4FE
IMEI	359405082912450	AccountToken.txt : 0x2D
IMSI	310260249043715	com.apple.commcenter.plist : 0x405
Last user ICCID	8901260243790437158	CellularUsage.db : 0x6FD9
Last used MSISDN	+16104572655	CellularUsage.db : 0x6FEC
MSISDN	16104572655	com.apple.commcenter.plist : 0x4C2
Backup Data		
Last backup computer name	LAPTOP-7DC6J7M7	data_ark.plist : 0x5C0
Last backup computer type	PC	data_ark.plist : 0x601
Network Interfaces		
Bluetooth device address	D6:A3:3D:B5:F4:6C	NetworkInterfaces.plist : 0x8D4
MAC address	D6:A3:3D:B5:F4:35	NetworkInterfaces.plist : 0x2BF
MAC address	D6:A3:3D:B5:F4:CA	NetworkInterfaces.plist : 0x3C4

Your instructor will show you how to properly navigate the options in Physical Analyzer. For those new to this tool, the more you use it, the easier it becomes. The key areas highlighted are the toolbar, where many features are accessible and will be shown in the next few slides. A logical keyword search or an Advanced Search at the top right, which lets you search generically for a keyword or as an "and" or "or" request. This also includes the ability to conduct an "in content" search which takes some time but will dive into files that are not parsed by the tool.

A good place to start is the Extraction Summary, which is just that – a summary of the files parsed from the extraction. From there, most go to the Analyzed Data section of the tree pane on the left and let the investigation take it away. The Insights from Installed Apps are something you should consider for triage and then a deep dive, if relevant. Throughout this week, you will have an opportunity to use Physical Analyzer in many labs. All the intricacies will appear in the labs and the reviews.

The Tools menu in Physical Analyzer allows the user to access carving functions, and specialty tools within the software, such as the AppGenie, fuzzy model plugin, enrichments (media categorization) the ability to create dictionary files, Watch List Editor, and Malware Scanner. A watch list is a list of keywords that are created by the examiner and are used to search for and identify items of interest in the extracted data.

Specialized watch lists can be developed and run across multiple extractions. The malware scanner is a signature-based scanner that searches data extractions from mobile devices for known malware. Be sure to update the malware signature database before running the malware scanner so that you are using the most recent definitions. More will be covered on this in Section 4. The fuzzy model plug-in will help identify and parse databases not already analyzed by Cellebrite. The AppGenie digs for additional application artifacts. Together, these two carvers are powerful! The results from AppGenie and Fuzzy Model Plugin will be in the Analyzed Data section under **Manual Data Collection.**

The Settings and Project settings are important and are covered below. Make sure you enable the Deep Carving for SQLite and other features discussed by your instructor, so you get the same lab results as explained in the eWorkbook.

The Project settings are where you set the time zone settings. When determining what to select for parsing, we recommend customizing your options. In a lab environment, where time is not a concern, you should select all checkboxes under **Decoding.** For this class, we want you to check the following boxes:

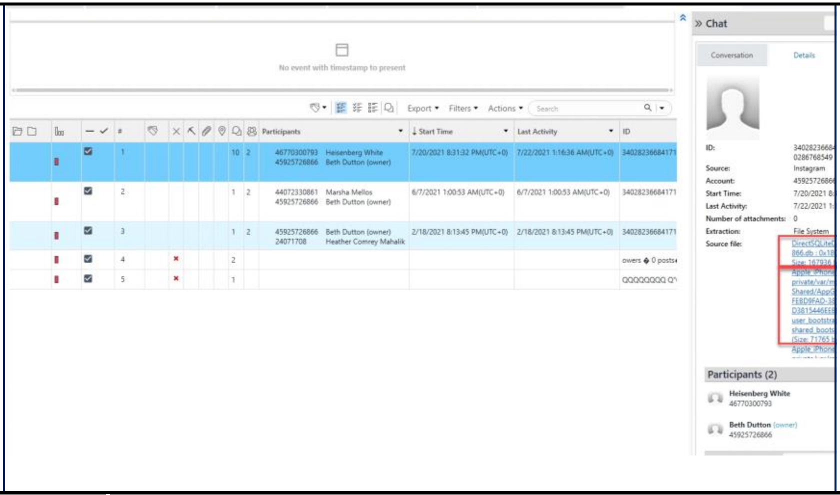
- Recover deleted data for Android and Windows Phone devices via carving from unallocated space
- Use deep carving for SQLite
- Recover data from archive files

Under **Timeline,** it is pertinent that you select all boxes. Otherwise, you may see gaps in your timeline for multimedia files.

Installed applications can be viewed in the **Insights** view or within the **Table View** in Physical Analyzer. The **Table View** is probably what most are used to for those who have used Physical Analyzer in the past. Here, we can tell how the app was parsed (Decoded By) if it was deleted (the pickaxe) and the category it falls under. More importantly, it shows the **source**, which is the fastest way to hop directly into the file system. The App Genie, as previously mentioned, is a tool that tried to identify databases with artifacts of interest like chats, contacts, and more. Cellebrite recommends using these results for **triage** and then manually review them to ensure they are legitimate prior to putting them in your forensic report – that concept is the theme behind this class

The **Insights** view can be used to view the chunks of categorized apps. Under utilities, we see the groups of application categories and the files associated with them. From here, one can launch AppGenie (if not previously run), run the Android Emulator or launch the SQLite Wizard.

- Source files provide a direct hop to the file system
- File system is available in the left tree pane



No event with timestamp to present

✓	#	×			Participants		↓ Start Time	Last Activity	ID
☑	1		10	2	46770300793 45925726866	Heisenberg White Beth Dutton (owner)	7/20/2021 8:31:32 PM(UTC+0)	7/22/2021 1:16:36 AM(UTC+0)	34028236684171
☑	2		1	2	44072330861 45925726866	Marsha Mellos Beth Dutton (owner)	6/7/2021 1:00:53 AM(UTC+0)	6/7/2021 1:00:53 AM(UTC+0)	34028236684171
☑	3		1	2	45925726866 24071708	Beth Dutton (owner) Heather Comrey Mahalik	2/18/2021 8:13:45 PM(UTC+0)	2/18/2021 8:13:45 PM(UTC+0)	34028236684171
☑	4	×	2						owers ◆ 0 posts
☑	5	×	1						QQQQQQQQ Q'

» Chat

ID:	3402823668 0286768549
Source:	Instagram
Account:	4592572686
Start Time:	7/20/2021 8:
Last Activity:	7/22/2021 1:
Number of attachments:	0
Extraction:	File System
Source file:	DirectSQLite 866.db : 0x18 Size: 167936 Apple_iPhone private/var/m Shared/AppG FE8D9FAD-38 D3815446EE user_bootstra shared_boots (Size: 71765 b Apple_iPhone

Participants (2)

Heisenberg White
46770300793

Beth Dutton (owner)
45925726866

The file system is easily accessible in Physical Analyzer as it's right in the left pane of the primary view and remains there as you navigate within the tool. The easiest way to locate files of interest in the file system is to click on the source file for the artifact of interest. It will take you directly to the file. If you land in the Hex view, simply change it to the view of your choice by selecting your tab of interest.

Some of the key features of Physical Analyzer are highlighted above. The DB Viewer is where you can carve for deleted artifacts and BLOBs hidden inside of databases while the SQLite Wizard is where you can draft queries. There are built in viewers for plists, xml and protobuf files. Data carving is available upon parsing and after loading the data. Keep in mind that some carving features must occur while parsing. The timeline view is useful when trying to decide when an event occurred. Finally, the “Go to” lets you jump from artifact results straight to the timeline.

The View menu option allows the user to easily navigate to the welcome screen to navigate between projects. The View menu also allows the user to open a Trace Window, which opens a log of all the operations and actions performed by the software on the data extraction so that the examiner can know what is going on behind the scenes.

The **Plug-ins** menu option allows the user to **Add/Remove Plug-ins** by displaying a list of pre-installed plug- ins and allowing the user to manage them by adding or removing them from the list. Also in the Plug-ins menu is the **Run Plug-in** option, which enables the examiner to select and run one or more specific pre-installed plug-ins against the active data. The **Chain Manager** option in the Plug-ins menu displays the Chain Manager window and allows the user to create and edit device-processing chains, which essentially consist of multiple plug-ins run in a determined order.

There are several keyword searching options available in Physical Analyzer. The first is the logical search, which simply searches for files parsed or file names in the parsed file system, data file and analyzed data. The next search is an **Advanced** Search, which is shown above. To get to this search, you simply select **Advanced** by the logical search window for All Projects. One can search for a keyword or as an "and" or "or" request as shown in the slide.

FOR585 | Smartphone Forensic Analysis In-Depth

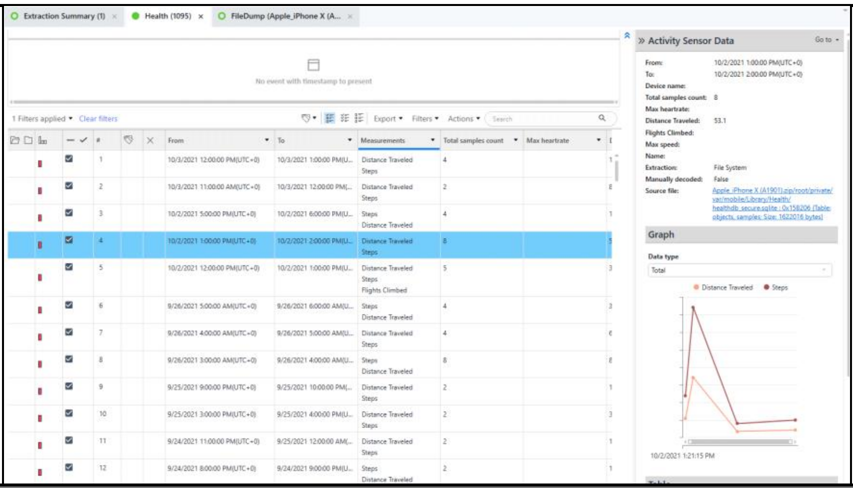
The final and most thorough search is a physical search in the Hex. This is done by selecting the magnifying glass, using the dropdown (strings shown here), and then ticking the box for the encoding schema. Individual searches can be performed on each section of data, including searching for text, phone numbers, emails, geocoordinates, IP addresses, MAC addresses, and credit card numbers; also, a regular expressions search. Data can be searched from a single device, all devices involved in the case, or all acquired devices. Physical Analyzer is the most thorough at physically searching data at a physical level for mobile devices.

You can toggle between parsed data in the Analyzed Data section and the device image

File

Dump of data

Analyzed Data



#	From	To	Measurements	Total samples count	Max heartrate	
1	10/3/2021 12:00:00 PM(UTC+0)	10/3/2021 1:00:00 PM(U...	Distance Traveled Steps	4		1
2	10/3/2021 11:00:00 AM(UTC+0)	10/3/2021 12:00:00 PM(...	Distance Traveled Steps	2		8
3	10/2/2021 5:00:00 PM(UTC+0)	10/2/2021 6:00:00 PM(U...	Steps Distance Traveled	4		1
4	10/2/2021 1:00:00 PM(UTC+0)	10/2/2021 2:00:00 PM(U...	Distance Traveled Steps	8		5
5	10/2/2021 12:00:00 PM(UTC+0)	10/2/2021 1:00:00 PM(U...	Distance Traveled Steps Flights Climbed	5		3
6	9/26/2021 5:00:00 AM(UTC+0)	9/26/2021 6:00:00 AM(U...	Steps Distance Traveled	4		3
7	9/26/2021 4:00:00 AM(UTC+0)	9/26/2021 5:00:00 AM(U...	Distance Traveled Steps	4		6
8	9/26/2021 3:00:00 AM(UTC+0)	9/26/2021 4:00:00 AM(U...	Steps Distance Traveled	8		8
9	9/25/2021 9:00:00 PM(UTC+0)	9/25/2021 10:00:00 PM(...	Distance Traveled Steps	2		1
10	9/25/2021 3:00:00 PM(UTC+0)	9/25/2021 4:00:00 PM(U...	Distance Traveled Steps	2		3
11	9/24/2021 11:00:00 PM(UTC+0)	9/25/2021 12:00:00 AM(...	Distance Traveled Steps	2		1
12	9/24/2021 8:00:00 PM(UTC+0)	9/24/2021 9:00:00 PM(U...	Steps Distance Traveled	2		1

Activity Sensor Data

Go to

From: 10/2/2021 1:00:00 PM(UTC+0)
To: 10/2/2021 2:00:00 PM(UTC+0)
Device name:
Total samples count: 8
Max heartrate:
Distance Traveled: 53.1
Flights Climbed:
Max speed:
Name:
Extraction: File System
Manually decoded: False
Source file: Apple_iPhone X (A1901).zip/root/private/var/mobile/Library/Health/healthdb_secure.sqlite : 0x158206 (Table: objects, samples; Size: 1622016 bytes)

Graph

Data type: Total

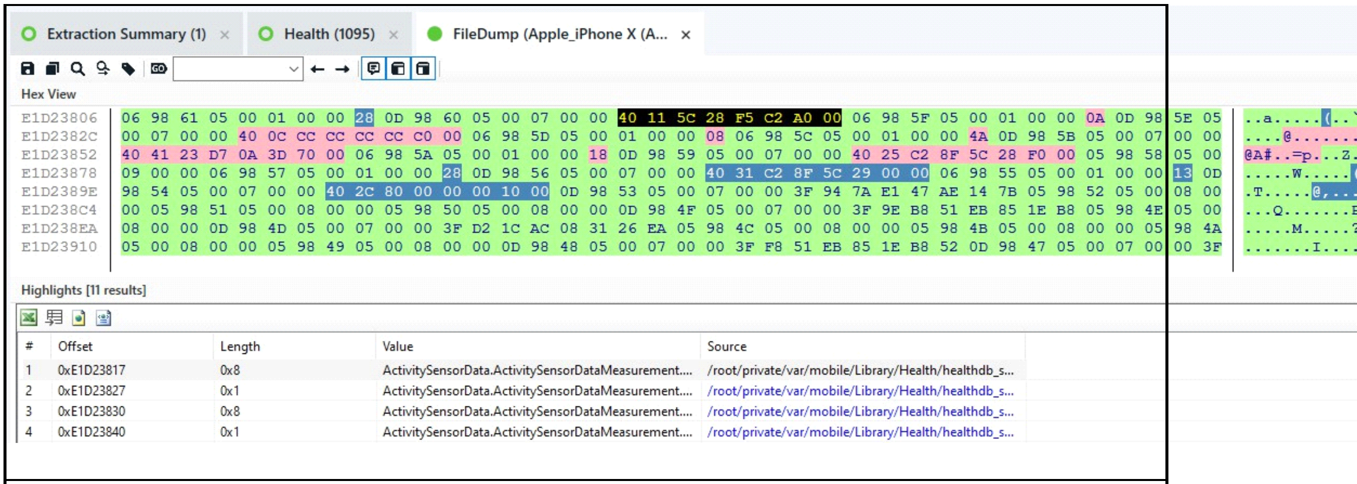
It is possible to navigate between an entry from the Analyzed Data section in Physical Analyzer and the source of that data (offset) in the original image. To do so, you must have both the tab for the associated analyzed data (in this case, “Health”) and the tab containing the data from image or extraction that the data was parsed from open, as noted by the arrows.

To navigate to the location of the parsed data within the image that contains the analyzed data, simply highlight the data in the analyzed data tab and switch tabs to the image tab. The cursor is adjacent to the data that was shown, as parsed in the analyzed data tab. This is shown in the next slide.

You can toggle between parsed data in the Analyzed Data section and the device image

Analyzed Data

Offset of Data in File Dump



```
E1D23806 | 06 98 61 05 00 01 00 00 28 0D 98 60 05 00 07 00 00 40 11 5C 28 F5 C2 A0 00 06 98 5F 05 00 01 00 00 0A 0D 98 5E 05 | ..a.....(..
E1D2382C | 00 07 00 00 40 0C CC CC CC CC C0 00 06 98 5D 05 00 01 00 00 08 06 98 5C 05 00 01 00 00 4A 0D 98 5B 05 00 07 00 00 | ....@.......
E1D23852 | 40 41 23 D7 0A 3D 70 00 06 98 5A 05 00 01 00 00 18 0D 98 59 05 00 07 00 00 40 25 C2 8F 5C 28 F0 00 05 98 58 05 00 | @A#..=p...Z.
E1D23878 | 09 00 00 06 98 57 05 00 01 00 00 28 0D 98 56 05 00 07 00 00 40 31 C2 8F 5C 29 00 00 06 98 55 05 00 01 00 00 13 0D | .....W.....
E1D2389E | 98 54 05 00 07 00 00 40 2C 80 00 00 00 10 00 0D 98 53 05 00 07 00 00 3F 94 7A E1 47 AE 14 7B 05 98 52 05 00 08 00 | .T.....@,...
E1D238C4 | 00 05 98 51 05 00 08 00 00 05 98 50 05 00 08 00 00 0D 98 4F 05 00 07 00 00 3F 9E B8 51 EB 85 1E B8 05 98 4E 05 00 | ...Q.......
E1D238EA | 08 00 00 0D 98 4D 05 00 07 00 00 3F D2 1C AC 08 31 26 EA 05 98 4C 05 00 08 00 00 05 98 4B 05 00 08 00 00 05 98 4A | .....M.....
E1D23910 | 05 00 08 00 00 05 98 49 05 00 08 00 00 0D 98 48 05 00 07 00 00 3F F8 51 EB 85 1E B8 52 0D 98 47 05 00 07 00 00 3F | .......I....
```

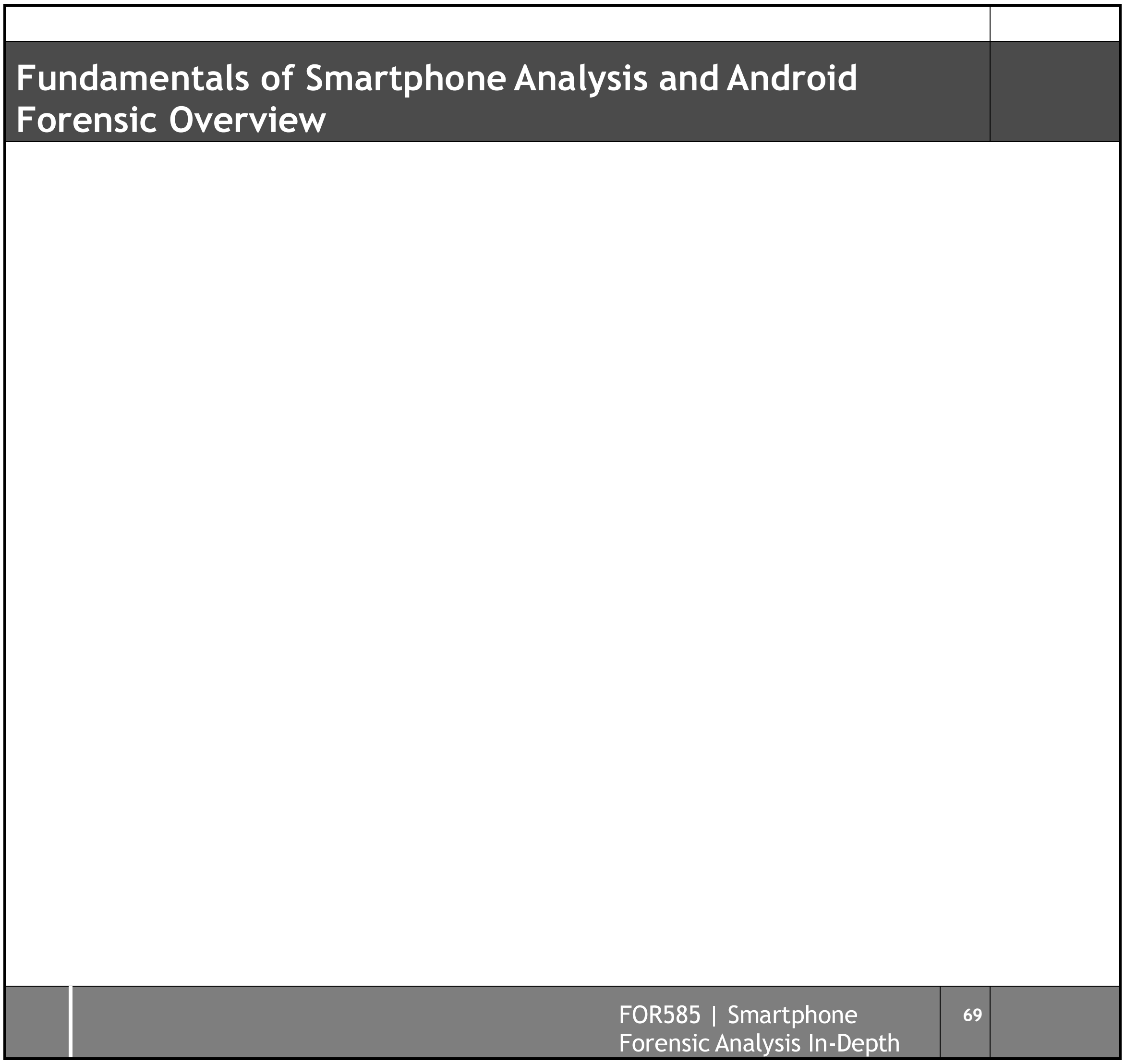
#	Offset	Length	Value	Source
1	0xE1D23817	0x8	ActivitySensorData.ActivitySensorDataMeasurement....	/root/private/var/mobile/Library/Health/healthdb_s...
2	0xE1D23827	0x1	ActivitySensorData.ActivitySensorDataMeasurement....	/root/private/var/mobile/Library/Health/healthdb_s...
3	0xE1D23830	0x8	ActivitySensorData.ActivitySensorDataMeasurement....	/root/private/var/mobile/Library/Health/healthdb_s...
4	0xE1D23840	0x1	ActivitySensorData.ActivitySensorDataMeasurement....	/root/private/var/mobile/Library/Health/healthdb_s...

Here we can see the offset in the file dump where the data exists. This offset will be different than the offset of the content in a database, plist or other file.

Lab 1.1

Familiarization with Physical Analyzer

This page intentionally left blank.



Fundamentals of Smartphone Analysis and Android Forensic Overview

This page intentionally left blank.

- AXIOM Process – Processes electronic data
  - Support for third-party images (UFED, XRY, GrayKey, iTunes backups)
  - Import options for Media cards, Drones
  - Support for Mobile, Cloud, and Computer
- Axiom Examine – The analytical platform

- Magnet Acquire – Device acquisition tool

Magnet AXIOM is a great tool for acquiring and examining mobile devices. Your instructor will provide a demo of the tool, and you will become more familiar with it through your lab work in this course. By the end of this course, you should be familiar enough with all of the tools to know the strengths and weaknesses of each. AXIOM is strong at parsing browser artifacts, carving for additional SQLite databases, timelining, filtering and categorizing data.

Magnet offers additional products:

- Official reseller of GrayKey

- AXIOM Additional Features:
  - Cloud – Token and Password Account Based Retrieval, Warrant Returns, Self-archives, AWS, O365, and OSINT
  - Magnet.AI – detection of bedrooms, buildings, child abuse, ID Documents, Paper Documents, Drones/UAVs, Drugs, Hate Symbols, Human Faces, License Plates, Militants, Money, Nudity, Screen Captures, Vehicles, and Weapons

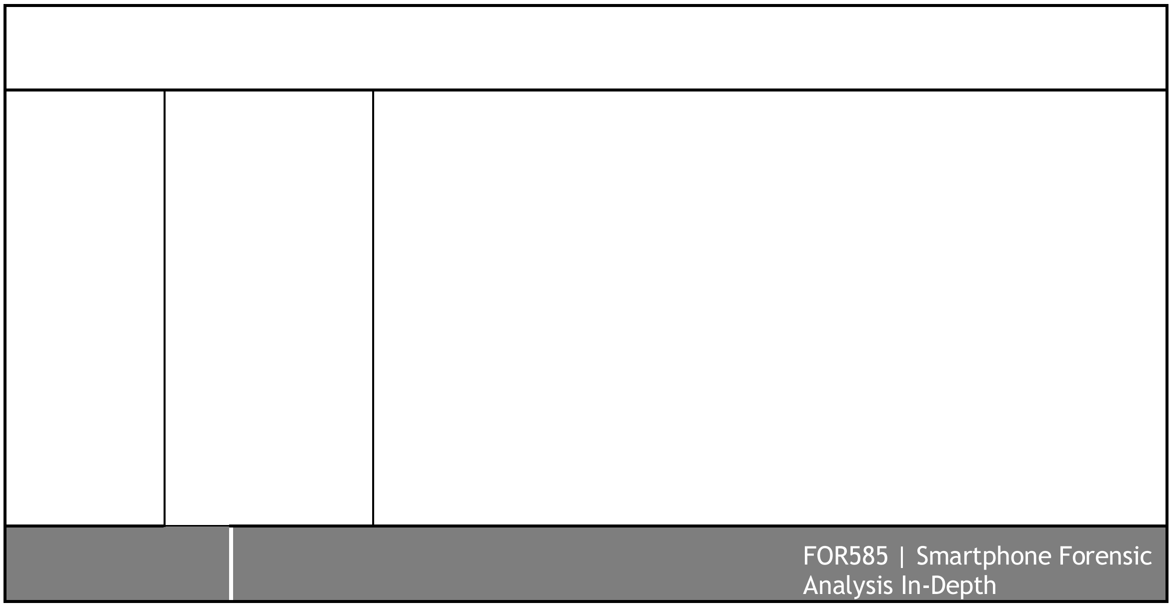
- Plist Viewer – binary and xml
  Plist decoding including
  NSKeyed Archiver format
- Custom Artifacts and Artifact Exchange
- Connections – relationships between artifact attributes

Once the image file is loaded into AXIOM, the **Artifacts** view is the most common place to start. Here, you can triage what the tool has parsed. Keyword searching, filtering, timelining, establishing connections and tagging artifacts of relevance are available in this field. To dive into the File System, one can simply double click the artifact or change views from **Artifacts** to **File system.** Should you ever want to return to the **Artifacts** view, simply press the **home** button.

Start with a solid triage in AXIOM. Web-related artifacts will highlight all parsed browser data in an organized way. The categorization takes the guesswork out of the equation, as it narrows down immediately if something is cache, synced, cookies, bookmarks, history, and more.

Chats are another popular area to triage. You can switch to a conversation view for ease of examination. The **Conversation view** button is shown above. You can click on a link in **Source** in the right pane to directly jump into the file system where the databases, plists, and other files exist.

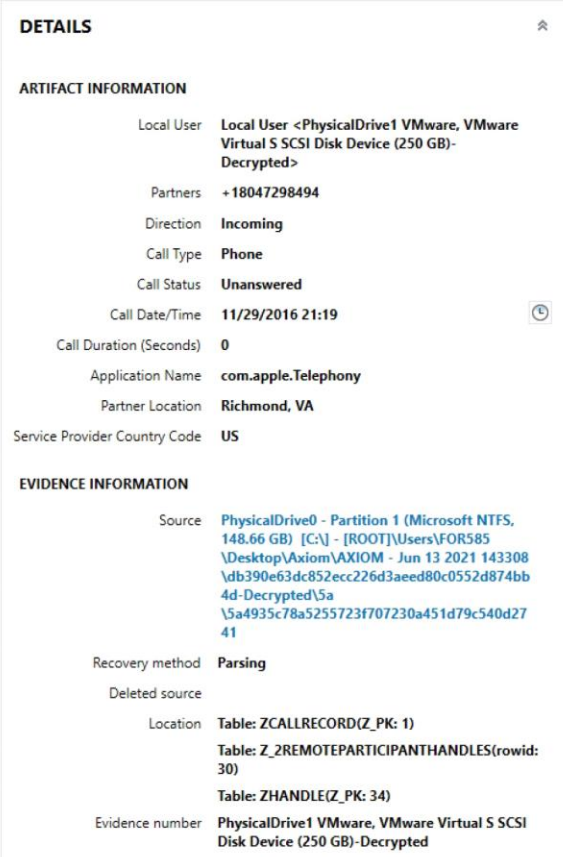
Several features are provided in AXIOM via the main toolbar. From here, you can create a timeline, view connections, examine artifacts with geolocation information and more. The Dynamic App Finder is a great feature within AXIOM that is selected during the parsing piece in AXIOM Process. Here, the examiner can elect to search for additional artifacts that are part of SQLite databases not parsed by the tool.



FOR585 | Smartphone Forensic Analysis In-Depth

There are hidden features inside of AXIOM that are extremely beneficial. One is the Relative Time, which I like to call "Timeline on the Fly." Not all artifacts show this little clock to the right of them, but when they do, it can be extremely helpful. Let's say you are working an investigation and you need to know what happened immediately before an artifact hit the device or need to see what happened after an application or piece of malware was installed – this feature is built to do just that. Timeline a specific item that is relative to that item.

AXIOM also does an “in-content” search, but it is not as thorough as Oxygen.



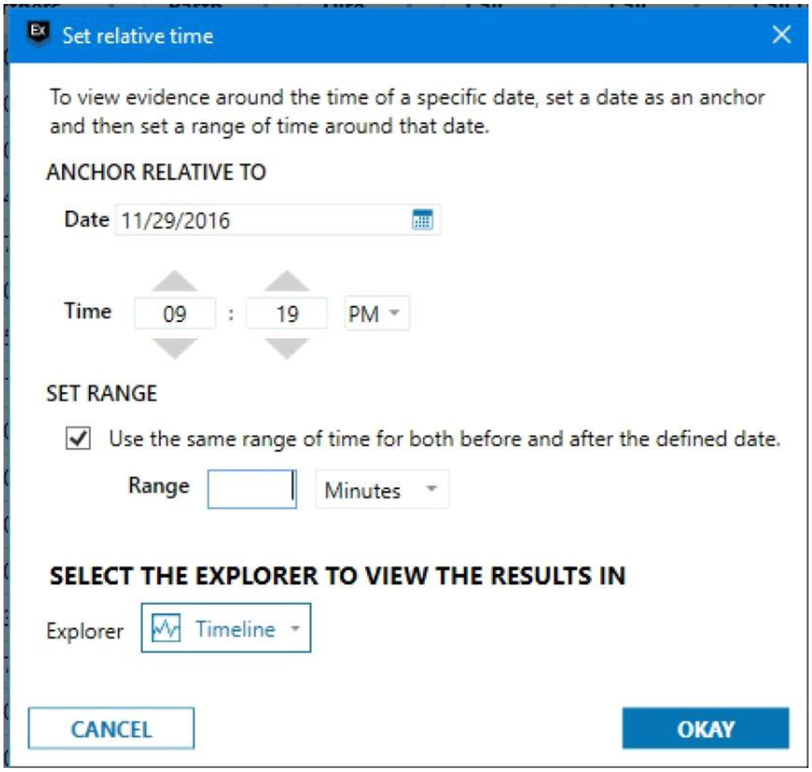
DETAILS

ARTIFACT INFORMATION

Local User	Local User <PhysicalDrive1 VMware, VMware Virtual S SCSI Disk Device (250 GB)-Decrypted>
Partners	+18047298494
Direction	Incoming
Call Type	Phone
Call Status	Unanswered
Call Date/Time	11/29/2016 21:19
Call Duration (Seconds)	0
Application Name	com.apple.Telephony
Partner Location	Richmond, VA
Service Provider Country Code	US

EVIDENCE INFORMATION

Source	PhysicalDrive0 - Partition 1 (Microsoft NTFS, 148.66 GB) [C:\] - [ROOT]\Users\FOR585\Desktop\Axiom\AXIOM - Jun 13 2021 143308\db390e63dc852ecc226d3aeed80c0552d874bb4d-Decrypted\5a\5a4935c78a5255723f707230a451d79c540d2741
Recovery method	Parsing
Deleted source	
Location	Table: ZCALLRECORD(Z_PK: 1) Table: Z_2REMOTEPARTICIPANTHANDLES(rowid: 30) Table: ZHANDLE(Z_PK: 34)
Evidence number	PhysicalDrive1 VMware, VMware Virtual S SCSI Disk Device (250 GB)-Decrypted



Set relative time

To view evidence around the time of a specific date, set a date as an anchor and then set a range of time around that date.

ANCHOR RELATIVE TO

Date 11/29/2016

Time 09 : 19 PM

SET RANGE

☑ Use the same range of time for both before and after the defined date.

Range Minutes

SELECT THE EXPLORER TO VIEW THE RESULTS IN

Explorer Timeline

This page intentionally left blank.

This page intentionally left blank.

SANS DFIR

DIGITAL FORENSICS & INCIDENT RESP

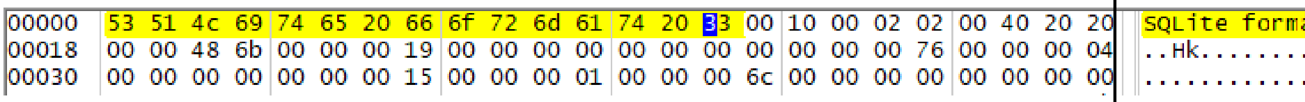
Lab 1.2

Familiarization with AXIOM

This page intentionally left blank.

Fundamentals of Smartphone Analysis and Android Forensic Overview

This page intentionally left blank.



```
00000   53 51 4c 69 74 65 20 66 6f 72 6d 61 74 20 33 00 10 00 02 02 00 40 20 20   SQLite forma
00018   00 00 48 6b 00 00 00 19 00 00 00 00 00 00 00 00 00 00 00 76 00 00 00 04   ..Hk........
00030   00 00 00 00 00 00 00 15 00 00 00 01 00 00 00 6c 00 00 00 00 00 00 00 00   ............
```

- **Relational Databases,** used to store, access, and modify data
- **OS Portable** and supported on Windows, Linux, iOS, OS X, and Android
- **Open Source** and not licensed (free = popular)
- Supported by Perl, Python, Java, PHP, Ruby scripting languages
- Optimized for small/medium applications like smartphones
- Not as fast as commercial databases for large datasets
- Size can range from 512 bytes to 281 TB
- Deleted data is retained in Free Pages

SQLite databases are widely used by smartphones, especially Android and iOS devices, for data storage and organization of structured or relational data, such as contacts, SMS, MMS, call history, application data, and so on. Each database consists of one or more pages, which are logical units that store data.[1]

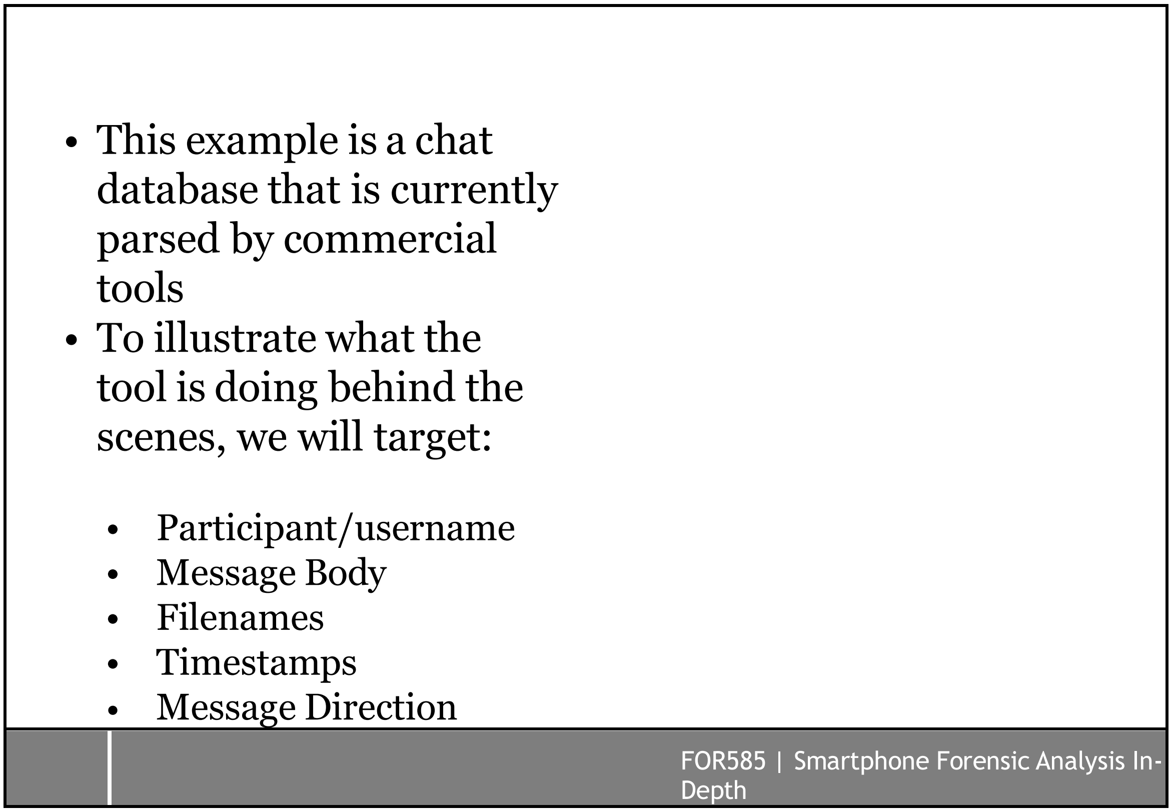
A great deal of information about SQLite database structures and uses can be found at https://www.sqlite.org/docs.html.

SQLite databases were invented in 2000 by D. Richard Hipp from General Dynamics while he was working on Guided Missile Destroyers for the U.S. Navy. SQLite databases are **relational databases** that are used to store, access, and modify data. SQLite databases are **OS Portable** and supported on Windows, Linux, iOS, OS X, and Android platforms, making them ideal for smartphone applications.

SQLite is also popular because it's open source, free, and doesn't require licensing. SQLite is supported by Perl, Python, Java, PHP, and Ruby scripting languages, making it ideal for forensics purposes. It is optimized for small/medium applications like smartphones, but it is not as fast as commercial databases for large datasets. SQLite database sizes can range from 512 bytes to 281 TB.[1]

Most every phone operating system we discuss will utilize SQLite databases, and this is great for examinations because the databases are often not encrypted and contain deleted data. You have many options to include commercial tools and free scripts that are capable of carving out the deleted items from the free pages, but it is just as easy to view deleted content by viewing the database in a hex viewer.

Reference:
[1] https://for585.com/1baqh (SQLite.org)

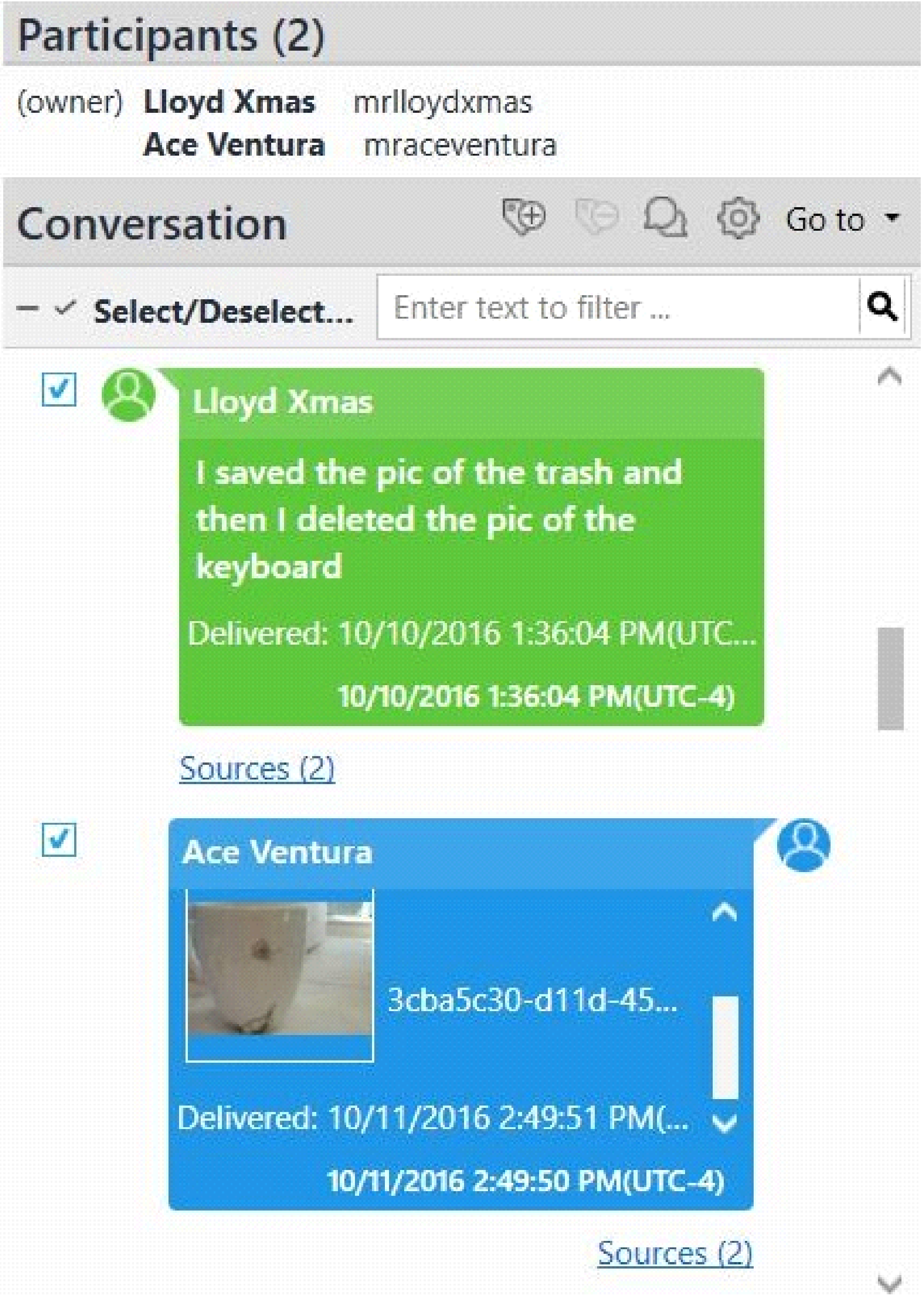


- This example is a chat database that is currently parsed by commercial tools
- To illustrate what the tool is doing behind the scenes, we will target:
  - Participant/username
  - Message Body
  - Filenames
  - Timestamps
  - Message Direction

To illustrate what the tools are doing with SQLite databases behind the scenes, take a look at this popular chat application and the types of information that was extracted by the commercial tool, Cellebrite, to aid in analysis.

The application was identified as chats/messaging and as expected, it contains data like messages, timestamps, participants, etc. In addition to these items, these databases can store coordinates, phone numbers, addresses, notes, files, file paths, and more.

While commercial tools may be able to extract and package this information for the analyst in a concise manner, if the tool does not parse the application, this same data can be presented by manually querying data from the original database file where it is all contained. This is why it is important to understand how to get just the important data from databases if they aren’t being parsed for you.



Participants (2)

(owner) **Lloyd Xmas** mrlloydxmas
Ace Ventura mraceventura

Conversation

☑ **Lloyd Xmas**

I saved the pic of the trash and then I deleted the pic of the keyboard

Delivered: 10/10/2016 1:36:04 PM(UTC...

10/10/2016 1:36:04 PM(UTC-4)

Sources (2)

☑ **Ace Ventura**

3cba5c30-d11d-45...

Delivered: 10/11/2016 2:49:51 PM(...

10/11/2016 2:49:50 PM(UTC-4)

Sources (2)

- Look for the application of interest

Application Directory

- Search **Data** and **Shared** folders

Shared

Directory

- Folder names have been modified by the tools

The first step in manually analyzing a database is to locate the file of interest.

The example above depicts locating the database file that contains all of the message data for the Kik application installed on an iOS device. Data can be contained in one or more databases and other associated files: plists (iOS), xml (Android), and others.

The database (on the sample iOS device) was located by examining the contents of the file system in two main areas where application data is expected. This may look different depending on which forensic tool you are utilizing to display the file system.

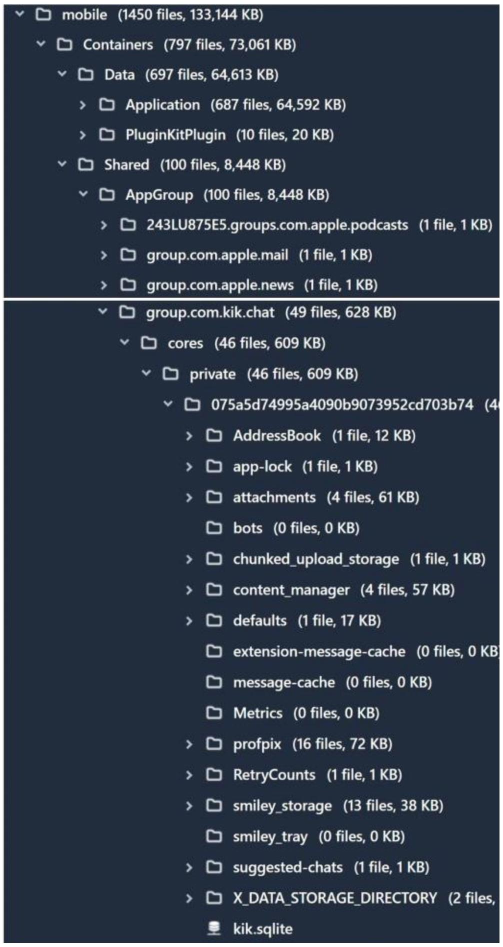
Cellebrite: This is the true path to where the data resides locally on a physical or jailbroken (FFS) iOS device:
private/var/mobile/Containers/**Data**/Application/<GUID>/ – Where GUID is represented by the application directory (Ex: com.kik.chat)
private/var/mobile/Containers/**Shared**/AppGroup/<GUID>/ – Where GUID is represented by the shared directory (Ex: group.com.kik.chat)

Axiom: Loading the same dataset into Axiom, you will notice that the paths differ slightly as each artifact is referenced by its particular "Domain"

AppDomain-com.kik.chat/
AppDomainGroup-group.com.kik.chat/

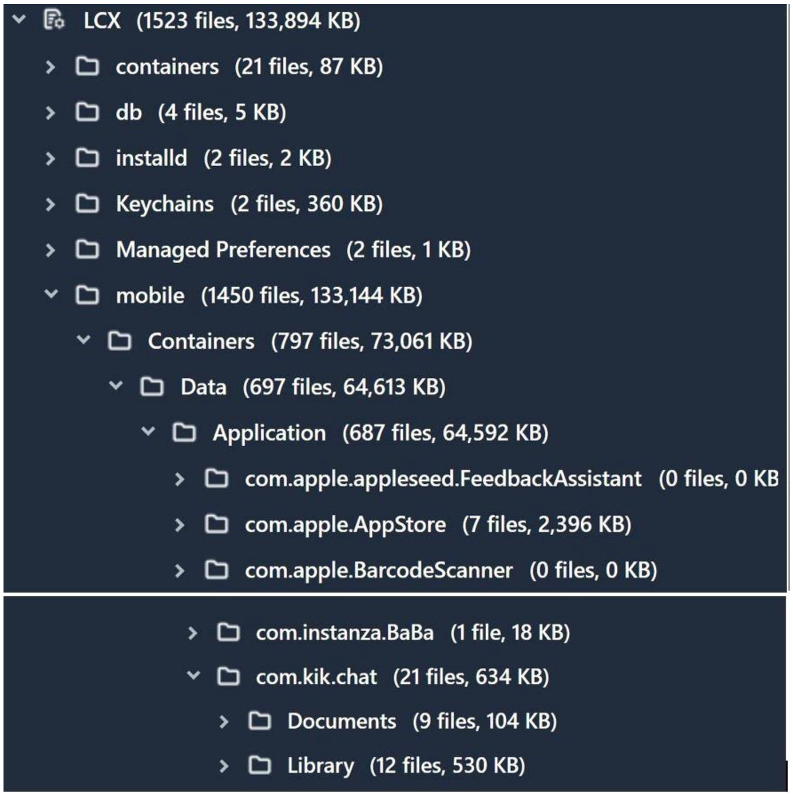
Neglecting to review the Shared (Group) directories would have meant overlooking the database that holds all of the important information.

Use this same approach to locate database files on the iPhone located in the mobile/Library, such as call history, address books, SMS, etc.



```
mobile (1450 files, 133,144 KB)
  Containers (797 files, 73,061 KB)
    Data (697 files, 64,613 KB)
      Application (687 files, 64,592 KB)
      PluginKitPlugin (10 files, 20 KB)
    Shared (100 files, 8,448 KB)
      AppGroup (100 files, 8,448 KB)
        243LU875E5.groups.com.apple.podcasts (1 file, 1 KB)
        group.com.apple.mail (1 file, 1 KB)
        group.com.apple.news (1 file, 1 KB)
        group.com.kik.chat (49 files, 628 KB)
          cores (46 files, 609 KB)
            private (46 files, 609 KB)
              075a5d74995a4090b9073952cd703b74 (4
                AddressBook (1 file, 12 KB)
                app-lock (1 file, 1 KB)
                attachments (4 files, 61 KB)
                bots (0 files, 0 KB)
                chunked_upload_storage (1 file, 1 KB)
                content_manager (4 files, 57 KB)
                defaults (1 file, 17 KB)
                extension-message-cache (0 files, 0 KB
                message-cache (0 files, 0 KB)
                Metrics (0 files, 0 KB)
                profpix (16 files, 72 KB)
                RetryCounts (1 file, 1 KB)
                smiley_storage (13 files, 38 KB)
                smiley_tray (0 files, 0 KB)
                suggested-chats (1 file, 1 KB)
                X_DATA_STORAGE_DIRECTORY (2 files,
                kik.sqlite
```

Cellebrite



```
LCX  (1523 files, 133,894 KB)
  containers  (21 files, 87 KB)
  db  (4 files, 5 KB)
  installd  (2 files, 2 KB)
  Keychains  (2 files, 360 KB)
  Managed Preferences  (2 files, 1 KB)
  mobile  (1450 files, 133,144 KB)
    Containers  (797 files, 73,061 KB)
      Data  (697 files, 64,613 KB)
        Application  (687 files, 64,592 KB)
          com.apple.appleseed.FeedbackAssistant  (0 files, 0 KB
          com.apple.AppStore  (7 files, 2,396 KB)
          com.apple.BarcodeScanner  (0 files, 0 KB)

          com.instanza.BaBa  (1 file, 18 KB)
          com.kik.chat  (21 files, 634 KB)
            Documents  (9 files, 104 KB)
            Library  (12 files, 530 KB)
```

Axiom

These three columns depict the tree pane showing the kik.android directory

FOR585 | Smartphone Forensic Analysis In-Depth

The same technique applies to Android devices. First, locate the database files by digging into the Application directory.

The path as displayed by Cellebrite:
USERDATA/Root/data/kik.android

The path to where the data is actually located on the device:

USERDATA/data/kik.android

SQL Queries can eliminate unwanted or unnecessary data

FOR585 | Smartphone Forensic Analysis In-Depth

In order to narrow down the information that an analyst needs to process, SQLite queries can be used to pull only necessary or valuable data from database tables.

The following slides will illustrate how to use a free tool like SQLite Database Browser to query a database for items of interest and return intelligible results.

Several for-purchase tools also provide the ability to query SQLite databases using point-and-click or drag- and-drop techniques instead of writing out an actual query. Cellebrite has incorporated an SQLite Wizard and Sanderson Forensics has a compilation of tools called Sanderson Forensic Toolkit for SQLite.

Tables

Columns

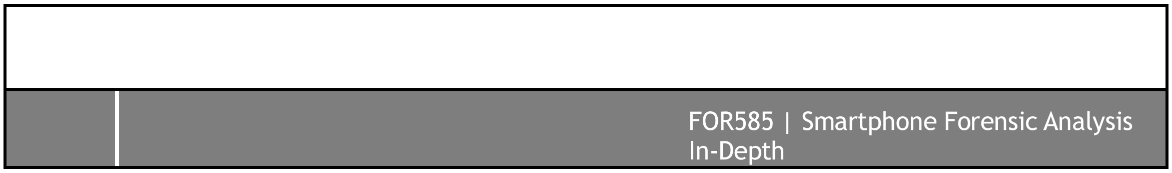
Rows

FOR585 | Smartphone Forensic Analysis In-Depth

Each SQLite file corresponds to a **database** that can contain many **tables**. Each **table** contains **rows** of data.

In order to successfully perform a query on a database, it is important to become familiar with some of the SQLite terminology.

- Identify a database of interest from the device.
- Each database is made up of a series of items:
  - **Tables** are the different categories of data contained in the database. In this example, **tables** contain categories like ZKIKCHAT, ZKIKUSER, and ZKIKMESSAGE.
  - Each **table** or **category** consists of multiple **columns**. The **columns** describe the data in each category. They are descriptions or properties. In this example, some of the **columns** that make up the ZKIKMESSAGE table are ZUSER, ZRECEIVEDTIMESTAMP, ZTIMESTAMP, and ZBODY.
  - Finally, the user data that is categorized in **tables** and defined by **column** descriptors can be found in each **row** or **record**.
- Identify items of interest by looking at the contents of each **table** and their associated **columns**.


FOR585 | Smartphone Forensic Analysis In-Depth

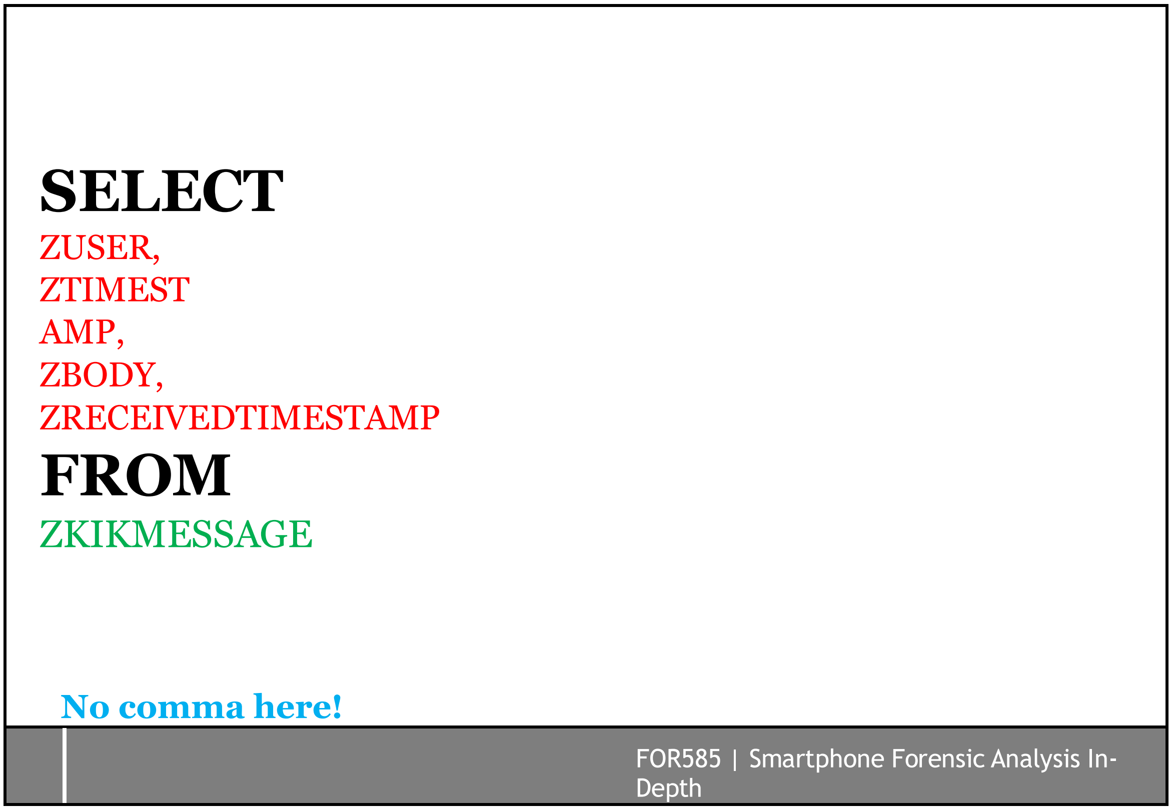
With the database located, it was opened in Physical Analyzer. A quick peek into the ZKIKMESSAGE table proved to contain lots of meaningful information. In fact, this table may contain MORE information than what is essential to analyzing communication with this application. This particular table holds more than 20 different columns.

To simplify this example, examine just the ZKIKMESSAGE table. ZKIKMESSAGE holds a significant amount of data, but we are interested in just pulling the following information:

- The User
- Message Timestamp
- Content of the Message
- The Timestamp that the Message was Received

(These are pictured in the screenshot above by rearranging the columns.)

To do this, we can query the particular table within this database to return just the isolated results that we are seeking. In addition, we can display the columns in terms that make sense to us if they are obscure or not altogether self-explanatory in their native form.



SELECT

ZUSER,
ZTIMEST
AMP,
ZBODY,
ZRECEIVEDTIMESTAMP

FROM

ZKIKMESSAGE

No comma here!

To accomplish this task manually, the free tool, SQLite Database Browser, is being utilized to open and query the database for those items of interest.

The SELECT statement is what is used to retrieve specific data from these databases. When writing a SELECT statement, you must identify:

- The information you wish to retrieve (**columns**)
- The **table** that contains the above information using the previous database as an example, to pull:
  - The User
  - Message Timestamp
  - Content of the Message
  - The Timestamp that the Message was Received

A simple SQLite query would be:

S
E
L
E
C
T

ZUSER, ZTIMESTAMP, ZBODY, ZRECEIVEDTIMESTAMP FROM ZKIKMESSAGE

where ZUSER, ZTIMESTAMP, ZBODY, and ZRECEIVEDTIMESTAMP are the **columns** and ZKIKMESSAGE is the **table**.

Pay special attention to the commas that separate each column. More detailed instructions on generating SELECT statements can be found in the link in the reference below.[1]
This information was entered into SQLite Database Browser by selecting the tab “Execute SQL”.

Reference:
[1] https://for585.com/sqlitestatements

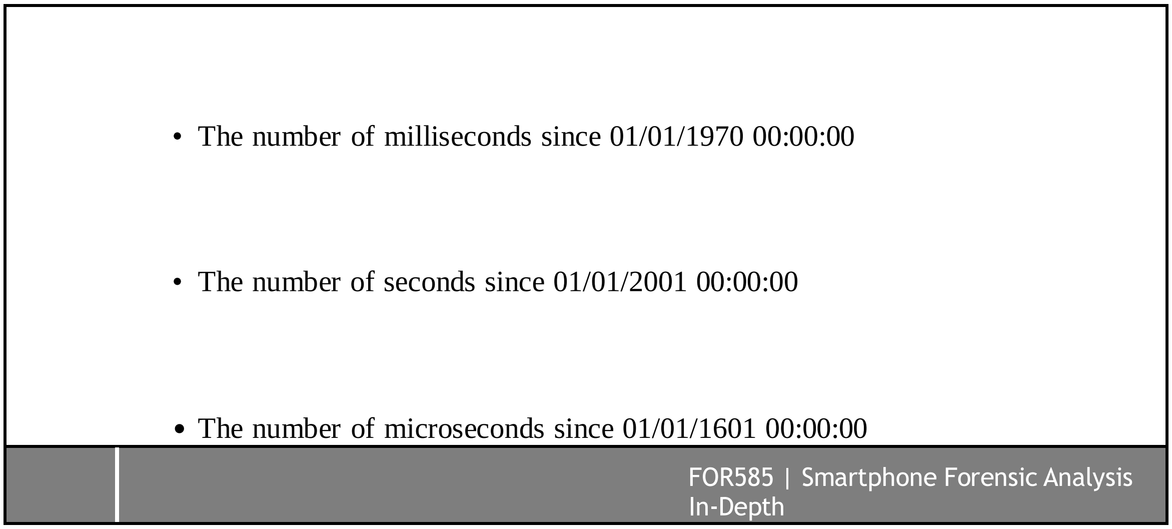
Use **AS** along with quotation marks to change the name of a resulting column to make the output of the query make more sense.

The example below shows the original query language that has been replaced by the new strings to account for the renaming of the columns:

SELECT ~~ZUSER~~ ZUSER **AS** “User”, ~~ZTIMESTAMP,~~ ZTIMESTAMP **AS** “Timestamp”, ~~ZBODY,~~ ZBODY **AS** “Message”, ~~ZRECEIVEDTIMESTAMP,~~ ZRECEIVEDTIMESTAMP **AS** “Message Received” FROM ZKIKMESSAGE

*Strikethroughs represent the original query. The new query language with applied column names appears directly below.

- The number of seconds since 01/01/1970 00:00:00



- The number of milliseconds since 01/01/1970 00:00:00
- The number of seconds since 01/01/2001 00:00:00
- The number of microseconds since 01/01/1601 00:00:00

FOR585 | Smartphone Forensic Analysis In-Depth

Timestamps are stored in the databases as one of several numerical representations. Most applications will use
UNIXEPOCH time, where the time is representative of the number of seconds since 01/01/1970 00:00:00. This is usually easy to spot, since it is a 10-digit number. This ten-digit number is converted to supply the resulting time in UTC:
SELECT datetime(time COLUMN, 'unixepoch');

or in local time as suggested by the device settings: **NOTE: This will be reflective of your forensic workstation time zone settings!**
SELECT datetime(time COLUMN, 'unixepoch', 'localtime');

UNIXEPOCH is also calculated to the millisecond in some applications for an even more accurate representation of the time. Also, easy to spot, this is a 13-digit number that must first be divided by 1,000 before making the conversion.[1]

MILLISECONDS
(13-digit number)
Convert
milliseconds in
your query, like
this
SELECT datetime(time COLUMN/1000,'unixepoch','localtime');

Another commonly used time format is **Mac Absolute** time, which is defined as the number of seconds since 01/01/2001 00:00:00. Like the name suggests, this timestamp is utilized by most iOS applications. In order to correctly convert this timestamp, first add the number of seconds since UNIXEPOCH time to Mac Absolute time, 978307200, then convert to local time, as in the queries above.[2]
SELECT datetime(time COLUMN + 978307200, 'unixepoch', 'localtime');

These are not the only timestamps that you may come across when analyzing application data, but they are the timestamps that appear most frequently. Some timestamps, like **Chrome** as an example, account for time accurate to the MICROSECOND, which requires dividing the number by 1,000,000:[3]
SELECT datetime(time COLUMN/1000000 + (strftime('%s','1601-01-01')),'unixepoch','localtime');

References:

- https://for585.com/unixepochtimestamps
- https://for585.com/macabsolutetimestamps
- https://for585.com/chrometimestamps

FOR585 | Smartphone Forensic Analysis In-Depth

It may not always be apparent by simply looking at the timestamps, but by using the process of elimination, it is usually easy to apply the correct timestamp conversion to the database.

Below is a list of possible query strings to convert the timestamp:

- datetime(ZTIMESTAMP, ‘unixepoch’,'localtime') AS "Timestamp"
- datetime(ZTIMESTAMP/1000,’unixepoch','localtime') AS "Timestamp"
- datetime(ZTIMESTAMP+ 978307200,'unixepoch','localtime') AS "Timestamp“
- datetime(ZTIMESTAMP+ (strftime('%s','1601-01-01')),’unixepoch’,'localtime') AS “Timestamp”

By process of elimination, we can likely eliminate option 1 because the field contains more than 10 digits and option 2 because the field also doesn’t contain 13 digits. This leaves option 3, the conversion for Mac Absolute time. If you remembered that we have been utilizing the kik.sqlite database from an iOS device, this would also be another way to narrow down the correct timestamp query.

Now, using the original query that we have been building upon, add in the correct conversions for both columns that contain timestamps.

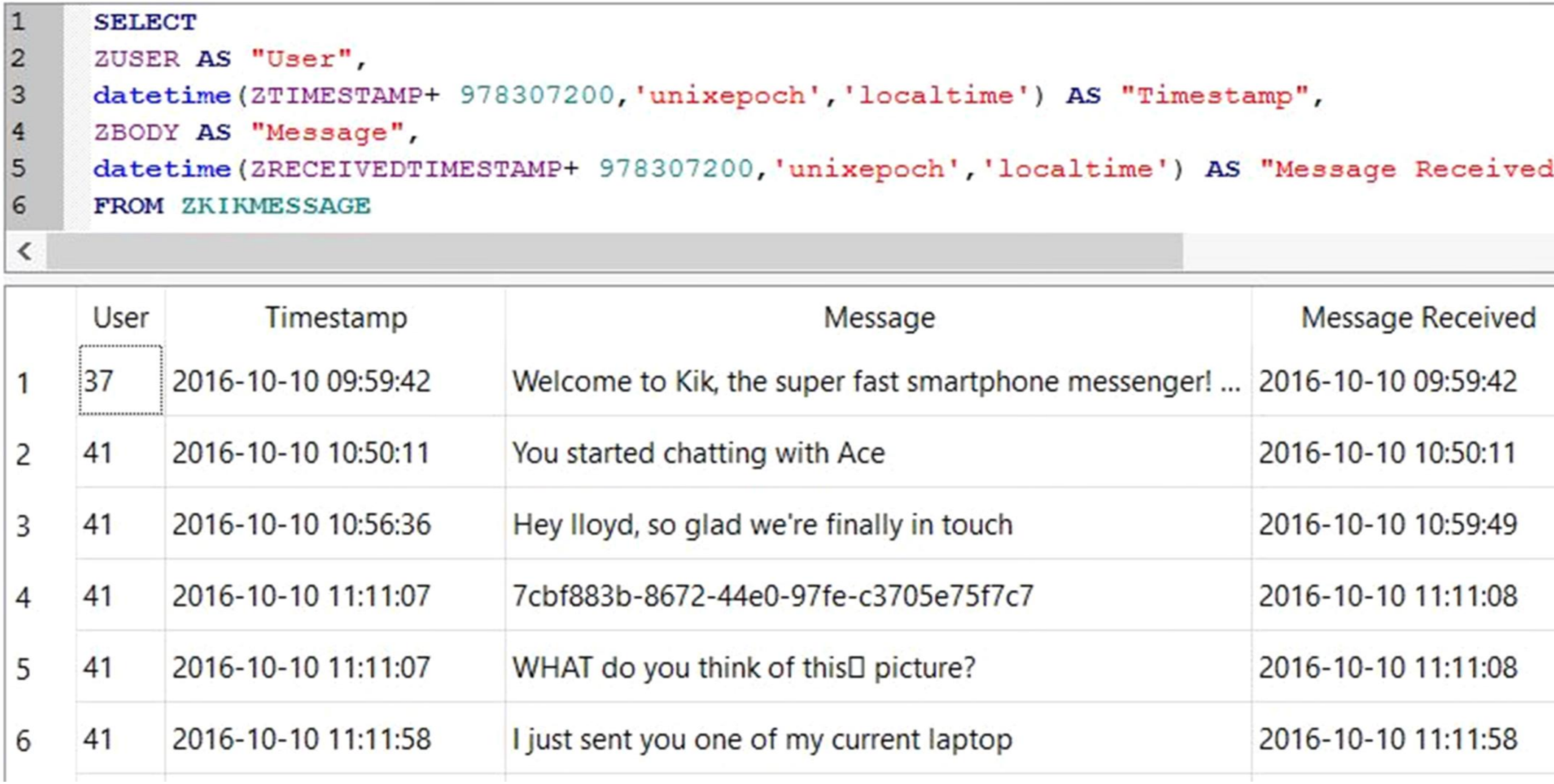
```
SELECT
ZUSER AS "User",
~~ZTIMESTAMP AS “Timestamp”,~~
datetime(ZTIMESTAMP+
978307200,'unixepoch','localtime') AS "Timestamp",
ZBODY AS "Message",
~~ZRECEIVEDTIMESTAMP AS “Message Received”~~
datetime(ZRECEIVEDTIMESTAMP+ 978307200,'unixepoch','localtime')
AS "Message Received" FROM ZKIKMESSAGE
```

The result of applying the Mac Absolute time conversion

should be legible timestamps. Each timestamp conversion

will ALWAYS begin with **datetime**.

Immediately following this syntax, you will reference the ORIGINAL column name that contains the
timestamp (e.g., **ZTIMESTAMP** and **ZRECEIVEDTIMESTAMP** in the above example).



```sql
SELECT
ZUSER AS "User",
datetime(ZTIMESTAMP+ 978307200,'unixepoch','localtime') AS "Timestamp",
ZBODY AS "Message",
datetime(ZRECEIVEDTIMESTAMP+ 978307200,'unixepoch','localtime') AS "Message Received
FROM ZKIKMESSAGE
```

	User	Timestamp	Message	Message Received
1	37	2016-10-10 09:59:42	Welcome to Kik, the super fast smartphone messenger! ...	2016-10-10 09:59:42
2	41	2016-10-10 10:50:11	You started chatting with Ace	2016-10-10 10:50:11
3	41	2016-10-10 10:56:36	Hey lloyd, so glad we're finally in touch	2016-10-10 10:59:49
4	41	2016-10-10 11:11:07	7cbf883b-8672-44e0-97fe-c3705e75f7c7	2016-10-10 11:11:08
5	41	2016-10-10 11:11:07	WHAT do you think of this□ picture?	2016-10-10 11:11:08
6	41	2016-10-10 11:11:58	I just sent you one of my current laptop	2016-10-10 11:11:58

Depending on the information you are analyzing, you may have hundreds or thousands of results returned as a result of your query, and it can be very beneficial to sort and/or filter results for brevity or ease of viewing.

Sorting is a good way to make sure you still have the same number of records and that you don't accidentally redact important information. This can be accomplished using the **ORDER BY** syntax.

Filtering will limit your records to just those that satisfy a certain criteria. For example, the **GROUP BY** syntax will limit results to just one record for any particular specified column.

The **WHERE** syntax is easily used to only return results that match a specified value. If you want it to match on the entire value, you can utilize an equal (=) sign, for example:
WHERE column = 0

If you are looking for a particular value within a result, like for example a word within the message body, use **LIKE** along wit

h a wildcard (%). For example:
WHERE column LIKE ‘%yourvalue%’ where the exact phrase is contained between two %s.

Should you want to filter on just the beginning or end of a particular phrase or value, you can use **LIKE ‘A%’** for anything that begins with the letter A for example, or **LIKE ‘%Z’** for anything that ends in Z. The % in these examples acts as a wildcard and says any value can come after the A or before the Z respectively.

Several examples have been provided on the following pages.

ORDER BY will return the same number of records but sort them in ascending or descending order. This is a good choice if you do not want to accidentally eliminate any records.

GROUP BY will return just one unique record per column requested. Notice that only one record per user is returned in the result below.

WHERE is selective in that it only returns results that match your particular request. In the statement below, only records where the user is equal to 41 are returned.

WHERE can also be used along with LIKE to match on an entire element or to matching on records that begin or end with a specified value or values. Use % as a wildcard. Notice below we want only those message that contain the word “Lloyd”.

We could use wildcards to return records with a message body that begin with the letter “C”.

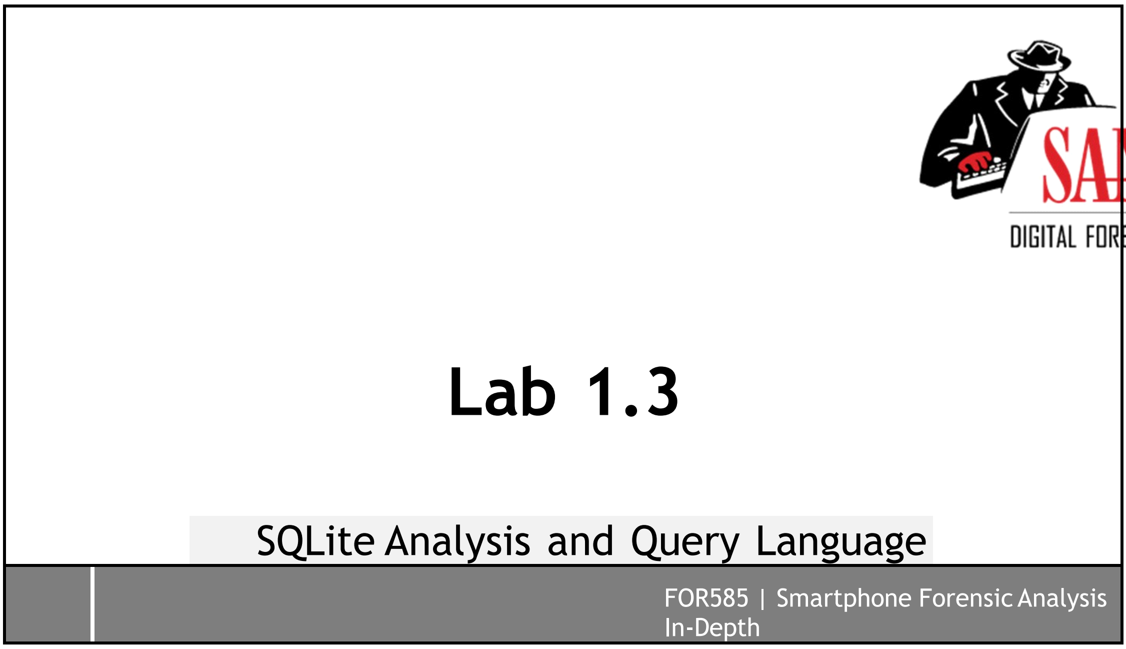
Or we could use wildcards to return records sent by a user number which ends in the number “2”.

FOR585 | Smartphone Forensic Analysis In-Depth

Now that we’ve shown you how to complete a simple SQLite query on a database, why is this so important to examiners?

- Smartphones are almost exclusively using SQLite databases to store data.
- New features are added to applications constantly: Applications are updated constantly. Sometimes the update may fix an existing bug or security flaw or sometimes the update may change or add to the functionality of an application.
- Database schemas can change: If even a small change is made to the database schema, like a change in a COLUMN or TABLE name, this will break a query. An overall change to the file path will also break the query if the tool no longer knows where to look to find the data.
- All data is not always parsed: As noted earlier, the applications that have the most users or may garner the biggest payloads are often the apps that are built into commercial tools. If only a handful of the population is using an application, it may not make sense to invest engineering dollars on parsing out the contents.
- Not every bit of information is always extracted: You may notice that there is often only a subset of the data presented to you when a commercial tool parses an application. This may help prevent information overload for the examiner, but what other data is stored in those applications that may be of interest to your analysis?
- These methods are the same way that the commercial tools are dissecting and parsing these databases: Commercial vendors realize that they don't have time to parse out every single application available in the app stores these days, so some have even offered easy point-and-click alternatives for creating SQLite queries that are easier than the method you just learned.

These tips will make it possible to parse even those applications that your tool cannot handle.

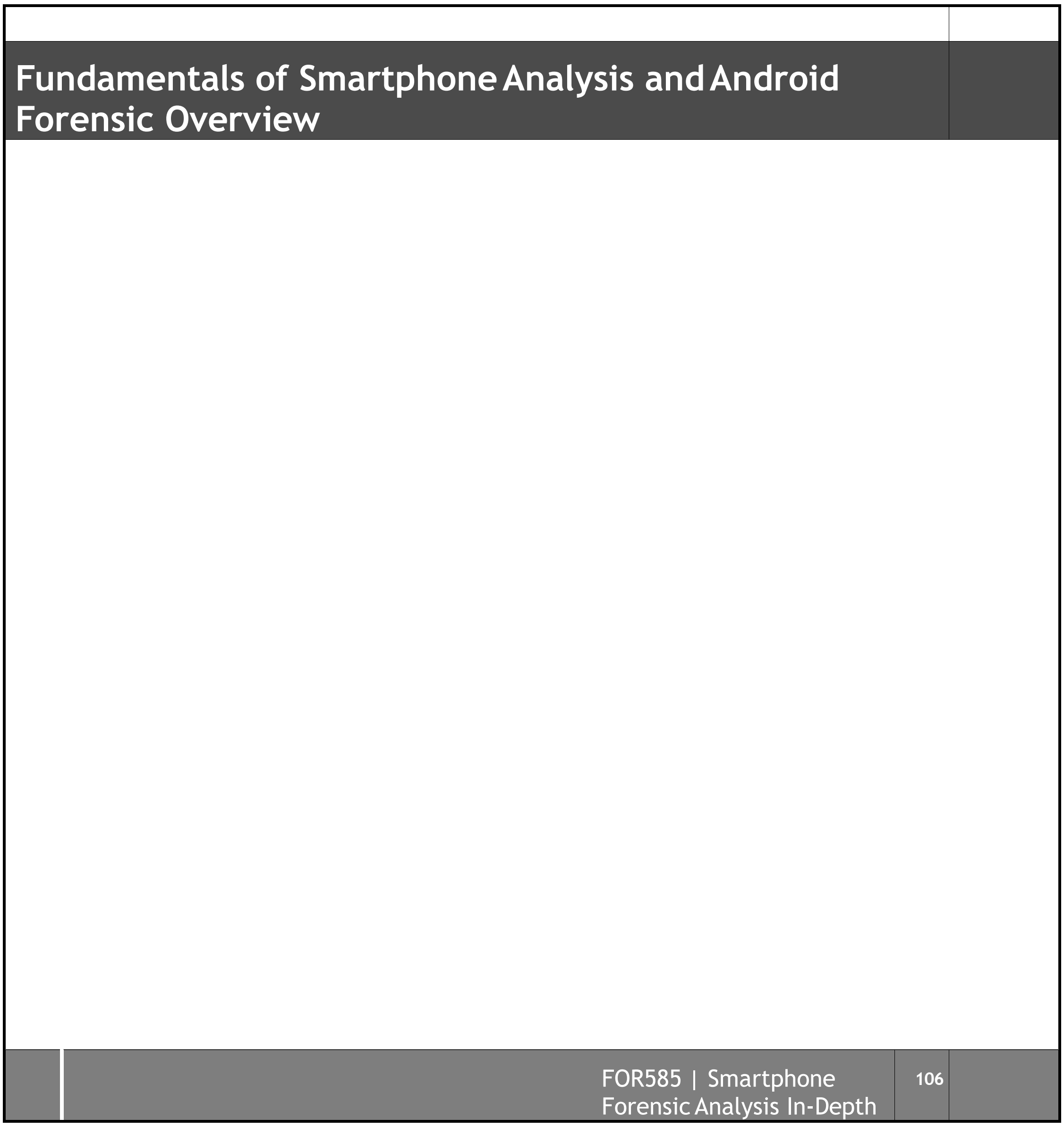


Lab 1.3

SQLite Analysis and Query Language

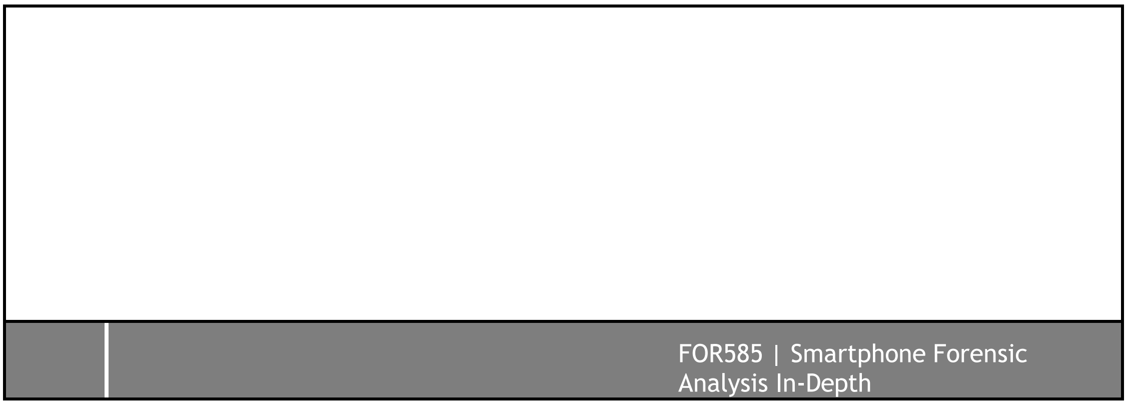
FOR585 | Smartphone Forensic Analysis In-Depth

This page intentionally left blank.



Fundamentals of Smartphone Analysis and Android Forensic Overview

This page intentionally left blank.



FOR585 | Smartphone Forensic Analysis In-Depth

Android devices vary based on the device and manufacturers, making forensic acquisition and analysis challenging if you're relying on tools that simply provide support for one file system, manufacturer, or version of Android. The commercial vendors have made drastic headway on forensic acquisition of these devices, but examiners need to be prepared to manually dig for artifacts once the file system is extracted for analysis.
Examiners must also be prepared for limited support on Android Pie v9+ devices. Don't assume you will only be dealing with EXT. Don't assume your version of Android will be supported. Don't expect to extract a ton of data unless you get at least a full file system extraction. Expect the unexpected and be ready to interact with the device in order to extract the most data. We hope this course prepares you with an arsenal of methods to tackle the Android devices that land on your desk.

User data may be stored internally and externally on Android devices. The internally stored data is saved to the NAND flash memory. The NAND is non-volatile memory, which means the data is not lost when the smartphone loses power. The NAND stores the bootloader, operating system, and user data. Although application data may be open and present in RAM, it is only temporarily stored in RAM while in use. The actual application data is stored on the NAND flash memory, or the SD card inserted in the device. A RAM capture was possible on older Android devices that had root access. We have hope that tools like Cellebrite Premium and GrayKey will eventually support this type of extraction.

Android devices use NAND flash memory for internal storage. The operating system is based on variations of the Linux kernel's long-term support (LTS) branches (depending on the device) and the software stack has evolved with versions. Further details on the operating system version and software stack can be found on https://en.wikipedia.org/wiki/Android_(operating_system)#Linux_kernel.

Historically, Android Runtime (ART) used ahead-of-time (AOT) technology, which boosted performance. Marshmallow (v6) uses the Linux kernel 3.18.10 and ART, while Nougat (v7) leverages Linux kernel 4.4.1 and ART.[1] With the introduction of Oreo (v8), Google for the first time specified a requirement for the Linux kernel being used, requiring that at least Linux kernel 4.4 be used. Android Pie (v9) runs on Linux versions 4.4, 4.9, or 4.14, where the version is dependent upon the device. Android 10, 11, and 12 have continually progressed and the Android OS Wiki mentioned in the previous slides has detailed descriptions for each. As new versions of Android are released, it's important to understand the underlying components.

Some devices may use SELinux (Security-Enhanced Linux), a version of Linux developed by the National Security Agency that was designed to push the security control to the kernel level.[2] SELinux protects the device from breaches or unauthorized access via applications on the device. The PrivatOS, which runs on the Blackphone and Blackphone II, offers additional restrictions to protect the user's data and offer full disk encryption.[3]

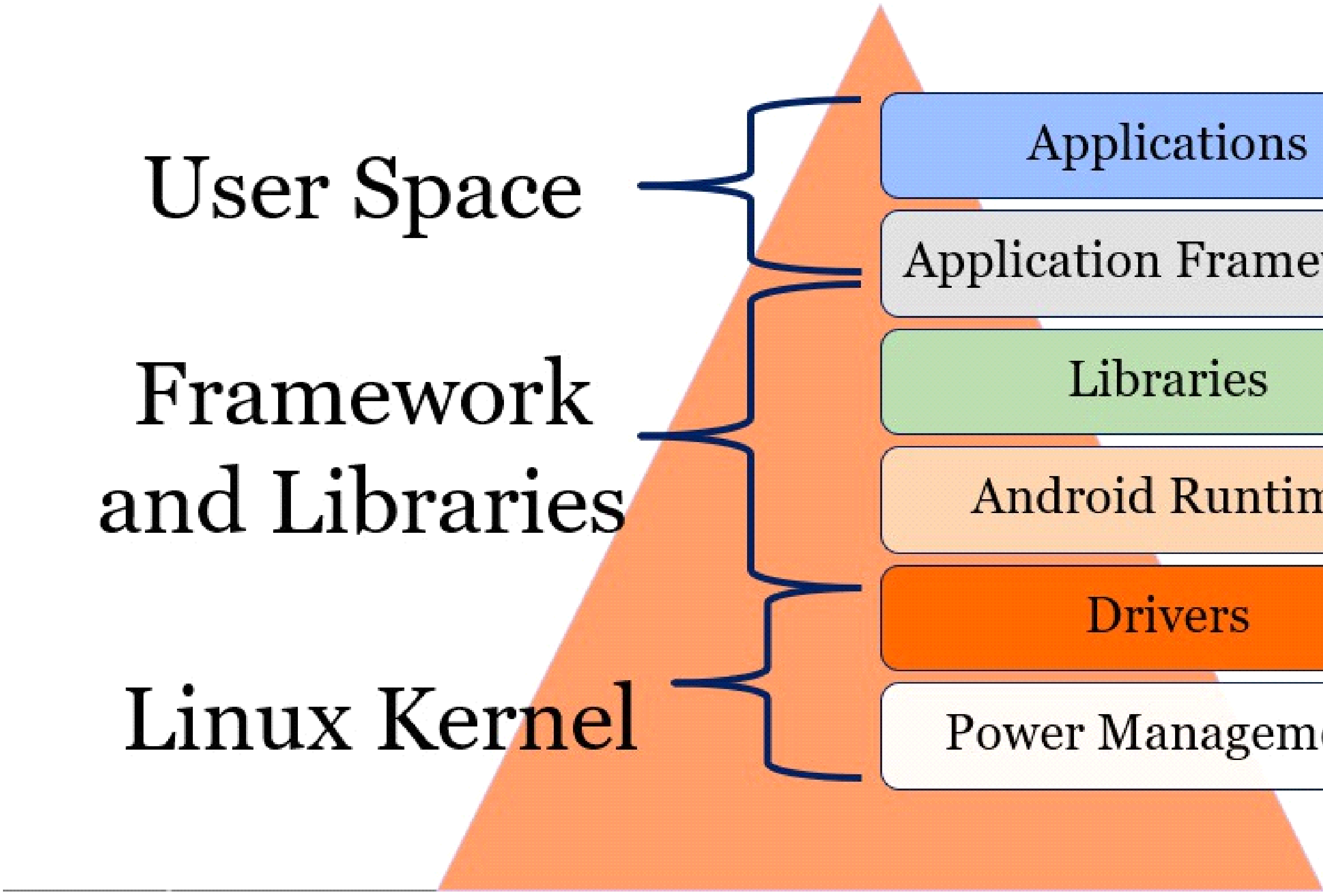
The four major file systems are discussed in the upcoming slides. Keep in mind that, although only four major file systems exist, multiple file systems function on Android devices. Essentially, if it can be mounted in Linux and run on an Android, you have yourself another file system. Android has always been known for being open source and making the code available for anyone to modify and use via the Apache license. You will find that newer devices are becoming more restrictive but remain open source. Most of the artifacts we discuss in this section will be SQLite databases and XML files. Just like Linux systems, Android sandboxes the applications to protect the user. The applications are sandboxed at the kernel level and are assigned a unique user ID (UID) to keep track of the applications being run. These files are some of the easiest to examine, as the formats are consistent and multiple tools will provide access to the data contained within the files. The skills you learned earlier in Section 1 will assist you with your Android examinations. The hard part is obtaining an acquisition method that is capable of extracting these mentioned files.

The introduction of Lollipop (v5) was an eye opener for Android. New levels of security and encryption were introduced. The changes in Lollipop and current versions introduced are discussed throughout this section.
One major change that was first introduced with new devices (with Lollipop installed with shipment) was that the user does not have to dive into the system settings on the device to enable encryption, as it is often offered the minute the user first powers on a device during initial setup. For devices to be fully encrypted, the device must have enough memory and AES processing power for the encryption to stick. Currently, we haven't seen full disk encryption causing us issues on too many devices, even on the devices running Android 7, due to the decryption support available by tools like Cellebrite and services like CAS (Cellebrite Advanced Services).
While Android versions 9–12 prove to be more difficult to acquire due to security patches, we are chugging along and trying to get crafty with our techniques to access user data.

For devices with file-based encryption, we can only expect it to cause further issues during acquisition and analysis. The various encryption options on smartphones are covered in this section and throughout the rest of this course. Our goal is to prepare you, so you are ready to handle the tough issues that end up on your desk!

References:

- Tamma, Skulkin, Mahalik, and Bommisetty, *Practical Mobile Forensics*, Third Edition (Birmingham, UK: Packt, 2018).
- https://for585.com/linux (SELinux)
- https://for585.com/privat (Silent Circle)



From a user perspective, we function from the tip of the pyramid and dive down. The device functions from the base up. Overall, all the architectural components are important, whether you realize it or not. This course teaches you that the outliers of data will sometimes be the missing link for which you were looking.

The user space is comprised of the Applications layer. This includes the primary phone applications like Contacts, Home, Phone, Browser, and all user functionality features.

The system is comprised of the framework, libraries, and the Linux kernel. A comprehensive breakdown of these components is defined in detail in *Practical Mobile Forensics*, Second Edition. The framework and libraries are essentially the glue that ties everything together and can be explained as follows:

Application Framework: Activity Manager Window Manager Content Providers
View System Package Manager Telephony Manager Resource Manager Location Manager
Notification Manager

Libraries: Surface Manager OpenGL
SGL
Media Framework FreeType
SSL

SQLite WebKit Libc

Android Runtime:
Core Libraries
Dalvik Virtual Machine (DVM)

Linux Kernel controls the smartphone drivers and power. This includes:[1]
Display Driver Camera Driver
Flash Memory Driver Binder Driver
Keypad Driver
Wi-Fi Driver Audio Drivers
Power Management

- Andrew Hoog, *Android Forensics: Investigation, Analysis and Mobile Security for Google Android* (Waltham, MA: Elsevier, 2011).

The Linux kernel area is often overlooked by examiners, but it may contain pertinent information. It is recommended to acquire all “spaces” of the device and narrow your analytical search from there. It’s best to start with everything and limit the scope instead of starting with just the user space and realizing you need the kernel and framework data.

- Can be ignored

- Exceptions: Samsung Kies/Smart Switch, Huawei, and LG specific software

- Requires username and password to access
- Ping/Ring, lock, and wipe ®

All Android devices request a Google account to set up the device, but the user can ignore and use the device without the use of Google. Should the user opt to use the device without a Google account, their data will not be backed up. Should you have a device lacking the use of a Google account, search for backup programs that the user may be relying on to protect his data should his Android fail or the device is lost. Most devices are not backed up by the user even though backup programs are available. An example of backup software is Samsung Kies, which has an .sbu file extension and can be parsed with a tool like MSAB XRY. Other backup software that you may come across includes Samsung Smart Switch, which is popular when acquisition for the devices isn't working. For Huawei, the HiSuite (PCTool) or Kobackup (APK) are options. The advantage of using HiSuite and Kobackup is that they both contain backup of App data, no matter if the app is preventing an ADB Backup. There is a tool to decode these backups created by RealityNet. A blog on the methods can be found on https://blog.digital-forensics.it/2019/07/huawei-backup-decryptor.html. You may also find the use of LG PC Suite to be helpful.

Regardless of solutions to backup, most Android devices simply rely on the use of Google. This cloud-based syncing service provides the user the ability to store and sync data for free (up to 15GB).

Gaining access to Android backup files and creating an Android backup for a forensic examination will be discussed shortly. Legal authority or consent is required to access and analyze user backup files, even if you stumble upon them. Always make sure you have permission to examine the media and backup files associated with the Android device.

Android Device Manager was introduced as the answer to those seeking a similar solution to the iOS Find My iPhone application. Android Device Manager was renamed to Find My Device and is accessed by logging in to the user's Google account. This account must be the one associated to the Android device you want to access. Just like Find My iPhone, a list of Android devices associated with that account are displayed. From here, the user can locate a lost Android, ring the device, and perform a remote wipe. At the time of writing this, Find My Device does not automatically enable remote wipe access unless the feature is enabled by the user. A remote wipe renders the Android untraceable, as the device must remain active and online for Find My Device to track and access it.[1] For Samsung devices, Find My Mobile can provide the same features.[2]

Reference:
https://for585.com/remote (Samsung Find My Mobile)

- Not all devices are "upgradable"
- Some devices may be "stuck" in their current version
- Network Service Providers and Manufacturers control if the device is eligible for an OS upgrade

Android updates are much different from those of iOS. While Apple is persistent and attempts to push updates to all iOS users as quickly as possible, Android users do not get this same experience. For Android, several things may control if the user is offered an update. The Network Service Provider may not push updates to customers for months after a new version is released. Another factor is the device manufacturer. Some manufacturers support updates for a set period and then no longer push updated versions to the device. Finally, not all devices are eligible to update regardless of NSP. Some of the models don't meet the requirements to host the update, and they will be stuck at the latest version for that device.

•FDE
Introduced,

datory*

oduced but

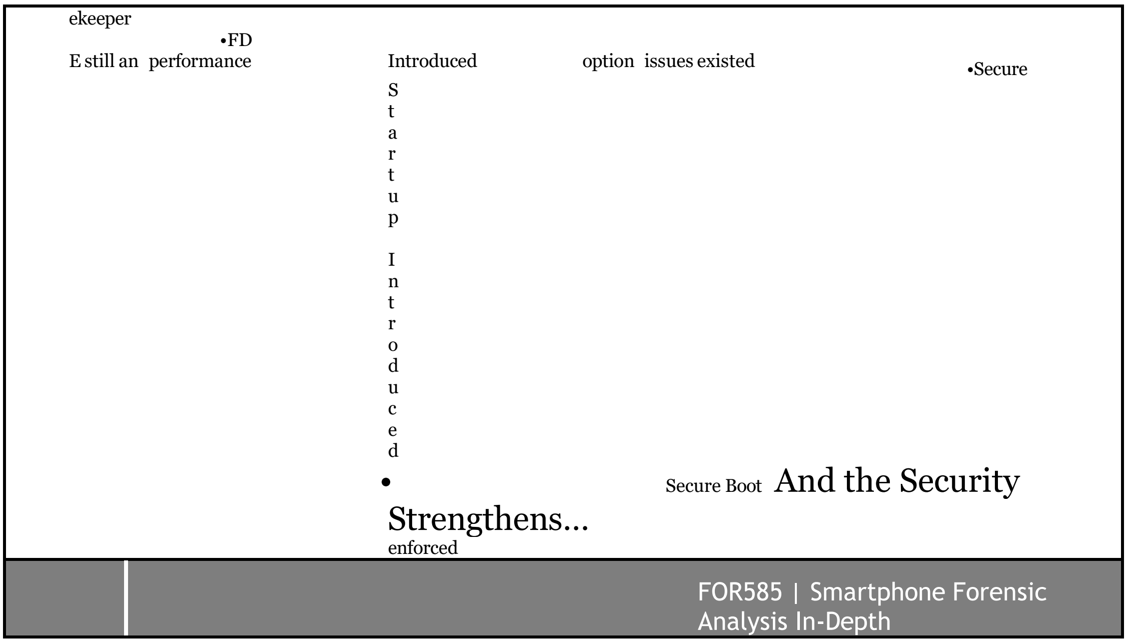
•FDE

Man

Intr

•Gat

•FBE



ekeeper

•FD

E still an performance

Introduced

option issues existed

•Secure

Startup Introduced

•

Secure Boot And the Security

Strengthens...

enforced

FOR585 | Smartphone Forensic Analysis In-Depth

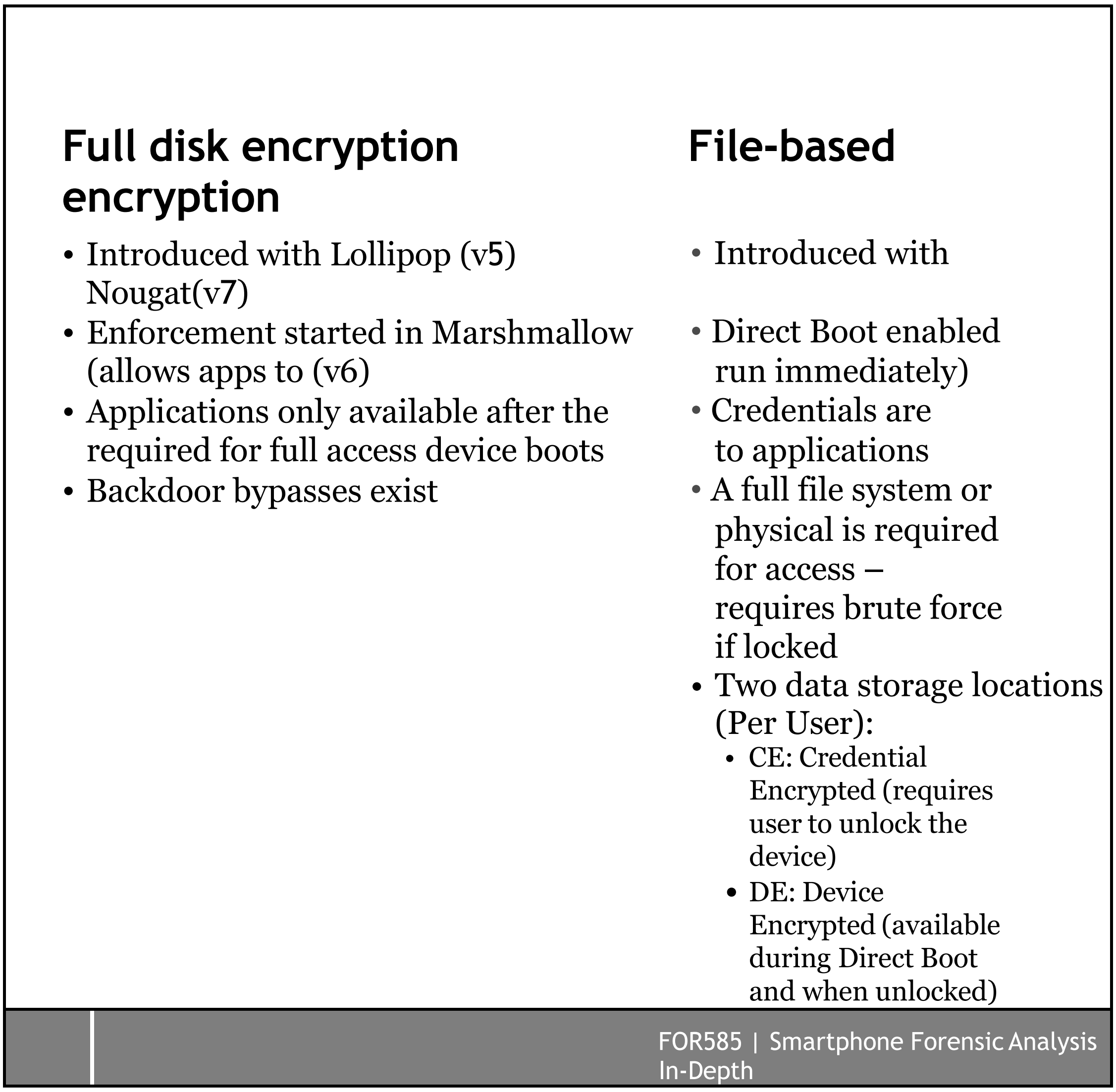
Remember the days of just dealing with lock screens on Android? Those days are gone. Let's take a look back. Google announced the release of version 5, Lollipop, in fall 2014.[1] Full disk encryption was introduced, but performance issues existed, so Google backed down on enforcing it. At this point, brute force was an easy option for us to gain access. Android 6, Marshmallow, was introduced in the fall of 2015. Marshmallow made full disk encryption mandatory *if* the device hardware met requirements set by Google. In addition, Gatekeeper password storage was introduced and prevents the password from being salted and stored in the file (gatekeeper.password.key). This makes cracking a Gatekeeper password almost impossible if a backup password is not in use. More will be covered on this in the locked Android portion of Section 2.

Nougat was the gamechanger. Like iOS, Nougat (Android v7) introduced file-based encryption as an option to the OEM. Each file is locked with a different key, making the data more secure. This also enhances the speed of the device because a file being requested can be decrypted rather than dealing with decrypting the entire file system. This also enables the device to unlock files needed for booting the system and providing notifications to the user while the device is locked.[1] The OEM has the choice between FBE or FDE on each device. Also, Secure Startup and Secure Boot were introduced. We'll look at each of these and their differences in a few slides.

Android Oreo (v8) was publicly released in August 2017, followed by Android Pie (v 9) in late summer 2018. While these new versions are sought after by users, most devices are not capable of upgrading to this version. Android 10 , Android 11, and Android 12 are more difficult to acquire, which makes analysis and research harder than before. We often rely on root access to fully examine the data in these new releases, and the lack of a functioning root for multiple devices makes this difficult. In this section and Section 2, we will look at changes that Nougat initially introduced and how it changes the playing field in our examinations. Look out for blog posts on www.smarterforensics.com and https://thebinaryhick.blog as new artifacts are uncovered, as this research is ongoing. We have tried to go file by file on as many devices and operating systems as possible. Why? So you won't have to!

Reference:

- Tamma, Skulkin, Mahalik, and Bommisetty, *Practical Mobile Forensics*, Third Edition (Birmingham, UK: Packt, 2018).



Full disk encryption encryption

- Introduced with Lollipop (v5) Nougat(v7)
- Enforcement started in Marshmallow (allows apps to (v6)
- Applications only available after the required for full access device boots
- Backdoor bypasses exist

File-based

- Introduced with
- Direct Boot enabled run immediately)
- Credentials are to applications
- A full file system or physical is required for access – requires brute force if locked
- Two data storage locations (Per User):
  - CE: Credential Encrypted (requires user to unlock the device)
  - DE: Device Encrypted (available during Direct Boot and when unlocked)

Magnet Forensics did a great job discussing Direct Boot, file-based, and full disk encryption in this blogpost: https://www.magnetforensics.com/resources/android-nougat-forensic-analysis/. Full disk encryption is secure, but it's bulky and may slow down the functionality on some Android devices. Essentially, devices with full disk encryption require the user to enter the passcode to initiate decryption, which can take time.

File-based encryption, similar to iOS, means that every file may be encrypted with a unique key. Thus, there isn't a brute force, backdoor method to bypass the acquisition. This is bad because it often keeps us from a physical image. One of the more interesting parts of file-based encryption is the additional data storage locations. Two additional locations are created on the devices using file-based encryption:

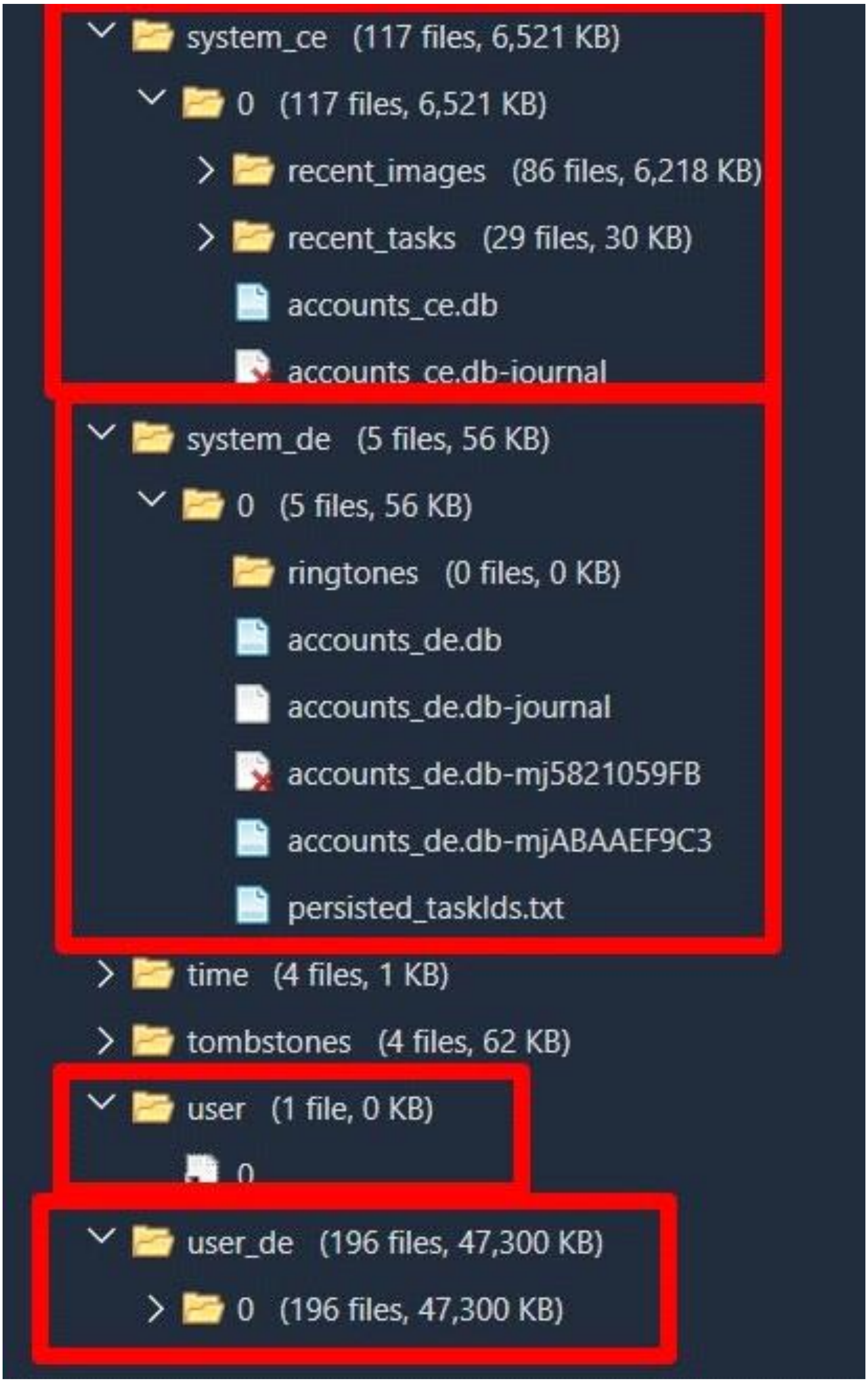
- CE: Credential Encrypted
- DE: Device Encrypted

Credential Encrypted (CE) storage includes a System directory created under the Userdata partition on the device. This data is only accessible after the user enters the proper credentials to unlock their device.[1] This makes the device more secure and enables multiple users to have accounts that are protected on the device.

Device Encrypted (DE) storage includes both a system and user partition (system_de and user_de) that includes data available during Direct Boot. This location will store data accessible after the device is unlocked as well, which is expected. The application data stored in the DE directory can only access the CE data when unlocked and decrypted. Should you stumble upon a device not using file-based encryption, the CE and DE directories will always be accessible. To determine which applications are accessible during Direct Boot, examine the APK. If “android:directBootAware=true”, then the application will be “somewhat” active prior to the device passcode being entered.[1]

Extraction options include a Full file-system (CE and DE extraction) or Partial file-system/BFU (DE only). BFU stands for “Before First Unlock” and essentially means the passcode for the device is not stashed in memory or anywhere on the device. At this phase, biometric unlocks should not work.

Direct boot, however, enables the device to boot right to the lock screen and will even allow notifications to come through to the device prior to unlocking the phone. This is much more efficient, and most users seem to prefer this method. Even though notifications from some applications are coming through to the device prior to the application data being decrypted, it is limited and depends on the application and the user-defined preferences/settings. The full passcode, PIN, or pattern must be entered before the user has access to all files on the device. Even on rooted devices, the file-based encryption stands, files are decrypted on the fly, and superuser access is granted as needed.

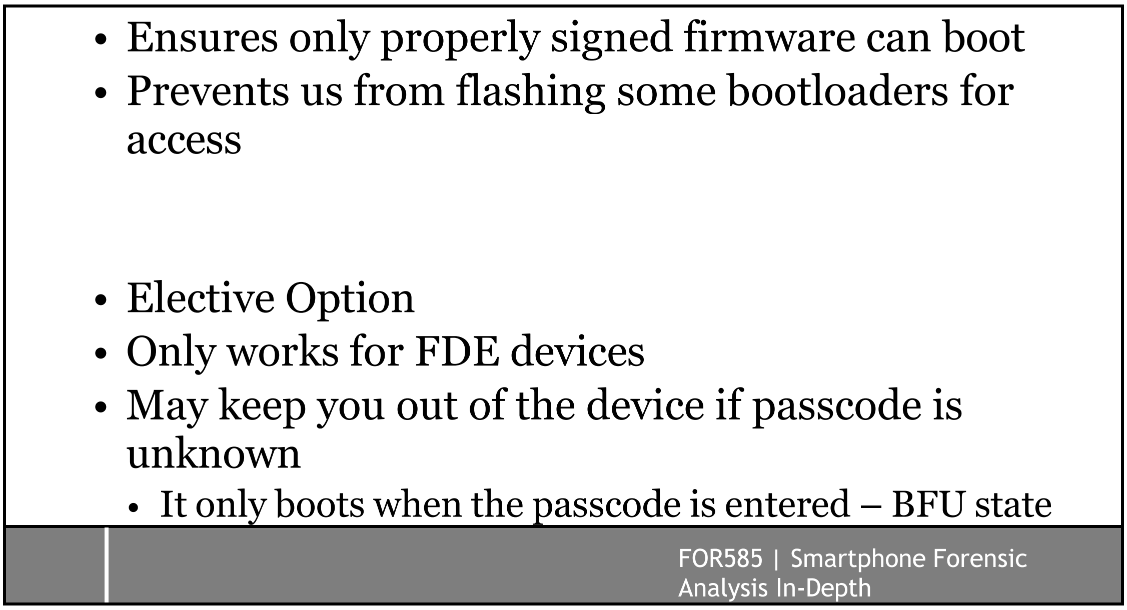


```
system_ce   (117 files, 6,521 KB)
  0   (117 files, 6,521 KB)
    recent_images   (86 files, 6,218 KB)
    recent_tasks   (29 files, 30 KB)
    accounts_ce.db
    accounts_ce.db-journal
system_de   (5 files, 56 KB)
  0   (5 files, 56 KB)
    ringtones   (0 files, 0 KB)
    accounts_de.db
    accounts_de.db-journal
    accounts_de.db-mj5821059FB
    accounts_de.db-mjABAAEF9C3
    persisted_taskIds.txt
time   (4 files, 1 KB)
tombstones   (4 files, 62 KB)
user   (1 file, 0 KB)
  0
user_de   (196 files, 47,300 KB)
  0   (196 files, 47,300 KB)
```

The screenshot below shows one of our test devices that has file-based encryption in use. For more detailed information on Android Encryption, check out the webinar provided by Heather Mahalik and Shahar Tal at https://for585.com/fde19.

Reference:
[1] https://for585.com/fbe (Encryption Explained)

- Is “on” by default on almost all phones

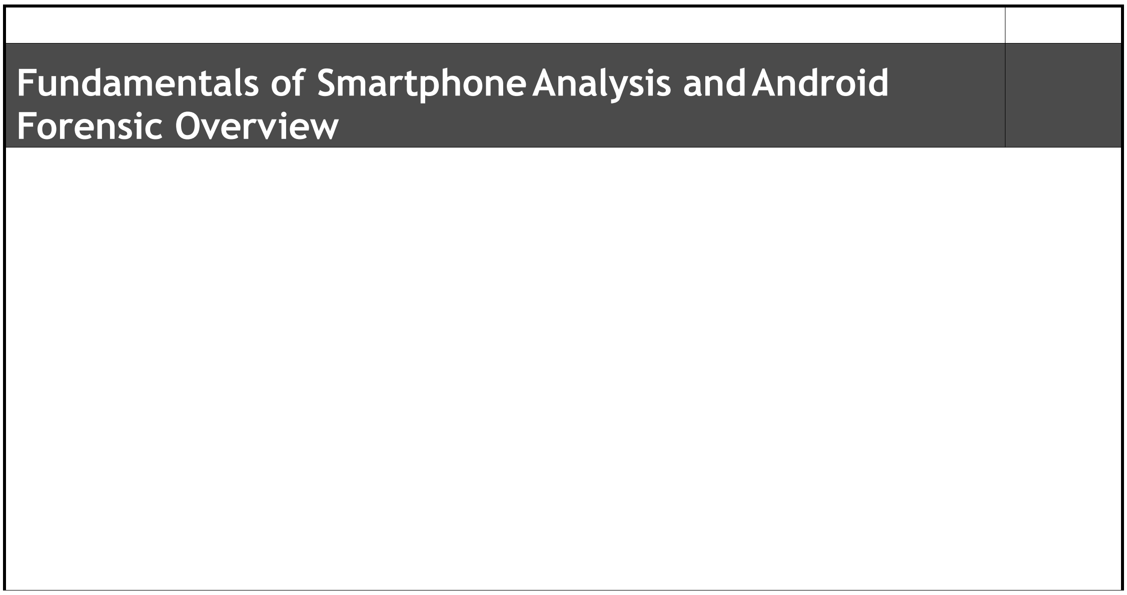


- Ensures only properly signed firmware can boot
- Prevents us from flashing some bootloaders for access

- Elective Option
- Only works for FDE devices
- May keep you out of the device if passcode is unknown
  - It only boots when the passcode is entered – BFU state

Secure boot affects almost all current smartphones. These devices, in general, ensure that only properly signed firmware is present before it boots and provides access to the files on the device. This prevents some bootloaders from being successful at flashing the device for additional access.

Secure Startup makes the device extremely secure and is offered on devices that enforce Full disk encryption. Most devices do not have it set as the default, although some rare devices may have it as a default option if you select "okay" through the out-of-the-box new phone setup experience. If set, Secure Startup can be a showstopper because we need that passcode in order to access the data in a decrypted state. You will know immediately if a device has Secure Startup because, upon powering, it will require the passcode (not a biometric) to even boot!



Fundamentals of Smartphone Analysis and Android Forensic Overview

This page intentionally left blank.

- Do you know what Google collects?

Smartphone backup files are an important part of any investigation. Users can wipe and delete data from their smartphones, but that doesn't delete or omit data that resides on a backup file on external media, the host computer, or in cloud databases until the data is synced again. Backup files are created when a user elects to save and back up his/her data and regularly when Google collects data for backups. Backup files may contain data that the user believes no longer exists. When a user deletes data from his/her device, that information still resides in the backup file and can be recovered during a forensic examination. The exception here is if a synced backup overwrites old backup data. Depending on the smartphone and backup method, multiple backup files may reside for the smartphone. Thus, the data continues to update, and all backups may contain unique data.

Creating a backup of a smartphone may be the only (and sometimes best) acquisition method, depending on the device.[1] This concept is discussed in the upcoming slides. This may be useful to your investigation should backup file creation be the only method for pulling data from a device.

Backup files outside of what we have covered so far may exist. We recommend that you apply what you have learned so far in this course. Consider the tools that supported backup files and the device for which the backup was extracted. Use that tool first. If this fails, replicate data on a test device. Should this fail, try to replicate the backup, and see what is required to access it either via cloud or acquisition. It takes effort to get results, but you can do it! Just apply what you know.

For example, Samsung Kies/Smart Switch is a backup program designed for Samsung devices. Backup files from Samsung Kies/Smart Switch may reside on external media or on host computers. The backup files have the .sbu file extension. MSAB XRY can import and parse Samsung backup files. Currently, this is the only tool capable of doing such a thing for Android-specific backups. For more information on parsing Samsung Kies/Smart Switch files using XRY, see the bonus slides within the course media files.

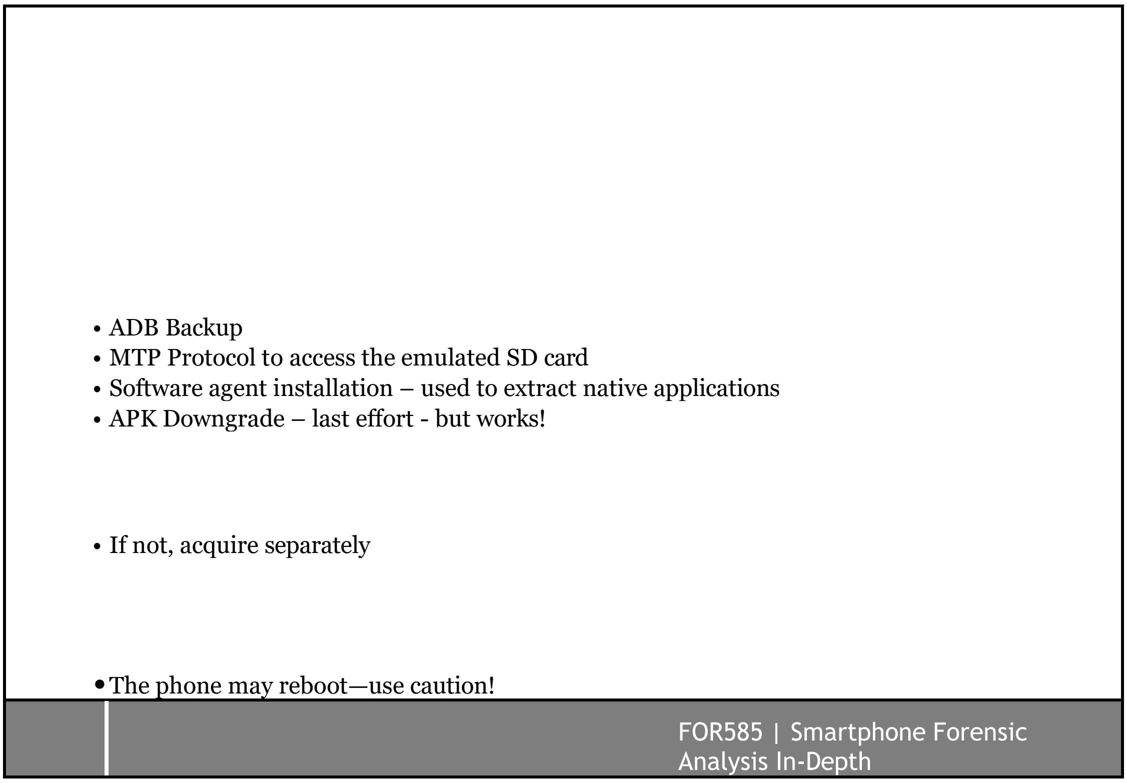
The software for Samsung Kies is similar to iTunes in that it lets the user select what they want to back up.

References:

- https://for585.com/backup (Methods for creating backup files for multiple smartphones)

The Android backup extractor can be used to examine.ab files:

- https://github.com/nelenkov/android-backup-extractor



- ADB Backup
- MTP Protocol to access the emulated SD card
- Software agent installation – used to extract native applications
- APK Downgrade – last effort - but works!

- If not, acquire separately

- The phone may reboot—use caution!

There are several methods for creating an Android backup file. Most commercial tools offer this capability, and you can even find methods for doing this for free, which we find works really well. Before adopting a method, you must realize what to expect from a backup file. Devices with Full Disk Encryption may require additional security in order to extract the backup. We will see this in an upcoming slide.

Some examiners are surprised when they select a File System acquisition and simply get a backup file. Again, File System acquisitions are Logical acquisitions or backups that provide access to raw files on the device for analysis. When you conduct an Advanced Logical, while technically not a backup, multiple extractions occur to pull the most data from the device. This section will show you some differences and what to expect when examining an Android backup that you created for forensic analysis.

During the backup, the examiner must interact with the phone to enable a full backup, or the acquisition will fail. The phone may also reboot, so make sure you are watching the device just in case it reconnects to the network.

Android Backup APK Downgrade should only be used as a last-ditch measure and only after another acquisition was obtained. This method works well, and several rely upon it. Just make sure you understand how it works before you use it.

USB Debugging enables the tools and forensic workstations to communicate with the handset. The phone must be in this state for a trust relationship to be created between the forensic workstation and the device, which is required by most commercial tools in order to acquisition to succeed. Even a locked device can be forensically acquired if USB Debugging is enabled. If the password cannot be bypassed and USB Debugging is not enabled, the examiner may not be able to forensically acquire the Android device unless physical acquisition support is available. Currently, Cellebrite is one of the few tools able to access devices physically without USB Debugging being enabled on the Android.

For devices that allow USB Debugging to be accessed via a switch (Droid X, Incredible, and other older models), it can be done by going into the Main Menu > Settings > Applications > Development. The examiner must ensure that the device remains in USB Debugging mode after the device is connected to the forensic workstation or kit. There are few things as frustrating as realizing USB Debugging was switched off during a phone reboot during the acquisition process.

For most devices now, the Build Number must be tapped seven times. Your forensic tool will instruct you on how to handle USB Debugging. You must follow every step, or your acquisition will fail. Acquiring an Android device requires patience. It's not always simple. And always double- or triple-check that USB Debugging remains switched on.

After acquisition, follow the rules of your workplace to decide if you want to put the device back in the state that it was provided to you. This is recommended if you conduct covert operations. Keep in mind, though, every single time you touch the device, you leave a trace. You just have to consider who will be doing forensics against you and if it matters that you left your mark on that Android. For most investigators, this is never a concern. For those of us who support covert ops, this is a great concern and needs to be considered.

Your VM is equipped to allow you to conduct Android extractions using ADB. Here, we are using ADB to pull a backup from the attached device. There are several ways to use ADB to extract a backup file. We are going to start as simple as possible.

- Type CMD in the Windows Search bar to obtain a Command Prompt
- Connect your device and ensure you select "allow access" on the phone, or ADB cannot communicate with it
- To ensure your device is recognized, type **adb devices**[1]
- If you see "device" you are good to go. If you see "unrecognized," you didn't "allow access" on the device
- Type **adb backup –all – shared –system –keyvalue –apk –f backup.ab**
- Your backup.ab will be created and saved to the directory you are running from, unless you specify another location

The command can be broken down as[2]: adb backup = the command

-all = backup all installed applications *(1) (2)

-shared = backup shared storage / SDcard
-system = backup system apps in -all
-apk = do backup of .apk files
-keyvalue = include apps that perform key/value backups *(3)

*(1): Of course, only apps where the "android:allowBackup" in the AndroidManifest.xml is set to "true", that's why we use APK downgrade process.
*(2): On Android 12 new restrictions were added to the ADB Backup Command.[3]
*(3): Two good references for the "keyvalue" option:
https://www.swiftforensics.com/2019/10/adb-keyvalue-backups-and-data-format.html
https://www.swiftforensics.com/2019/10/part-2-adb-keyvalue-backups-call-logs.html

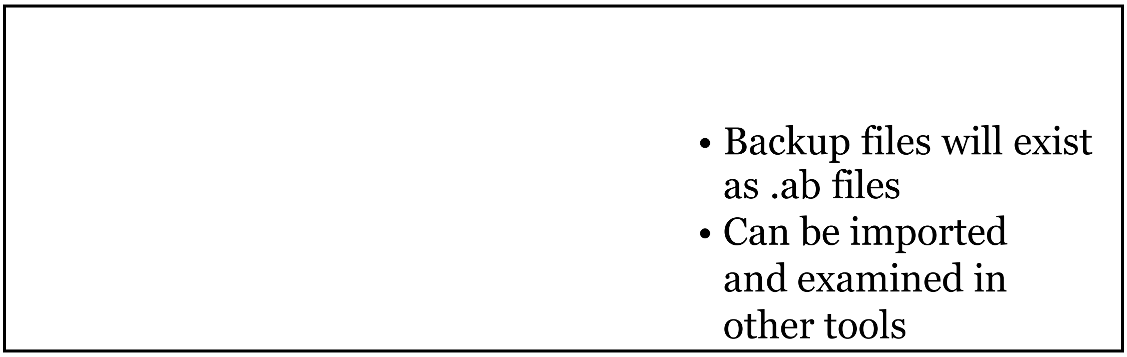
Play around with ADB on a test device. It's powerful and can accomplish a lot for you. We will cover some of the capabilities in Section 2, but it's a good idea to test it yourself.

References:

- Heather Mahalik, Rohit Tamma, and Satish Bommisetty, *Practical Mobile Forensics*, Second Edition (Birmingham, UK: Packt, 2016).
- https://for585.com/cmdadb (adb backup commands explained)
- https://for585.com/behavior (Android 12 behavior changes)

In the screenshot in the middle, we have an Android that has full disk encryption enabled. Because of this, the option to "Back Up My Data" is grayed out until a password is entered. REMEMBER the password you use if you are faced with this dilemma. After the backup is complete and you attempt to load the image into a forensic tool, it will either fail or prompt you for the password. When a tool fails, it is because it cannot handle parsing an encrypted file. Make sure you know what your tools are doing!

Additionally, this slide shows the interaction required between an Android device and the forensic tool. Make sure you pay attention, so you don't miss the option to back up the data and have to keep starting from the beginning. It can be a frustrating situation when this occurs.



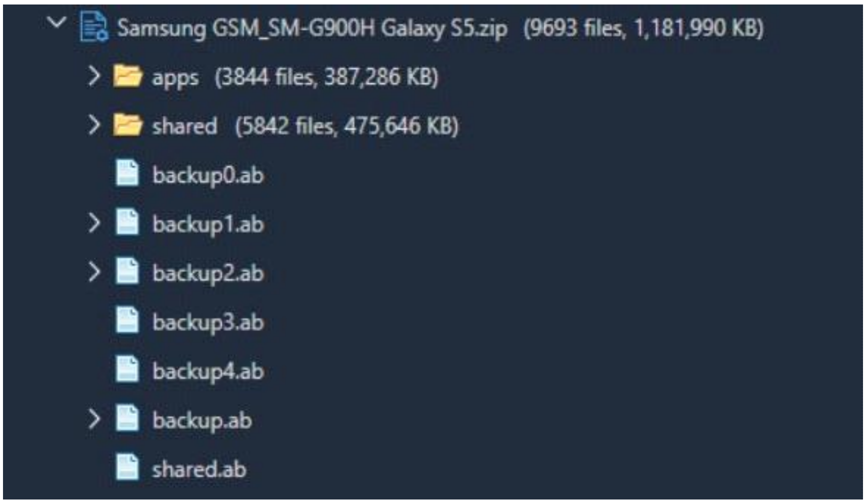
- Backup files will exist as .ab files
- Can be imported and examined in other tools

- May contain Google Drive data
- Directory structure is different
- Don't expect the same files you would from a physical or FS dump
- Use your cheat sheet to search for file names
- Get ready to dig!

Once you have an Android backup, we recommend you attempt to open it before handing the device back to the owner. You need to ensure the acquisition worked and that the data is not encrypted. Commercial support will vary for Android backups. We hope you gain experience with the tools and where they excel in the next lab to prep you for the forensic challenge in this course, as well as your upcoming investigations in your workplace.

The backup files can contain anything. One thing that may be glaringly missing are Android applications that are not permitted to be included in a backup. If this occurs, you may have to conduct a live acquisition of the device or use an emulator, if one is supported, assuming all other acquisition methods failed. This is the tough part of smartphone forensics, as we never know what we are going to get.

Additionally, you need to ensure your tool loads all of the backup “splits.” Physical Analyzer will accept them all in one load if the user properly points to the data. For this reason alone, we need you to try all the tools possible, so you know what to expect, what works best, and what makes the most sense for your lab. Andriller, a free tool available in you VM, will convert an .ab to .tar if that help with your access for analysis.



```
Samsung GSM_SM-G900H Galaxy S5.zip   (9693 files, 1,181,990 KB)
  apps   (3844 files, 387,286 KB)
  shared   (5842 files, 475,646 KB)
  backup0.ab
  backup1.ab
  backup2.ab
  backup3.ab
  backup4.ab
  backup.ab
  shared.ab
```

backup*.ab: data pulled from Android device.
shared.ab: additional internal media and SD card.

Finally, use your cheat sheet and the file names to search for data of interest. You will find that file paths differ for Android backups as compared to file system or physical extractions.

Lab 1.4

Android Backup File Analysis

FOR585 | Smartphone Forensic Analysis In-Depth