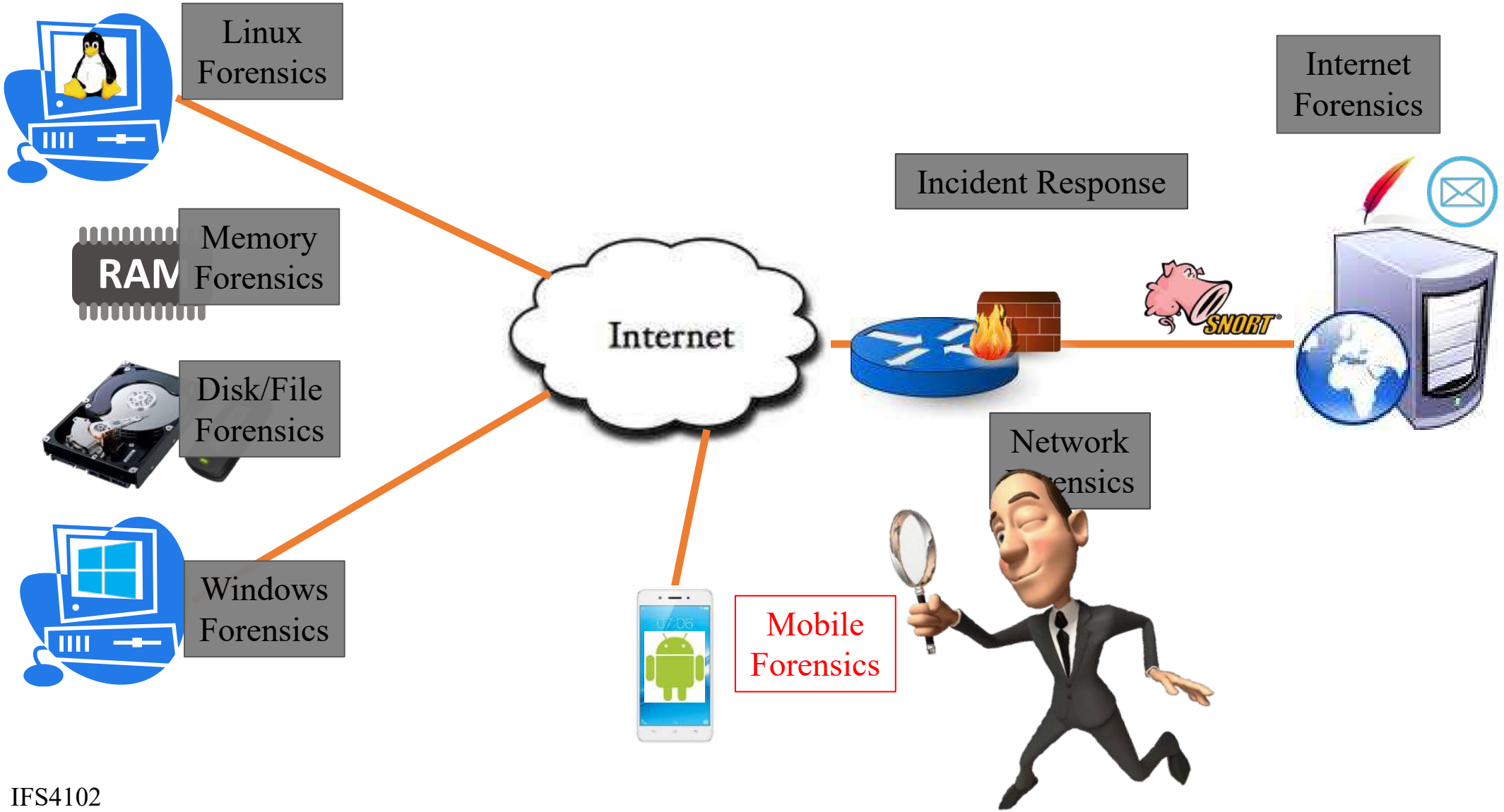
Pre-Lecture Activities

- There are ***no* pre-lecture review questions** for today
- But **you can**:
  - Ask questions about the **group project's** cases & instructions, especially **Case 1**
  - **Assignment 2**

IFS4102: Digital Forensics

Lecture 9: Mobile Forensics

This Lecture's Focus



Outline

- Mobile devices
- Mobile forensics tools
- Mobile device handling: preservation, isolation, acquisition
- Android background
- Android forensics *(main focus)*
- iOS forensics *(just a brief discussion)*
- Lab 9 exercises + Graded Lab Tasks #6 (*the last one*)

Mobile Devices

Smartphone & Mobile-Device Revolution

- Some ***fascinating statistics*** (2022):
  - The average user will tap, swipe and click their phone **2,617 times a day**
  - **47%** of US smartphone users say they **couldn't** live without their devices
  - The average time spent on smartphones is **2hrs 51mins a day**
  - Worldwide, more people now **own a cell phone** than a **toothbrush**
  - **70% of web traffic** happens on a mobile device
- Reference: https://review42.com/resources/smartphone-statistics/

Source: Wikipedia

Example of Mobile-Related Crime Case

- **“Australian lesbian lovers given life term for murder”,**
  7 March 2008, Sydney, Australia,
  https://www.reuters.com/article/idINIndia-32347120080307”:

  “Two lesbian lovers, one who drank blood as part of a vampire culture, were sentenced to life in prison on Friday for what an Australian judge said was the ‘evil’ killing of a girl they bludgeoned to death with a concrete block.

  ….

  The killers made a ***mobile phone video*** of the murder scene, …”

Computer Forensics vs Mobile Forensics

- **OS**:
  - Computers: Windows, Linux/UNIX, macOS (OS X)
  - **Mobile devices**: Android, iOS
- **Storage**:
  - Computers: mostly HDD, SSD, thumb drives
  - **Mobile devices**: non-removable flash memory, removable memory cards
- ***Unique*** to mobile devices:
  - Different **hardware** aspects: battery, touch screen, smaller screen, …
  - **SIM card**: subscribed network provider
  - "*Always-connected*" **mobile signal & data**, including SMS
  - Short-range communication **channels**: Bluetooth, NFC
  - Various portable ***sensors***: camera, microphone, GPS, …

Identity Modules (SIM Cards)

- Below is terms in ***GSM framework*** regarding a **mobile device a.k.a *Mobile Station (MS)***
- Contains 2 distinct **components**:
  - ***Mobile Equipment (ME)***
  - ***Universal Integrated Circuit Card (UICC)*:**
    - Identity module, or ***Subscriber Identity Module (SIM)***
    - A **removable** component
    - Contains essential information about the **subscriber**
    - Main purpose**: authenticating the user to the network** providing access to subscribed services
    - Also offers **storage for personal information**, e.g. phonebook entries, text messages, last numbers dialed (LND), and service-related information

SIM Cards: Form Factor & Hardware

- **Form factor**: mini SIM (2FF), micro SIM (3FF), nano SIM (4FF)

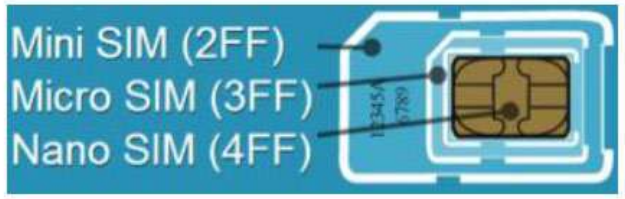


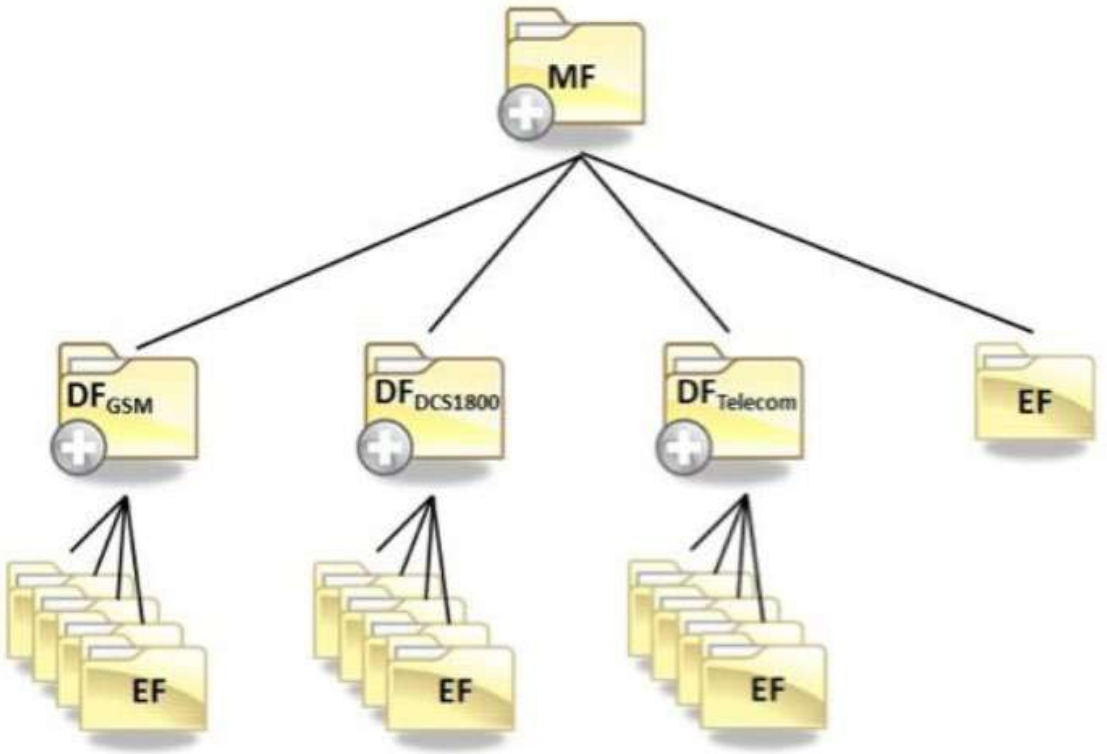
Figure 2: SIM Card Size Formats [Orm09]

From: Ayers et al., "*Guidelines on Mobile Device Forensics*", 2014

- **Hardware** features:
  - **A smart card**: contains a processor & persistent EEPROM (electronically erasable, programmable read only memory)
  - **RAM**: for program execution
  - **ROM**: for the OS, user authentication, encryption algorithms, & other apps
  - ***File system*** resides in persistent memory to store data *(elaborated next)*

SIM Cards: File System

- Three **types** of elements:
  - ***Master file (MF)***:
    the root of the file system
  - ***Directory file (DF)***:
    subordinate directory files
  - ***Elementary File (EF)***:
    files containing elementary data



MF - Master File (root and main container of DF and EF)
DF - Directory File
EF - Elementary File

Figure 3: SIM File System (GSM)

From: Ayers et al., "*Guidelines on Mobile Device Forensics*", 2014

SIM Cards: File System

- ***Data*** contained inside SIM file system:
  - **Service-related** Information:
    unique identifiers, the Integrated Circuit Card Identification (ICCID),
    the International Mobile Subscriber Identity (IMSI)
  - **Phonebook**: Abbreviated Dialing Numbers (ADN)
  - **Call information**: Last Numbers Dialed (LND)
  - **Messaging** info: SMS text messages & EMS simple multimedia messages
  - **Location info**: Location Area Information (LAI) for voice communications,
    and Routing Area Information (RAI) for data communications

What on Earth are ICCID, IMEI, IMSI, MSISDN??

- **ICCID** (Integrated Circuit Card ID):
  - Unique (unchangeable) **serial no** of a SIM/UICC
- **IMEI** (International Mobile Equipment Identity):
  - Unique (unchangeable) **serial no** for the **mobile device**: 15-digit no indicating the manufacturer, model type, country of approval for GSM devices
- **IMSI** (International Mobile Subscriber Identity):
  - Unique (assigned**) account no** for a **subscriber**:
    Mobile Country Code (MCC) + Mobile Network Code (MNC) +
    Mobile Station ID (MSID) to which the subscriber belongs
  - Stored in the SIM
- **MSISDN** (Mobile Station ISDN Number):
  - **Dialable full phone no** of a subscriber: it allows a device to be called
  - Stored in the SIM

Different Items Stored at *Different* Locations

Network Provider	Phone/Device Memory	SIM Card	Removable Memory
IMEI Phone number Calls made and received SMS/MMS made and received (no content) IMSI or ICCID Payment in/top ups Subscriber details Location info (cell site) Voicemail.	IMEI Images/Photos/Videos Audio/Recordings SMS/MMS Contacts Call Logs Internet data User files To do list Calender Games	ICCID IMSI Contacts SMS - content Last number dialled	Images/photos/videos Audio/Recordings Games Other files eg pdf, docs, xls etc,

Cellular Network: Components

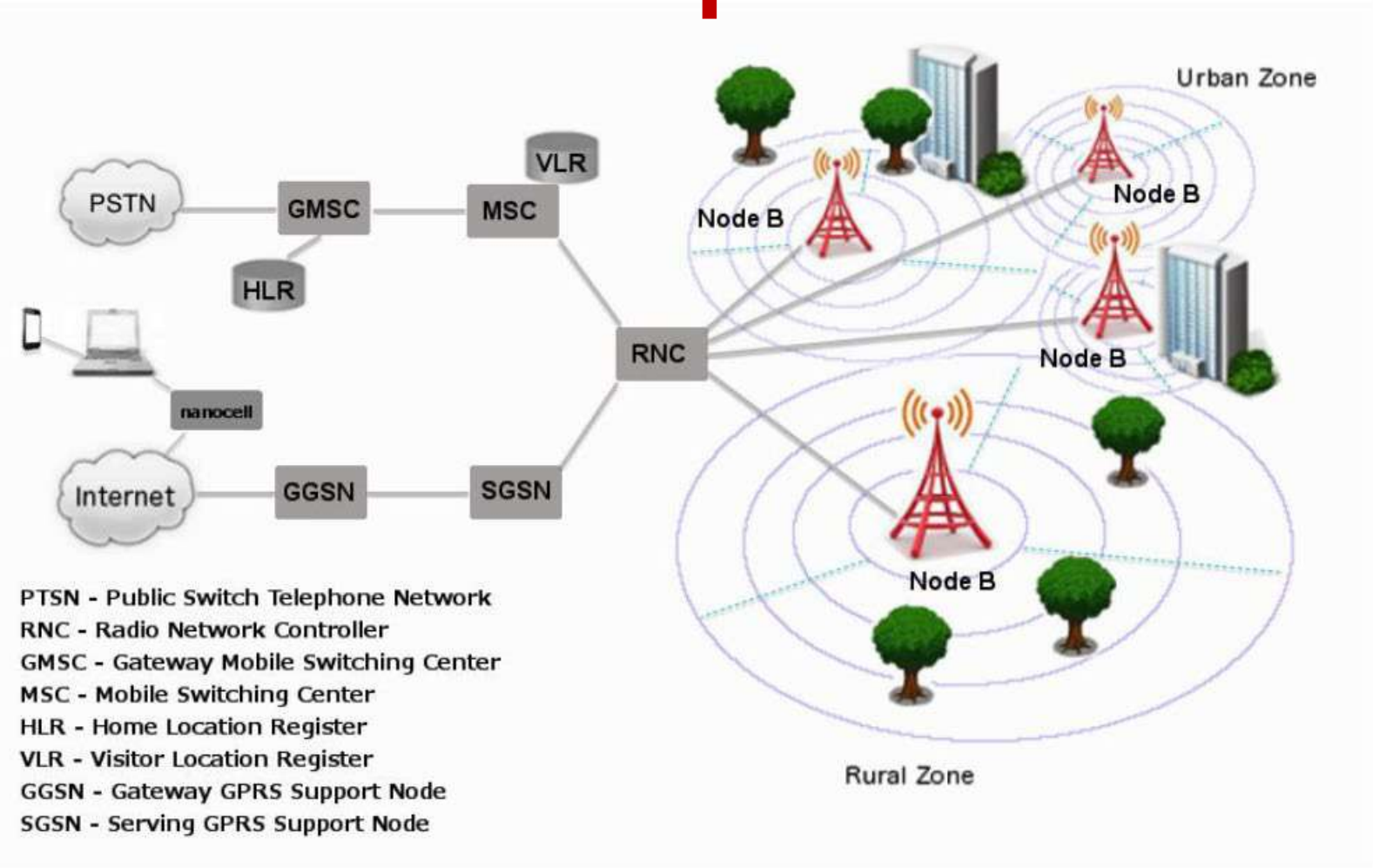


Figure 4: Cellular Network Organization

From: Ayers et al., "*Guidelines on Mobile Device Forensics*", 2014

Mobile Forensics Tools

Mobile Device Forensics: Tool Classification System

- **Tool classification** system:
  - **Objective**: to enable an examiner to easily **classify & compare** the ***extraction method*** of different tools
  - **Pyramid model**: from the bottom (Level 1) to the top (Level 5)
  - The **higher**, the more technical, invasive, time consuming & expensive

From: Ayers et al., "*Guidelines on Mobile Device Forensics*", 2014

Figure 6: Mobile Device Tool Classification System

Forensic Tool Classification System

- **Level 1 (*Manual Extraction*):**
  - Involves **viewing** the data content stored on a mobile device when **interacting** with the **device's UI**
  - Information discovered can be recorded using an **external digital camera**
  - **Drawbacks**:
    - Impossible to recover **deleted** information
    - Can be very **time consuming**
    - A device can be set to an **unknown language**

Forensic Tool Classification System

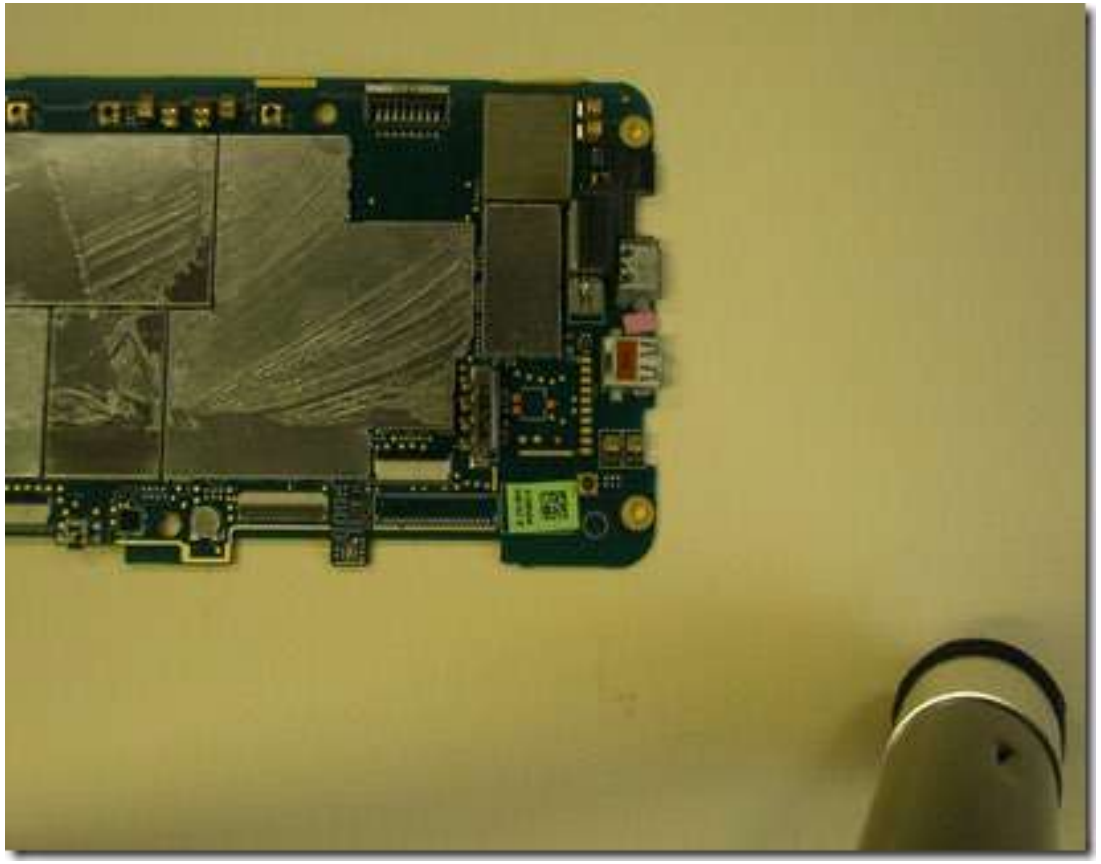
- **Level 2 (*Logical Extraction*)**:
  - Most frequently used
  - The computer **sends *commands*** to the mobile device over **the established interface**
  - **Connectivity**:
    - A **wired** connection: e.g. USB or RS-232; or
    - A **wireless** connection: e.g. IrDA, WiFi, or Bluetooth
  - The **response** (***mobile device data***) is sent back to the workstation, and presented to the forensics examiner for reporting purposes
  - ***Note***: this level covers **both** the standard "physical" (bit-by-bit) and "logical" (file system) non-volatile memory acquisitions

Forensic Tool Classification System

- **Level 3 (*Hex Dumping/JTAG Extraction*)**:
  - Allows ***direct access*** to the ***raw information*** stored in flash memory
  - ***Hex dumping***:
    - Uploads **a modified *boot loader*** into a protected area of the device's memory
    - **Connection set-up**:
      mobile device's data port ↔ a ***flasher/twister box*** ↔ workstation
    - **Flasher box**:
      - Sends commands to the device to place it in a **diagnostic mode**
      - Captures all/parts of **flash memory** & sends it to the forensic workstation

    - Challenge: the ability of a given tool to parse & decode the captured data
  - ***JTAG extraction***:
    - Communicate with a **JTAG-compliant component** by utilizing
      special purpose standalone programmer devices to probe defined test points

JTAG Illustration (On HTC EVO 4G)



From: http://lowcostwin4n6.blogspot.com/

Forensic Tool Classification System

- **Level 4 (*Chip-Off*)**:
  - Acquisition of data **directly** from a mobile device's ***flash memory***
  - Involves the ***physical removal*** of memory to extract data
  - Requires extensive training in electronic engineering & file system forensics
  - *See the illustration on the next slide*

- **Level 5 (*Micro Read*)**:
  - Involves the use of a high-powered microscope to ***view*** the ***physical state of gates***
  - The most invasive, sophisticated, technical, expensive, time consuming

Chip-Off Illustration

From:
http://www.binaryintel.com/services/jtag-chip-off-forensics/chip-off_forensics/

Sample Forensic Tools with Different Levels

Table 3: Mobile Device Forensic Tools

Tool	Acquisition Level	Network Type			Forensic Tool	Exam/Analysis	Reports	MISC
		GSM	CDMA	iDEN/TDMA				
ART	1	✓	✓	✓	✓	N/A	✓	N/A
Eclipse	1	✓	✓	✓	✓	N/A	✓	N/A
Project-A-Phone	1	✓	✓	✓	✓	N/A	✓	N/A
STE3000 FAV	1	✓	✓	✓	✓	N/A	✓	N/A
ZRT2	1	✓	✓	✓	✓	N/A	✓	N/A
Aceso†	2	✓	✓	×	✓	✓	✓	C/HW
Athena†	2	✓	✓	×	✓	✓	✓	C/HW
BitPIM	2	×	✓	×	✓[9]	×	×	×
CPA SIM Analyzer[10]‡	2	✓	×	×	✓	✓	✓	C/HW
FinalMobile Forensics	2	×	✓	×	✓	✓	✓	3PIA
iXAM[9]	2	✓	✓	×	✓	✓	✓	N/A
BlackLight	2	✓	✓	×	✓	✓	✓	3PIA
MOBILedit! Forensic‡	2	✓	✓	×	✓	✓	✓	C/HW
Oxygen Forensic Suite (Analyst)	2	✓	✓	×	✓	✓	✓	CCS
SD iPhone Recovery[12]	2	✓	✓	×	×	✓	✓	N/A
SecureView†	2	✓	✓	✓	✓	✓	✓	3PIA, C/HW
SIMIS†	2	✓	✓	×	✓	✓	✓	C/HW

Current tool list available at: http://www.dhs.nist.gov/tool_catalog/populated_taxonomy/

From: Ayers et al., "*Guidelines on Mobile Device Forensics*", 2014

Sample Forensic Tools with Different Levels

Tool	Acquisition Level	Network Type			Forensic Tool	Exam/Analysis	Reports	MISC
		GSM	CDMA	iDEN/TDMA				
SIMCon†	2	✓	×	×	✓	✓	✓	C/HW
SIMiFOR‡	2	✓	×	×	✓	✓	✓	C/HW
UFED Classic Logical‡	2	✓	✓	✓	✓	✓	✓	C/HW
UFED Touch Logical‡	2	✓	✓	✓	✓	✓	✓	C/HW
USIM Detective†	2	✓	×	×	✓	✓	✓	C/HW
WinMoFo	2	✓	✓	✓	✓	✓	✓	×
XRY Logical‡	2	✓	✓	✓	✓	✓	✓	C/HW
Zdziarski Method[11]	2	✓	✓	×	✓	×	×	N/A
CellXtract†	2/3	✓	✓	×	✓	✓	✓	C/HW
CellXtract TNT†	2/3	✓	✓	×	✓	✓	✓	CCS, C/HW
Device Seizure‡	2/3	✓	✓	✓	✓	✓	✓	3PIA, C/HW
EnCase Smartphone Examiner†	2/3	✓	✓	×	✓	✓	✓	3PIA, C/HW
Lantern	2/3	✓	✓	×	✓	✓	✓	3PIA
MPE+‡	2/3	✓	✓	✓	✓	✓	✓	3PIA, CCS, C/HW
Tarantula	2/3	✓	✓	×	✓	✓	✓	CCS, C/HW
UFED Classic Ultimate‡	2/3	✓	✓	✓	✓	✓	✓	3PIA, CCS, C/HW
UFED Touch Ultimate‡	2/3	✓	✓	✓	✓	✓	✓	3PIA, CCS, C/HW
XRY Complete‡	2/3	✓	✓	✓	✓	✓	✓	CCS, C/HW

Current tool list available at: http://www.cftt.nist.gov/tool_catalog/populated_taxonomy/

From: Ayers et al., "*Guidelines on Mobile Device Forensics*", 2014

Sample Forensic Tools with Different Levels

Tool	Acquisition Level	Network Type			Forensic Tool	Exam/Analysis	Reports	MISC	
		GSM	CDMA	iDEN/TDMA					
CDMA Workshop	3	✓	✓	×	×	×	×	×	http://www.dfinstitute.com/tool_catalog/populated_taxonomy/
Cell Phone Analyzer[12]†	3	✓	✓	×	✓	✓	✓	3PIA	
BeeProg2	4	✓			×	×	×	×	
FlashPAK III	4	✓			×	×	×	×	
NFI Memory Toolkit	4	✓			✓	✓	✓	×	
PC 3000 Flash	4	✓			✓	×	×	C/HW	
SD FlashDoctor	4	✓			✓	×	×	C/HW	
Soft-Center NAND Flash Reader	4	✓			×	×	×	×	
UP-828	4	✓			×	×	×	×	

† Denotes a tool that supports the logical acquisition of a UICC

‡ Denotes a tool that supports the logical acquisition of a UICC and the creation of a CNIC

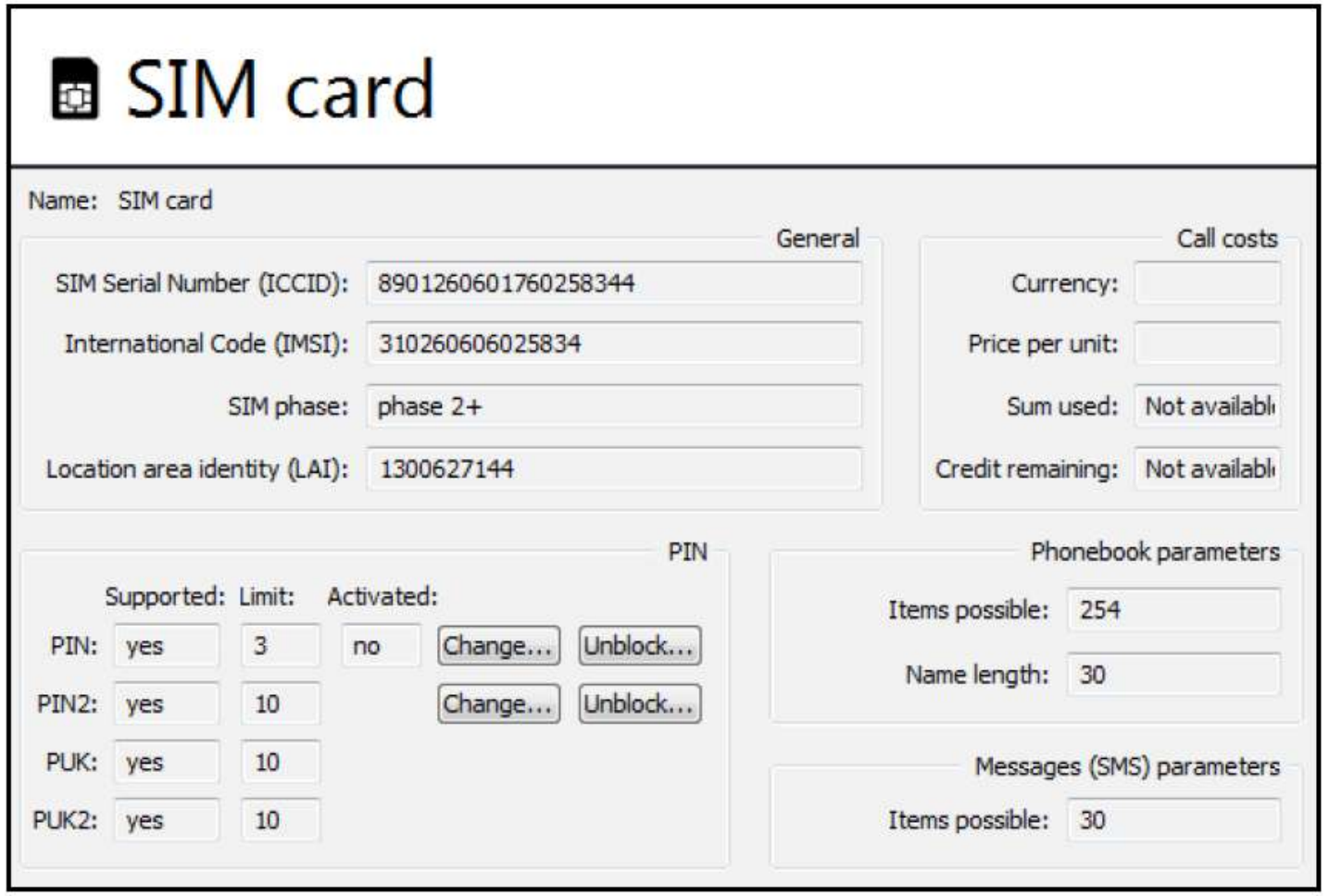
MISC: 3rd Party Tool Image Analysis (3PIA), Chinese Chipset Support (CCS), Cables/Hardware Available (C/HW)

From: Ayers et al., "*Guidelines on Mobile Device Forensics*", 2014

Additional Tools: *SIM-Access* Forensic Tools

- **SIM-access** forensic tools
- Performs **a *direct read* of a SIM card's contents** via a Personal Computer/Smart Card (PC/SC) **reader**
- **Features**:
  - Acquire various data stored on **SIM card**
  - Extract **deleted** SMS messages
  - **PIN administration** operations

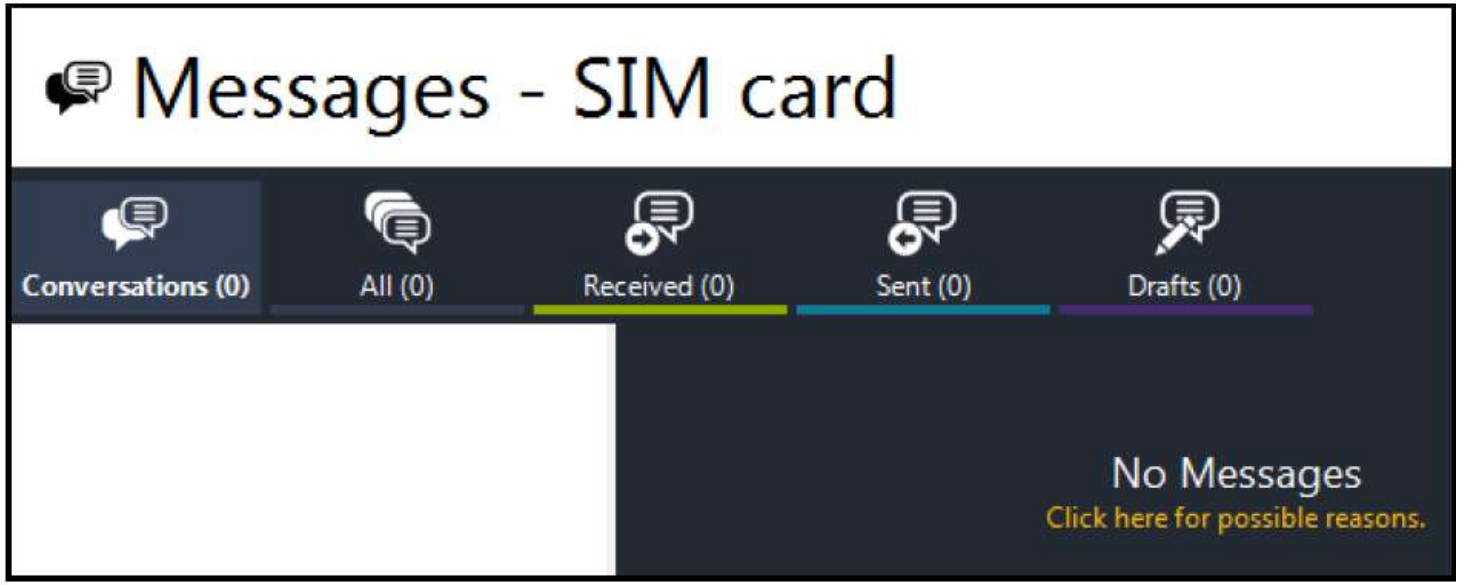
Additional Tools: *SIM-Access* Forensic Tools



SIM card extraction overview

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Additional Tools: *SIM-Access* Forensic Tools



From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Mobile Device Handling: Preservation, Isolation, Acquisition

Obstructed Mobile Devices

- ***Obstructed* device**:
  requires *successful authentication* to obtain access
- Examples:
  a device **screen-locked using** secret pattern, PIN, password, face, fingerprint, …
- Ways to **bypass** authentication mechanisms & recover data:
  - **Software-based**: e.g. software-based automated function to recover passwords
  - **Hardware-based**: cold boot attacks, JTAG, flasher boxes
  - **Investigative**: asking the owner, reviewing seized material, asking the service provider
- ***Caution***: data **auto-erases** after some failed authentication attempts

Mobile Device *Preservation*

- **Preservation**: involves the *search*, *recognition*, ***documentation*** & ***collection*** of electronic-based evidence
- ***Documenting*** **the scene**: take photos and record everything
  - **Photograph** the crime scene
  - **Collect** relevant non-electronic materials: invoices, manuals, packaging
  - **Document** the state of each digital device
- ***Isolating*** **the device**:
  - A mobile device can **remote lock** or **remote wipe**: even via a SMS
  - *How to properly deal with a mobile device?*
    Read the Scientific Working Group on Digital Evidence's (SWGDE)
    "*Best Practices for Mobile Phone Forensics*"

Mobile Device *Isolation*: Steps

- If the device is **connected** to a PC:
  pull the plug to eliminate data transfer or synchronization overwrites
- Should the media cards, SIM card & other hardware
  in the mobile device be removed? ***No!***
- **Issue**: mobile devices will be **live** and can **change** data
  if you connect to the network!
- **Isolate** the mobile device from **all radio networks**
  (e.g. cellular, WiFi, Bluetooth) by either:
  - Enable "**airplane mode**"
  - Keep the mobile device on, but radio isolated: **Faraday containers**
  - **Turn off** the mobile device

Isolating Cellular Network: Mechanisms

Isolation mechanisms are available as below:

- Shielded/signal-blocking ***containers/bags***: Faraday containers

https://www.idstronghold.com/

Isolating Cellular Network: Mechanisms

- Shielded/signal-blocking ***work areas***: a "Faraday tent"

Isolating Cellular Network: Mechanisms

- **Jamming/spoofing** devices
- **Cellular-network isolated card**:
  - ***SIM clone***: a forensically-sterile SIM, mimics the identity of the original SIM card but prevents network access to/from the handset (see https://www.mobiledit.com/sim-cloning)
- **Disabling** network service:
  - **Request** the cellular carrier operator to disable service to a mobile device

Beware of Some Custom Modifications

- **Add-on security mechanisms**:
  - Such as: login, biometric & other authentication mechanisms
  - May cause the device to **lock down** and even destroy its contents
- **Key remapping**:
  - Hardware keys may be remapped to perform a **different function** than the default
- **Geo-fencing**:
  - Automatically wipe all data when the GPS in the device determines that it has **left/entered** a specific **geographic area**
- Self-destruct upon a **specific action** carried out on the device

Data *Acquisition*: Targets

- ***Mobile device* identification**:
  - **Device model**: manufacturer logos, serial numbers
  - **Service provider**
  - **IMEI**: by keying in *#06#
- ***Device-stored* potential evidence**:
  - Subscriber & equipment identifiers
  - Date/time, language, and other settings
  - Phonebook/contact information
  - Calendar information
  - Text messages
  - Outgoing, incoming, and missed call logs

Target of Data Acquisition

- Electronic mail
- Photos
- Audio and video recordings
- Multi-media messages
- Instant messaging
- Web browsing activities
- Electronic documents
- Social media related data
- Application related data
- Location information
- Geolocation data

Target of Data Acquisition

- **Subscriber records**:
  - Customer name and address
  - Billing name and address (if other than customer)
  - User name and address (if other than customer)
  - Billing account details
  - Telephone number (MSISDN)
  - IMSI
  - UICC serial number (ICCID)
  - PIN/PUK for the UICC
  - Services allowed
- **Call records**:
  - Call Detail Records (CDRs): Maintained by the **service provider**; captures information needed to accurately bill a subscriber

Android Background

Modern Mobile OSes

- Worldwide **5G-subscription** projection (Ericsson Mobility Report): are expected to reach **1 billion** by the end of 2022, and to reach **5 billion** in 2028

Dominated by **Android** (Google) vs **iOS** (Apple)

- **Android**
  - OS kernel: **Linux**
  - **Open** ecosystem: Android Open Source Project (AOSP), various manufacturers, possible customization
- **iOS**
  - OS kernel: **Darwin** (shared with OS X)
  - **Closed** ecosystem

Android OS

- Android holds the **biggest market share** (Wikipedia):
  - May 2021: it had over **3 billion monthly active users** (the largest installed base of any OS)
  - January 2021: the Google Play Store featured over **3 million apps**
- Android devices have also become prominent targets of **malware/security attacks** too:
  - Widely reported by various security companies
  - Cautioned by the U.S. Dept. of Homeland Security in 2013
  - WikiLeaks says that CIA used Android exploits

Android System Stack

Android *Apps*

- Apps mostly written in **Java** & **Kotlin**
  - Java bytecode (`.class`) translated into **Dalvik bytecode** (`classes.dex`)
  - A possibility of **native libraries**, which are invoked via Java Native Interface (JNI)
  - All files are packaged and signed as a single **APK** file
- Two Android **runtime systems**:
  - **Dalvik VM**: prior to v5.0, JIT compilation
  - **ART (Android RunTime)**: later Android versions, ahead-of-time compilation
- **Dalvik/ART vs JVM**:
  - Register based VMs vs stack based JVM
  - Dalvik & ART differ from JDK: no Java security manager
- **Task 3 of Lab 9:**
  to do reverse-engineer and analyze `classes.dex` to infer app behavior

Android *Apps*

- Different **types** of Android apps:
  - Apps that come along with the **stock Android**
  - Apps installed by the **manufacturer**
  - Apps installed by the **wireless carrier**
  - Apps installed by the **user (3rd party apps)**
- The **internal data** of all apps (either system or user-installed apps) is automatically saved in `/data/data/<`*app-package-name*`>`
  - E.g. `/data/data/com.android.email`
  - ***Not*** accessible via adb (*discussed later*), unless a device is rooted

Android Debug Bridge: Sample Commands

Unrooted device:

```
adb shell
shell@android:/ $ cd /data/data
cd /data/data
shell@android:/data/data $ ls
ls: .: Permission denied
```

Rooted device:

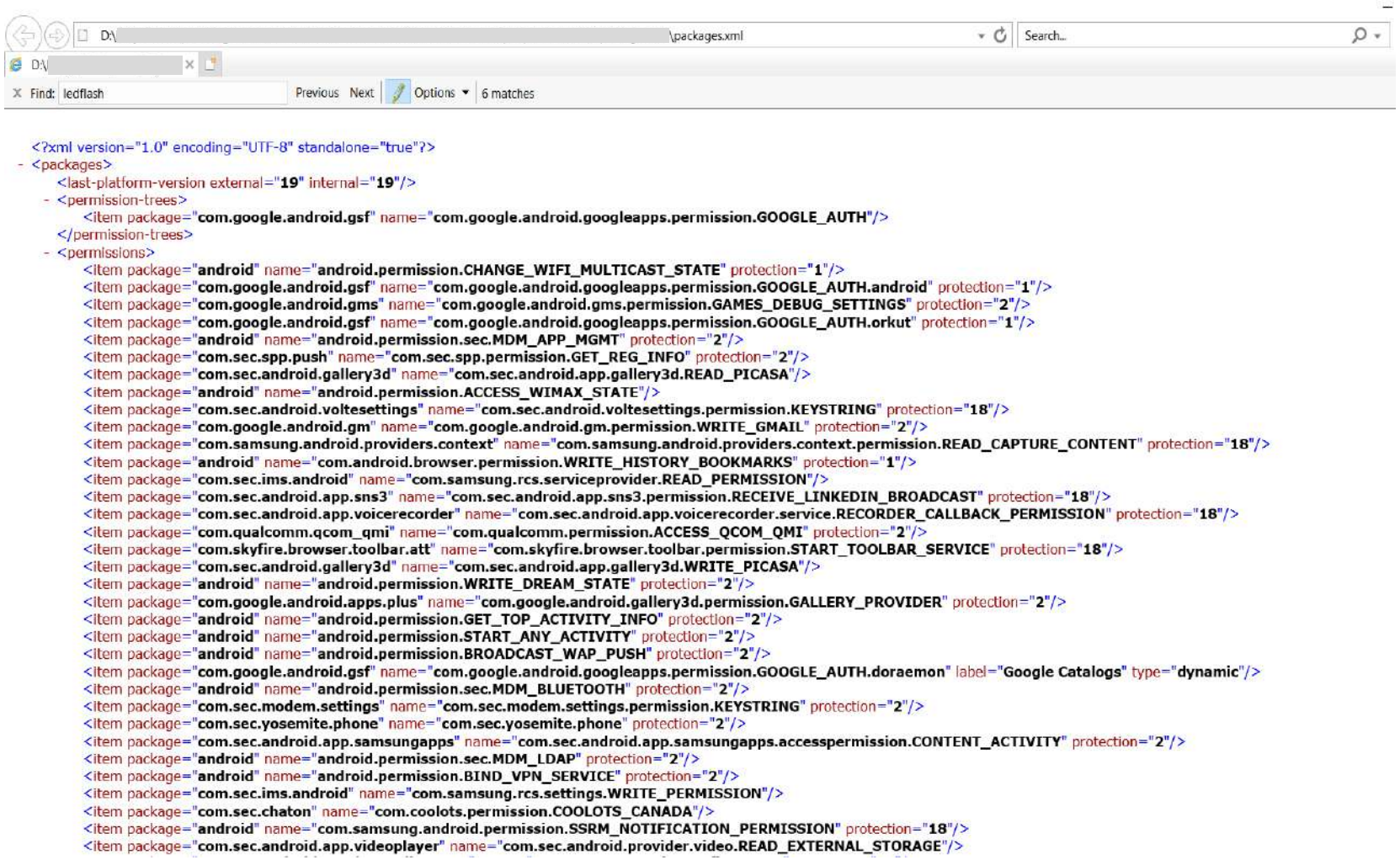
```
adb shell
shell@android:/ $ su
shell@android:/ # ls /data/data
android
 com.android.backupconfirm
 com.android.bips
 com.android.bluetooth
 com.android.bluetoothmidiservice
 com.android.calllogbackup
 com.android.camera2
 com.android.captiveportallogin
 com.android.carrierconfig
 com.android.carrierdefaultapp
 com.android.cellbroadcastreceiver
 com.android.certinstaller
 com.android.companiondevicemanager
 com.android.contacts
 com.android.cts.ctsshim
 com.android.cts.priv.ctsshim
 com.android.defcontainer
 com.android.development
```

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Some Android Security Mechanisms

- **App sandboxing**:
  - Each app runs with **unique user ID (UID) + group ID (GID)**
  - Sandboxed to its **IDs** – doesn't affect other apps/system
- **App signing**:
  - Verifies whether different apps come from the same developer
- **Permission system**:
  - **Permissions** needed to access resources/sensitive APIs (e.g. telephone function, network access, …):
    - At install time – all/nothing: prior to v6.0
    - At **runtime**: since Android 6.0 (API level 23)
  - **`/Root/System/Packages.xml`**: contains a list of installed apps and their associated permissions (see Lab 9, Task 1)

Sample /Root/System/Packages.xml




```
<?xml version="1.0" encoding="UTF-8" standalone="true"?>
- <packages>
    <last-platform-version external="19" internal="19"/>
  - <permission-trees>
      <item package="com.google.android.gsf" name="com.google.android.googleapps.permission.GOOGLE_AUTH"/>
    </permission-trees>
  - <permissions>
      <item package="android" name="android.permission.CHANGE_WIFI_MULTICAST_STATE" protection="1"/>
      <item package="com.google.android.gsf" name="com.google.android.googleapps.permission.GOOGLE_AUTH.android" protection="1"/>
      <item package="com.google.android.gms" name="com.google.android.gms.permission.GAMES_DEBUG_SETTINGS" protection="2"/>
      <item package="com.google.android.gsf" name="com.google.android.googleapps.permission.GOOGLE_AUTH.orkut" protection="1"/>
      <item package="android" name="android.permission.sec.MDM_APP_MGMT" protection="2"/>
      <item package="com.sec.spp.push" name="com.sec.spp.permission.GET_REG_INFO" protection="2"/>
      <item package="com.sec.android.gallery3d" name="com.sec.android.app.gallery3d.READ_PICASA"/>
      <item package="android" name="android.permission.ACCESS_WIMAX_STATE"/>
      <item package="com.sec.android.voltesettings" name="com.sec.android.voltesettings.permission.KEYSTRING" protection="18"/>
      <item package="com.google.android.gm" name="com.google.android.gm.permission.WRITE_GMAIL" protection="2"/>
      <item package="com.samsung.android.providers.context" name="com.samsung.android.providers.context.permission.READ_CAPTURE_CONTENT" protection="18"/>
      <item package="android" name="com.android.browser.permission.WRITE_HISTORY_BOOKMARKS" protection="1"/>
      <item package="com.sec.ims.android" name="com.samsung.rcs.serviceprovider.READ_PERMISSION"/>
      <item package="com.sec.android.app.sns3" name="com.sec.android.app.sns3.permission.RECEIVE_LINKEDIN_BROADCAST" protection="18"/>
      <item package="com.sec.android.app.voicerecorder" name="com.sec.android.app.voicerecorder.service.RECORDER_CALLBACK_PERMISSION" protection="18"/>
      <item package="com.qualcomm.qcom_qmi" name="com.qualcomm.permission.ACCESS_QCOM_QMI" protection="2"/>
      <item package="com.skyfire.browser.toolbar.att" name="com.skyfire.browser.toolbar.permission.START_TOOLBAR_SERVICE" protection="18"/>
      <item package="com.sec.android.gallery3d" name="com.sec.android.app.gallery3d.WRITE_PICASA"/>
      <item package="android" name="android.permission.WRITE_DREAM_STATE" protection="2"/>
      <item package="com.google.android.apps.plus" name="com.google.android.gallery3d.permission.GALLERY_PROVIDER" protection="2"/>
      <item package="android" name="android.permission.GET_TOP_ACTIVITY_INFO" protection="2"/>
      <item package="android" name="android.permission.START_ANY_ACTIVITY" protection="2"/>
      <item package="android" name="android.permission.BROADCAST_WAP_PUSH" protection="2"/>
      <item package="com.google.android.gsf" name="com.google.android.googleapps.permission.GOOGLE_AUTH.doraemon" label="Google Catalogs" type="dynamic"/>
      <item package="android" name="android.permission.sec.MDM_BLUETOOTH" protection="2"/>
      <item package="com.sec.modem.settings" name="com.sec.modem.settings.permission.KEYSTRING" protection="2"/>
      <item package="com.sec.yosemite.phone" name="com.sec.yosemite.phone" protection="2"/>
      <item package="com.sec.android.app.samsungapps" name="com.sec.android.app.samsungapps.accesspermission.CONTENT_ACTIVITY" protection="2"/>
      <item package="android" name="android.permission.sec.MDM_LDAP" protection="2"/>
      <item package="android" name="android.permission.BIND_VPN_SERVICE" protection="2"/>
      <item package="com.sec.ims.android" name="com.samsung.rcs.settings.WRITE_PERMISSION"/>
      <item package="com.sec.chaton" name="com.coolots.permission.COOLOTS_CANADA"/>
      <item package="android" name="com.samsung.android.permission.SSRM_NOTIFICATION_PERMISSION" protection="18"/>
      <item package="com.sec.android.app.videoplayer" name="com.sec.android.provider.video.READ_EXTERNAL_STORAGE"/>
```

Android APK File

- App packaged in a single **APK** file:
  - **A JAR (zip) file** format generated by jarsigner tool
  - **Signed**: can be self-signed
    - So users may not necessarily trust the signature
    - Useful more for **app updates & sharing**
  - **Content:**
    - **App manifest file**
    - **Dex files** (for Dalvik/ART VM)
    - Resources
    - …

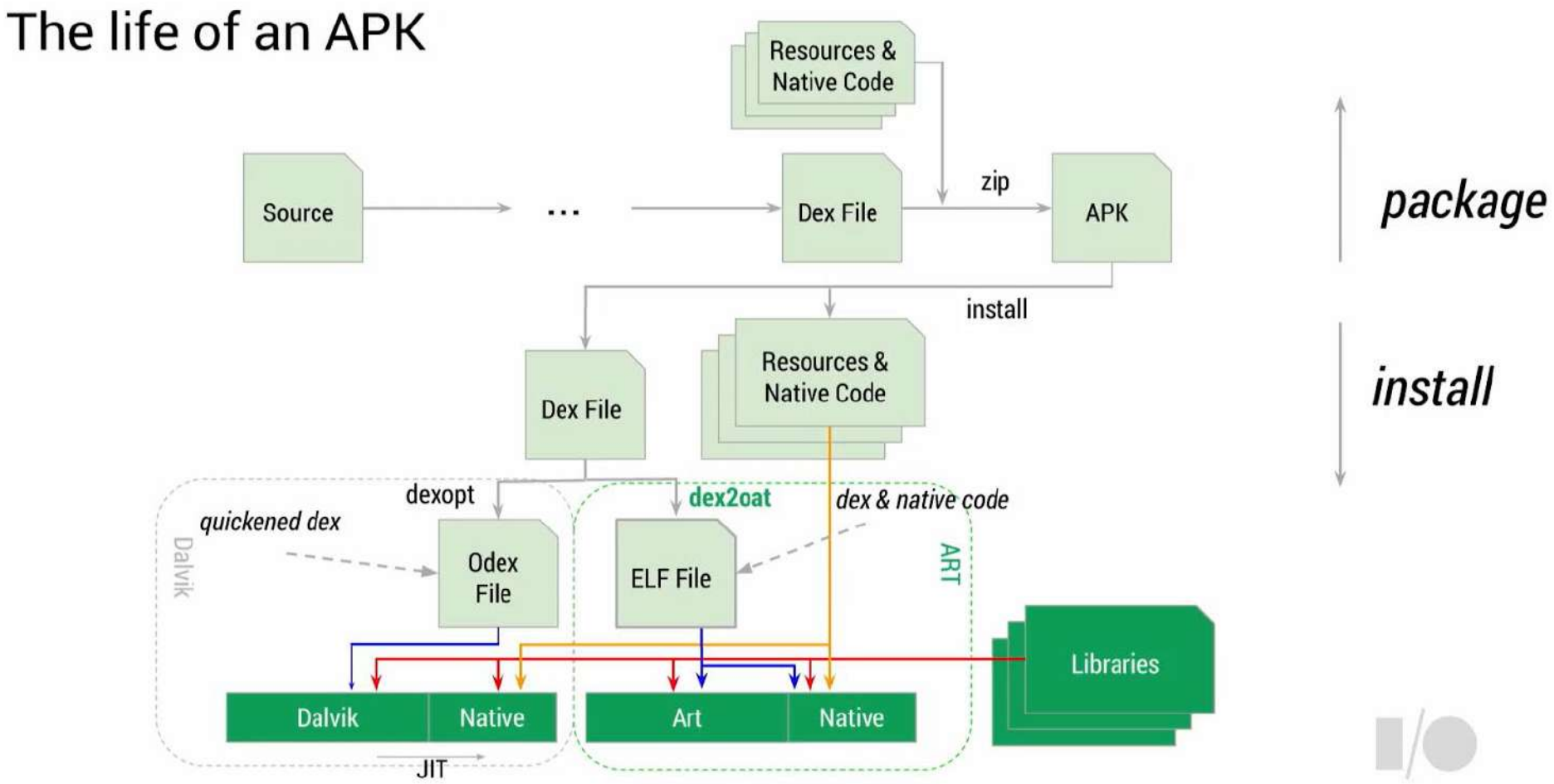
Android App Manifest File

- An app's **XML file component** that declares:
  - **Java package** for the app: serves as a **unique identifier** for the app
  - App **components**
  - Required ***permissions***:
    - Permissions that **the app must have** in order to access protected parts of the API and interact with other apps
    - Permissions that **other apps are required to have** in order to interact with the app's components
  - Minimum level of the **Android API** required by the app
  - …
- Read https://developer.android.com/guide/topics/manifest/manifest-intro.html

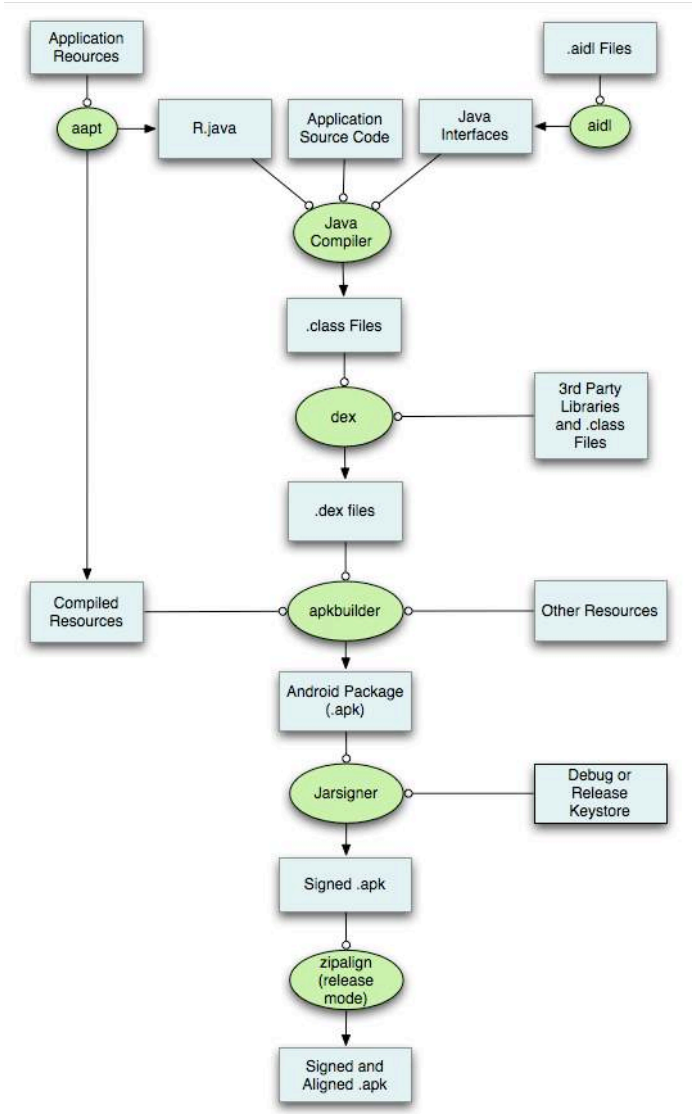
Sample Android App Manifest

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<manifest xmlns:android=http://schemas.android.com/apk/res/android
  package="com.example.browserapp" platformBuildVersionCode="21"
  platformBuildVersionName="5.0.1-1624448">

  <uses-permission android:name="android.permission.INTERNET"/>

  <application android:allowBackup="true" android:debuggable="true"
android:icon="@drawable/ic_launcher" android:label="@string/app_name"
android:theme="@style/AppTheme">

    <activity android:label="@string/app_name" android:name=".BrowserActivity">|
       <intent-filter>
          <action android:name="android.intent.action.MAIN"/>
          <category android:name="android.intent.category.LAUNCHER"/>
       </intent-filter>

    </aZctivity>

  </application>

</manifest>
```

App Overview: Building & Installation



App Overview: *App Building* in More Detail

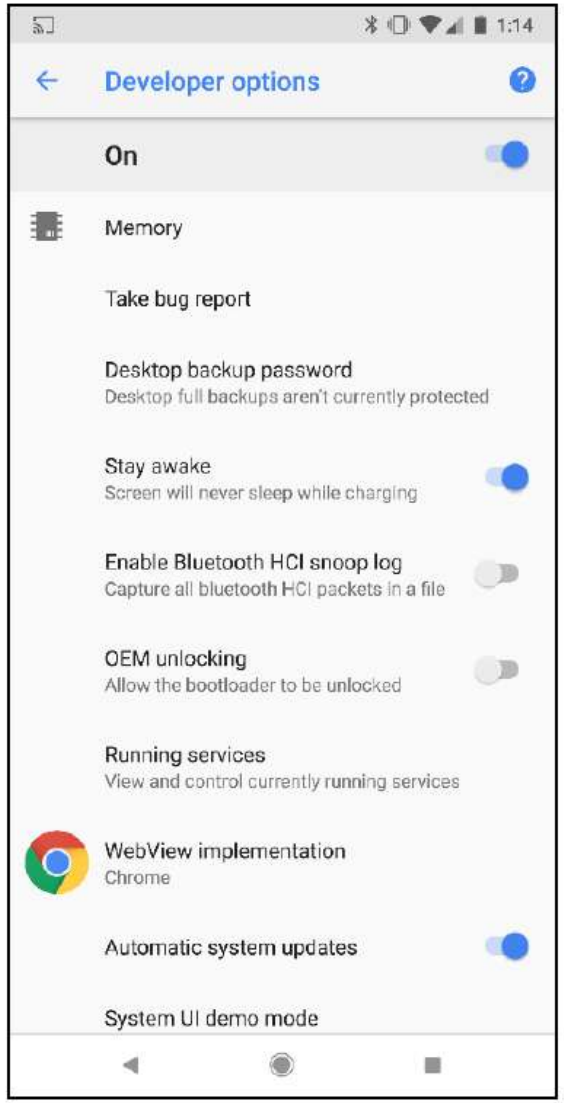


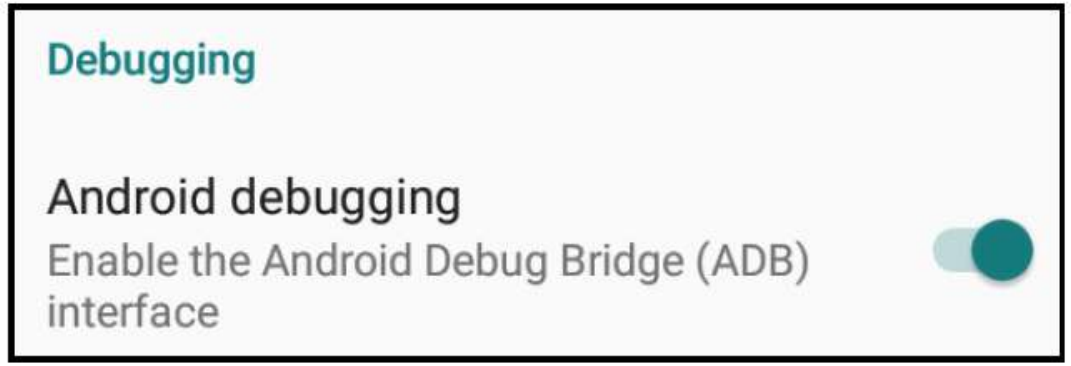
Source:
developer.android.com

Android Debug Bridge (adb)

- A CLI tool to **communicate** with Android device/emulator:
  - Client (adb.exe)
  - Server (on development machine)
  - adbd daemon (on Android device)
- For use, enable **adb debugging** on device via **Developer Options**: Settings | About Phone (or Settings | System | About Phone on Android 8.0 or higher), and tap on the Build number 7 times
- Relevant **forensics tasks** using adb:
  - Determine the path of an installed app
  - Pull the app
  - Push files for rooting purposes

Developer Options & adb





From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Android Debug Bridge (adb)

- Useful **adb commands**:
  - **Debugging**: `adb kill-server`, `adb start-server`, `adb devices`
  - **Package management**:
    `adb shell pm list packages`,
    `adb shell pm path` <*app-name*>,
    `adb install [-r]` <*apk-name*>,
    `adb uninstall [-k]` <*app-name*>
  - **File management**:
    `adb push` <*local*> <*remote*>,
    `adb pull` <*remote*> <*local*>
  - **Logcat**: `adb logcat [*: V|D|I|W|E|F|S]`
    → Verbose, Debug, Info, Warning, Error, Fatal, Silent
- More info on adb: http://adbshell.com/

Android Debug Bridge: Sample Commands

```
adb devices
List of devices attached
4df16ac5115e4e04        device
7f1c86454445606e         device


adb install C:\test.apk
Success


adb.exe pull /sdcard/Pictures/MyFolder/Sample.png C:\temp
[100%] /sdcard/Pictures/MyFolder/Sample.png


adb push C:\temp\test.png /sdcard/Pictures
[100%] /sdcard/Pictures/test.png
```

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Android Debug Bridge: Sample Commands

```
adb.exe logcat
--------- beginning of system
 09-17 10:04:52.463 2477 2477 I vold : Vold 3.0 (the awakening) firing up
 09-17 10:04:52.463 2477 2477 V vold : Detected support for: exfat ext4
f2fs ntfs vfat
 09-17 10:04:52.475 2477 2482 D vold : e4crypt_init_user0
 09-17 10:04:52.475 2477 2482 D vold : e4crypt_prepare_user_storage for
volume null, user 0, serial 0, flags 1
 09-17 10:04:52.475 2477 2482 D vold : Preparing: /data/system/users/0
 09-17 10:04:52.476 2477 2482 D vold : Preparing: /data/misc/profiles/cur/0
 09-17 10:04:52.476 2477 2482 D vold : Preparing: /data/system_de/0
 09-17 10:04:52.477 2477 2482 D vold : Preparing: /data/misc_de/0
 09-17 10:04:52.477 2477 2482 D vold : Preparing: /data/user_de/0
 09-17 10:04:52.477 2477 2482 D vold : e4crypt_unlock_user_key 0 serial=0
token_present=0
 09-17 10:04:52.712 2477 2480 D vold : Disk at 7:64 changed
 09-17 10:04:52.933 2590 2590 I android.hardware.wifi@1.0-service: Wifi Hal
is booting up...
 09-17 10:04:53.023 2619 2619 I installd: installd firing up
 09-17 10:04:53.166 2627 2627 I wificond: wificond is starting up...
 09-17 10:04:53.285 2626 2666 I /system/bin/storaged: storaged: Start
 09-17 10:04:55.120 2760 2760 I SystemServer: InitBeforeStartServices
 09-17 10:04:55.122 2760 2760 I SystemServer: Entered the Android system
server!
 09-17 10:04:55.358 2760 2760 I SystemServer: StartServices
 09-17 10:04:55.358 2760 2760 I SystemServer: Reading configuration...
 09-17 10:04:55.358 2760 2760 I SystemServer: ReadingSystemConfig
 09-17 10:04:55.359 2760 2760 I SystemServer: StartInstaller
 09-17 10:04:55.360 2760 2760 I SystemServiceManager: Starting
com.android.server.pm.Installer
```

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Android Device's *Non-Volatile Memory*

- Non-volatile memory :
  internal/built-in + removable memory cards
- **Linux system device** defaults to the first hard drive (`/dev/hd0`)
- ***Memory Technology Device (MTD)***:
  - Used to provide an **interface** between the Linux OS & the physical flash device
  - Is needed since flash memory devices are *not seen* as character or block devices
- For non-volatile **memory analysis**, we need to know: Android device **partitioning**, **file system type**, (common) **file hierarchy**

Common *Partitions* in Android

- BOOT: stores information & files required for the phone to boot
- CACHE: stores frequently-accessed data & other files, e.g. recovery logs and update-packages downloaded over-the-air
- **RECOVERY**: a recovery partition, which allows the device to boot into the recovery console so that phone **updates** & other **maintenance operations** are performed
- SYSTEM: stores all major components other than the kernel & RAMDisk
- **USERDATA**: data partition that stores the device's internal storage for application data

Sample Partitions Contained in a Device

```
j7xelte:/dev/block/platform/13540000.dwmmc0/by-name # ls -l
total 0
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 BOOT -> /dev/block/mmcblk0p10
lrwxrwxrwx 1 root root 20 2018-09-19 09:21 BOTA0 -> /dev/block/mmcblk0p1
lrwxrwxrwx 1 root root 20 2018-09-19 09:21 BOTA1 -> /dev/block/mmcblk0p2
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 CACHE -> /dev/block/mmcblk0p21
lrwxrwxrwx 1 root root 20 2018-09-19 09:21 CARRIER -> /dev/block/mmcblk0p8
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 CDMA-RADIO -> /dev/block/mmcblk0p13
lrwxrwxrwx 1 root root 20 2018-09-19 09:21 CPEFS -> /dev/block/mmcblk0p4
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 CP_DEBUG -> /dev/block/mmcblk0p23
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 DNT -> /dev/block/mmcblk0p16
lrwxrwxrwx 1 root root 20 2018-09-19 09:21 EFS -> /dev/block/mmcblk0p3
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 HIDDEN -> /dev/block/mmcblk0p22
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 OTA -> /dev/block/mmcblk0p12
lrwxrwxrwx 1 root root 20 2018-09-19 09:21 PARAM -> /dev/block/mmcblk0p9
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 PERSDATA -> /dev/block/mmcblk0p18
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 PERSISTENT -> /dev/block/mmcblk0p17
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 RADIO -> /dev/block/mmcblk0p14
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 RECOVERY -> /dev/block/mmcblk0p11
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 RESERVED2 -> /dev/block/mmcblk0p19
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 SYSTEM -> /dev/block/mmcblk0p20
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 TOMBSTONES -> /dev/block/mmcblk0p15
lrwxrwxrwx 1 root root 21 2018-09-19 09:21 USERDATA -> /dev/block/mmcblk0p24
lrwxrwxrwx 1 root root 20 2018-09-19 09:21 m9kefs1 -> /dev/block/mmcblk0p5
lrwxrwxrwx 1 root root 20 2018-09-19 09:21 m9kefs2 -> /dev/block/mmcblk0p6
lrwxrwxrwx 1 root root 20 2018-09-19 09:21 m9kefs3 -> /dev/block/mmcblk0p7
```

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Various Android-Supported File Systems

- **Flash memory file systems**:
  - Extended File Allocation Table (exFAT)
  - Flash Friendly File System (F2FS)
  - Journal Flash File System version 2 (JFFS2)
  - Yet Another Flash File System version 2 (YAFFS2)
  - Robust File System (RFS)
- **Media-based file systems**:
  - EXTended filesystem: EXT2/EXT3/EXT4
  - FAT (File Allocation Table): FAT12, FAT16, FAT32
  - VFAT (Virtual File Allocation Table)
- Pseudo filesystems: cgroup, rootfs, Procfs, sysfs, tmpfs

Android File Hierarchy & Some Directories

- `/data/data`: contains the **private data** of all apps
- `storage`: holds **SD card contents**, typically has Android, DCIM & Downloads folders
- `system`: contains libraries, system binaries, and other **system-related** files

```
j7xelte:/ # ls
acct                      init                      mnt                          res
bugreports                init.baseband.rc          nonplat_file_contexts        root
cache                     init.environ.rc           nonplat_hwservice_contexts   sbin
charger                   init.power.rc             nonplat_property_contexts    sdcard
config                    init.rc                   nonplat_seapp_contexts       sepolicy
cpefs                     init.rilchip.rc           nonplat_service_contexts     storage
d                         init.samsungexynos7870.rc oem                          sys
data                      init.samsungexynos7870.usb.rc plat_file_contexts       system
default.prop              init.target.rc            plat_hwservice_contexts      ueventd.rc
dev                       init.usb.configfs.rc      plat_property_contexts       ueventd.samsungexynos7870.rc
efs                       init.usb.rc               plat_seapp_contexts          vendor
etc                       init.wifi.rc              plat_service_contexts        vndservice_contexts
fstab.samsungexynos7870   init.zygote32.rc          proc
```

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Android File Hierarchy & Some Directories

```
j7xelte:/data # ls -1
adb
anr
app
app-asec
app-ephemeral
app-lib
app-private
backup
bootchart
cache
camera
dalvik-cache
data
drm
lineageos_updates
local
lost+found
media
```

```
mediadrm
misc
misc_ce
misc_de
ota
ota_package
property
resource-cache
ss
ssh
system
system_ce
system_de
tombstones
user
user_de
vendor
```

Content of data partition of an Android device

From: Oleg Skulkin et al., "*Learning Android Forensics: nalyze Android devices with the latest forensic tools and techniques*", 2nd edition

Android File Hierarchy & Some Directories

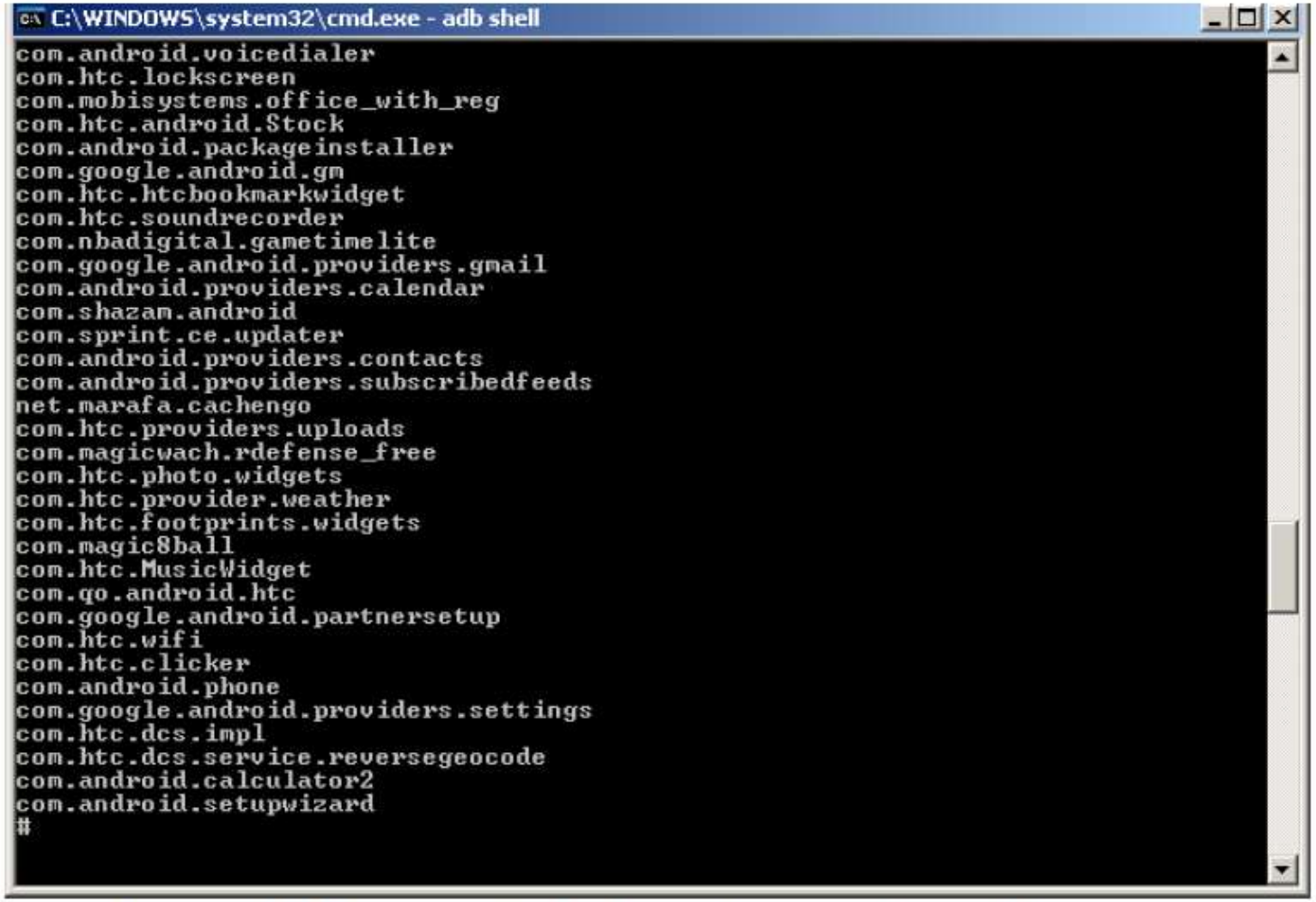


Figure 12. Contents of the /data/data directory.

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Break!

Android Forensics

Device *Seizure*: Possible Issue

- **Obstructed** Android device, i.e. a which is **screen-locked** using:
  - Secret pattern
  - PIN
  - Password
  - Smart lock: trusted face/fingerprint/ voice/location/nearby-device
- Screen-lock needs to be **bypassed**:
  - It requires retrieving or removing **a file** from the device:
    `/data/system/gesture.key` (pattern lock),
    `/data/system/password.key|gatekeeper.pattern.key` (PIN/password)
  - Several **software-based mechanism** are possible for the file access:
    adb, booting into a custom Recovery Mode, JTAG/chip-off
    → *see the next few slides*

Bypassing Screen-Lock (Software-based)

- **adb**:
  - Requires root & USB debugging
- Booting into a **custom Recovery Mode**:
  - Does not require root, USB debugging
  - But requires an **unlocked bootloader**
  - **Won't work** on devices with encrypted `userdata` partition
  - *(More on this Recovery Mode later)*
- **JTAG/chip-off**:
  - Does not require any specific device settings or options
  - But highly technical
  - Still **won't work** on devices with encrypted `userdata` partition

Device *Seizure*: Follow-Up Steps

- Additional **steps** to be done if an Android device is **unobstructed**
- Change the **device's settings** to allow *greater access* to it:
  - **Enable USB debugging**
  - Enable the "**Stay Awake**" setting: under Settings | Developer options
  - Increase **screen timeout**: Settings | Display | Screen Timeout
- Remember to ***isolate*** an Android device (discussed earlier):
  - **Android Device Manager** & several **3rd party apps** can remote-wipe or remote-lock
  - **Mobile Device Management** (**MDM**) software: used by companies to manage corporate devices, can remote-wipe via SMS

Data *Acquisition & Extraction* on Android Device

- Perform **data acquisition** on:
  - **Removable** memory card
  - **Internal** memory:
    - On a **rooted/rootable** device: volatile & non-volatile memory
    - On an **unrooted** device: *a rooting is required first*
- Acquisition **steps** are described on the next few slides
- Tool-based **automated solutions** are possible too:
  - Commercial solutions: Oxygen Forensics, UFED 4PC/Touch, *Magnet AXIOM (30-day free trial is available)*
  - Free tool: Magnet ACQUIR (https://www.magnetforensics.com/resources/magnet-acquire/)

Data Acquisition of *Removable* Memory Card

- Sample content of a **memory card**:

```
generic_x86:/sdcard # ls -l
total 40
drwxrwx--x 2 root sdcard_rw 4096 2018-09-16 13:22 Alarms
drwxrwx--x 3 root sdcard_rw 4096 2018-09-16 13:22 Android
drwxrwx--x 2 root sdcard_rw 4096 2018-09-16 13:22 DCIM
drwxrwx--x 2 root sdcard_rw 4096 2018-09-16 13:22 Download
drwxrwx--x 2 root sdcard_rw 4096 2018-09-16 13:22 Movies
drwxrwx--x 2 root sdcard_rw 4096 2018-09-16 13:22 Music
drwxrwx--x 2 root sdcard_rw 4096 2018-09-16 13:22 Notifications
drwxrwx--x 2 root sdcard_rw 4096 2018-09-16 13:22 Pictures
drwxrwx--x 2 root sdcard_rw 4096 2018-09-16 13:22 Podcasts
drwxrwx--x 2 root sdcard_rw 4096 2018-09-16 13:22 Ringtones
```

The content of /storage/self/primary (/sdcard is a symlink)

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Data Acquisition of *Removable* Memory Card

- Easy, you can use acquisition tools like **FTK Imager** (+ a hardware blocker & perhaps a card reader)

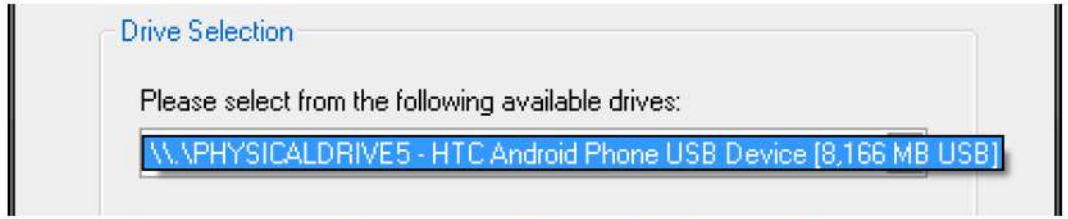


Figure 4. FTK Imager "Drive Selection" screen.

⊟ General	
Name	sdcard2.001
Sector count	15949824
⊟ MD5 Hash	
Computed hash	e3cbc7b88bc00cbc30227c528f31ade2
Report Hash	e3cbc7b88bc00cbc30227c528f31ade2
Verify result	Match
⊟ SHA1 Hash	
Computed hash	6c86800c1841e4a0aa80d1783248660d7ff06594
Report Hash	6c86800c1841e4a0aa80d1783248660d7ff06594
Verify result	Match

Figure 5. FTK Imager image summary screen.

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Data Acquisition & Parsing of *Internal* Memory

- General steps:
  1. Device needs to be first **unobstructed**
  2. **Root** the device (see Sun et al., "*Android Rooting: Methods, Detection, and Evasion*", SPSM '15)
  3. **Perform a data acquisition**: **non-volatile memory, volatile memory**
  4. **Parse/extract** the acquired **data**:
     - Parse **text (based) setting files**, e.g. file **`/Root/System/Packages.xml`** (see Lab 9, Task 1); also **shared preferences** of installed apps, which are usually stored in `/data/data/<app-package-name>/shared_prefs`
     - Extract **apps' data**, including **SQLite database** files (see Lab 9, Task 2)
     - Reverse engineer & decompile **apps**: `classes.dex` (see Lab 9, Task 3), …

Sample Shared Preferences Files

Name	Size	Type	Date Modified
code_cache	4	Directory	07.02.2016 12:06:09
no_backup	4	Directory	07.02.2016 12:08:10
files	4	Directory	04.09.2018 13:53:59
databases	4	Directory	13.09.2018 8:46:37
cache	4	Directory	01.10.2018 23:10:43
shared_prefs	4	Directory	02.10.2018 2:12:06
lib	1	Symbolic Li...	01.10.2018 14:10:33

Name	Size	Type	Date Modified
UnifiedEmail.xml	1	Regular File	07.02.2016 12:0...
AndroidMail.Main.xml	1	Regular File	07.02.2016 12:0...
MailAppProvider.xml	1	Regular File	07.02.2016 12:0...

Contents of the shared_prefs folder of the Android email app

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Analyzing SQLite Database Files

- Locate **SQLite database files** stored by apps on a device
- Examples:

Name	Size	Type	Date Modified
code_cache	4	Directory	07.02.2016 12:06:09
no_backup	4	Directory	07.02.2016 12:08:10
files	4	Directory	04.09.2018 13:53:59
databases	4	Directory	13.09.2018 8:46:37
cache	4	Directory	01.10.2018 23:10:43
shared_prefs	4	Directory	02.10.2018 2:12:06
lib	1	Symbolic Li...	01.10.2018 14:10:33

Internal storage of the Android YouTube app

Name	Size	Type	Date Modified
EmailProvider.db	132	Regular File	07.02.2016 12:08:22
EmailProvider.db-journal	0	Regular File	07.02.2016 12:08:22
EmailProviderBody.db	24	Regular File	07.02.2016 12:08:22
EmailProviderBody.db-journal	0	Regular File	07.02.2016 12:08:22

SQLite files present under the databases folder of the Android browser app

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

- Some SQLite database **browsers** are available (see Lab 9, Task 2)

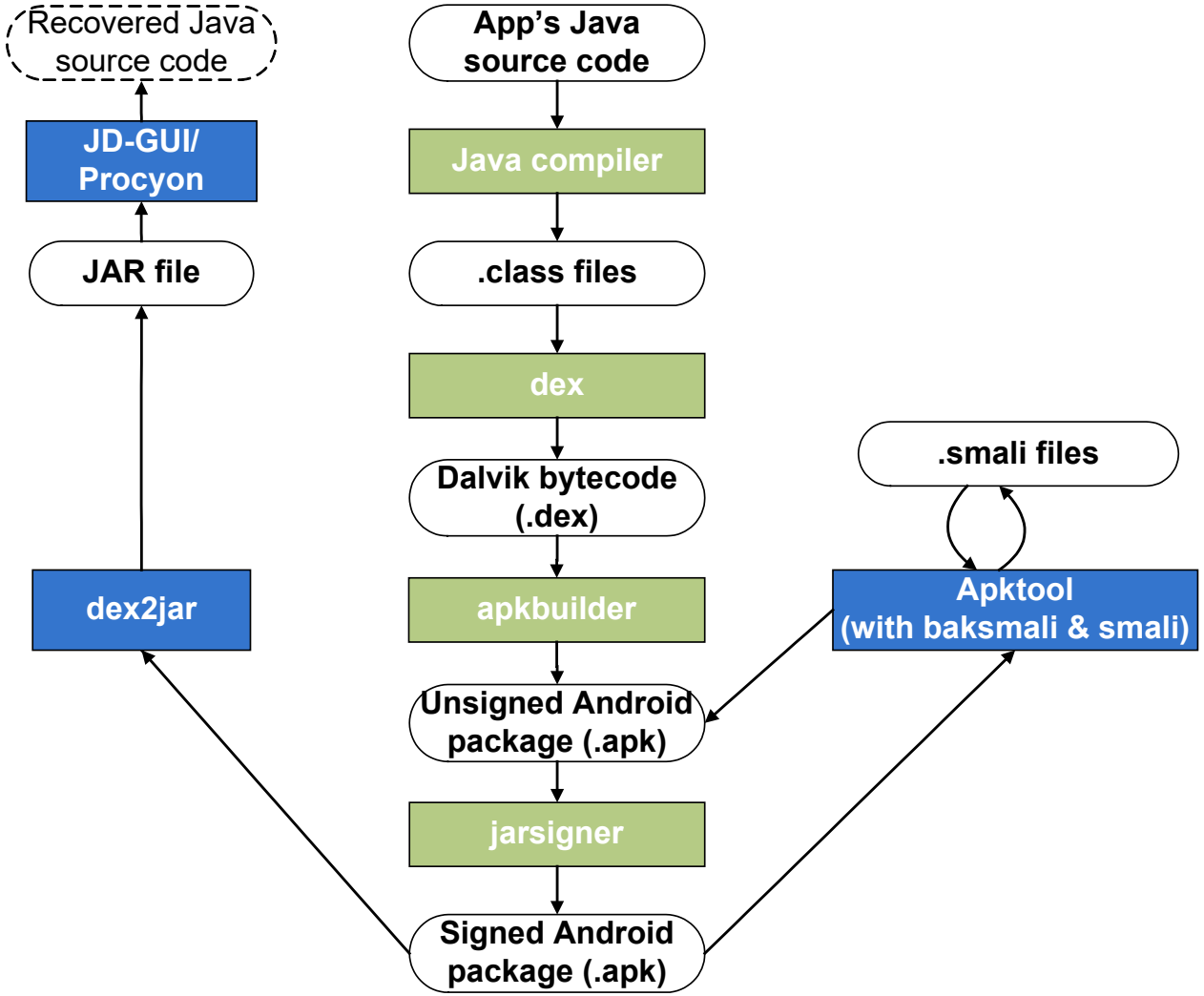
Analyzing SQLite Database Files

- How about ***deleted entries*** in an SQLite database?
- Some **tools** are available to recover deleted entries:
  - **SQLite-Parser:** https://github.com/mdegrazia/SQLite-Deleted-Records-Parser
  - **Undark**: http://pldaniels.com/undark/
  - **Sqlite Recovery**: https://www.sqlrecoverytool.com/recover-sqlite-database.html
- For the **internal mechanisms** of recovery process, see: https://sqliteforensictoolkit.com/recovering-deleted-records-from-an-sqlite-database/

App Analysis: App Reverse Engineering

- **App analysis** as *part* of mobile device forensics
- General **steps**:
  - App component **extraction**: simply unzip the apk file
  - App's `classes.dex` **decompilation**:
    - Into **smali code**: use Apktool (https://ibotpeaches.github.io/Apktool/)
    - Into **Java**: use dex2jar and then Java decompiler (see Lab 9, Task 3)
    - See the differences from the diagram on the next slide
  - App code **analysis**

Android App Reverse Engineering: Options



Other Data Analysis

- Lastly, **analysis of *app-specific data*** is also required

Package name: `com.android.providers.contacts`

Files of interest:

- `/files/`:
  - `photos/`
  - `profile/`
- `/databases/`:
  - `contacts2.db`
  - `calllog.db`

Package name: `com.android.chrome`

Files of interest:

- `/app_chrome/Default/`:
  - `Sync Data/SyncData.sqlite3`
  - `Bookmarks`
  - `Cookies`
  - `Google Profile Picture.png`
  - `History`
  - `Login Data`
  - `Preferences`
  - `Top Sites`
  - `Web Data`
- `/app_ChromeDocumentActivity/`

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

How about Android Device Rooting?

- Example of running **a *root exploit***:

```
> adb push asroot2 /data/local/
> adb shell chmod 0755
/data/local/asroot2
> adb shell
$ /data/local/asroot2 /system/bin/sh
# mount -o remount,rw -t yaffs2
/dev/block/mtdblock3 /system

# cd /system/bin
# cat sh>su
# chmod 4755 su
```

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Android Device Rooting

- Another example:

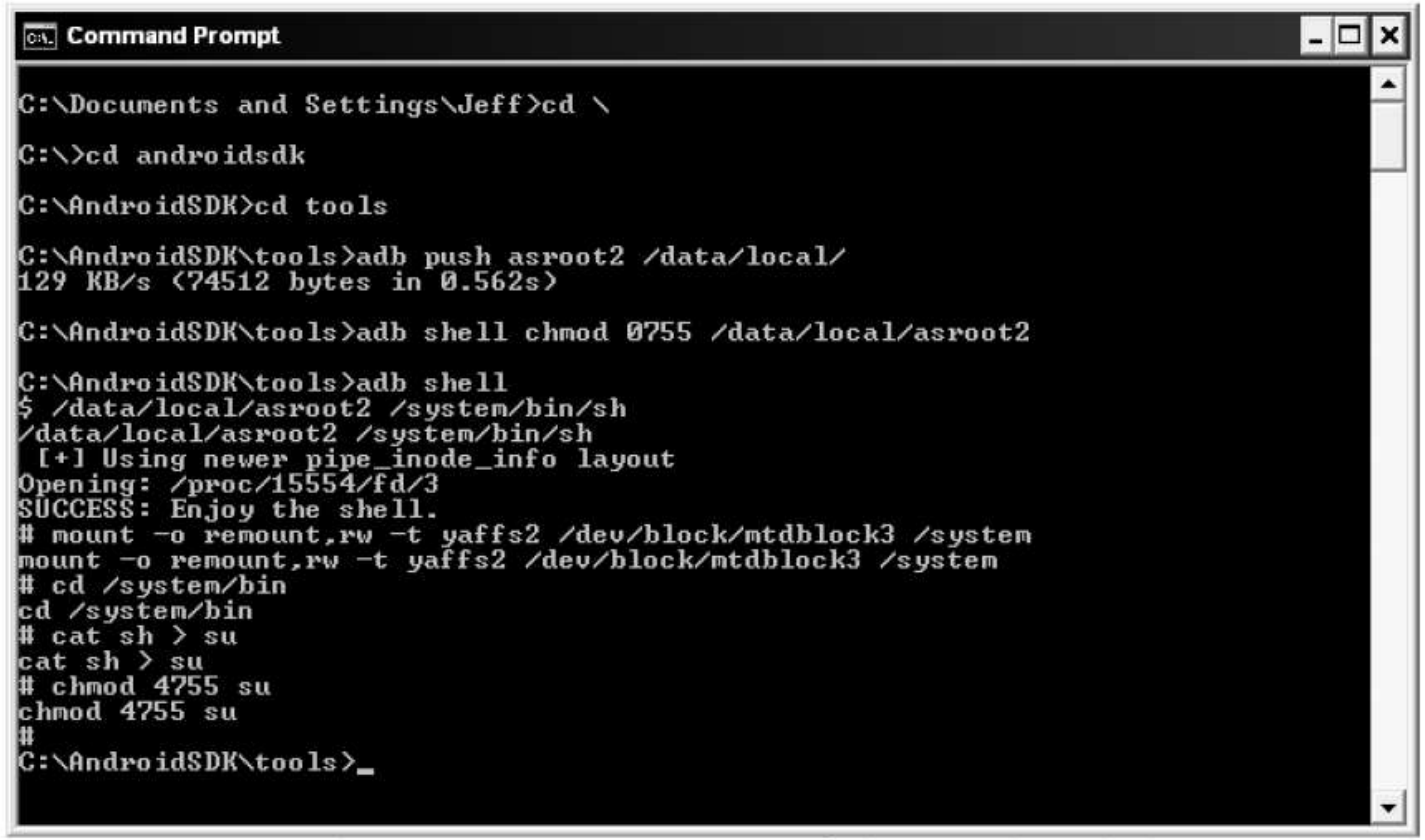


Figure 7. Obtaining root access of the Android device in Windows.

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Android Device Rooting

- A rooting **exploits vulnerabilities** on a device, then **install the su** binary
- Some well-known Android **root exploits**: psneuter, asroot, GingerBreak, …
- Some popular ***automated*** Android **rooting tools/apps**:
  - KingoRoot: https://www.kingoapp.com/
  - Root Genius: https://www.rootgenius.com/
  - iRoot: http://www.iroot.com/
- *Question: is rooting using a root exploit forensically secure/acceptable?*
  - The rooting will **change** the state of the device
  - ***But***, …..

Data Acquisition of *Internal* Memory

- **Non-volatile** memory: use **dd**

```
dd    if=/dev/mtd/mtd0    of=/sdcard/mtd0.dd
bs=1024
```

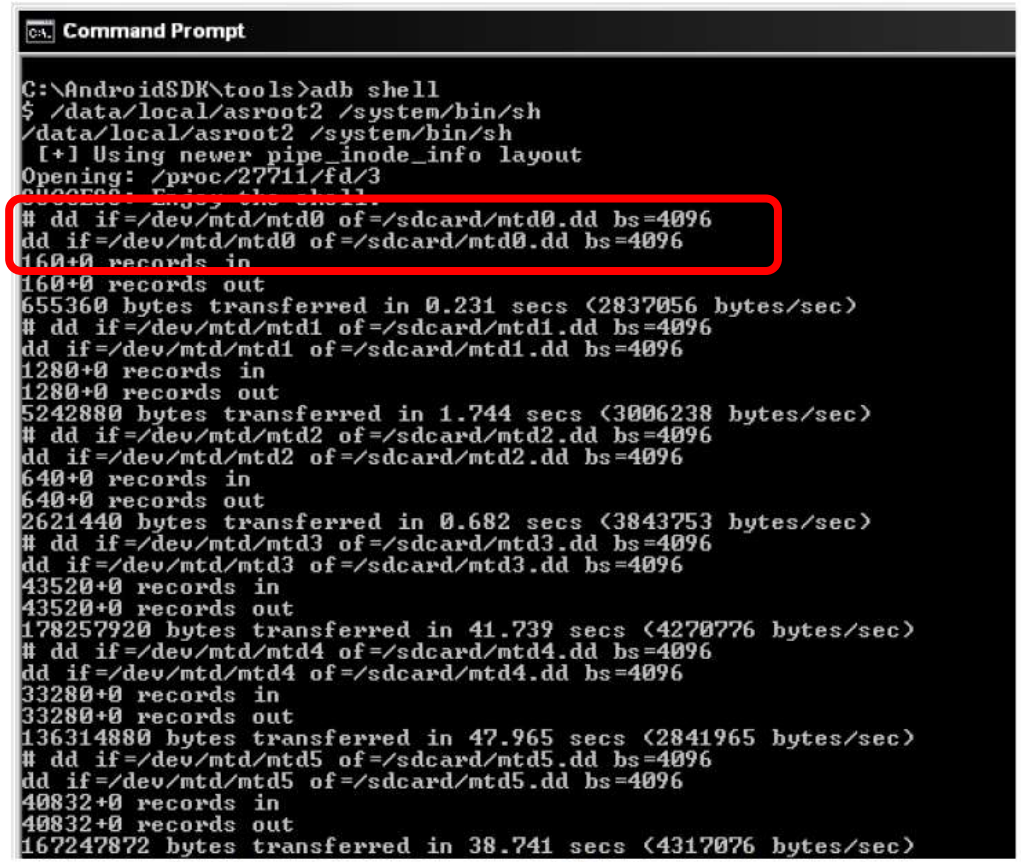
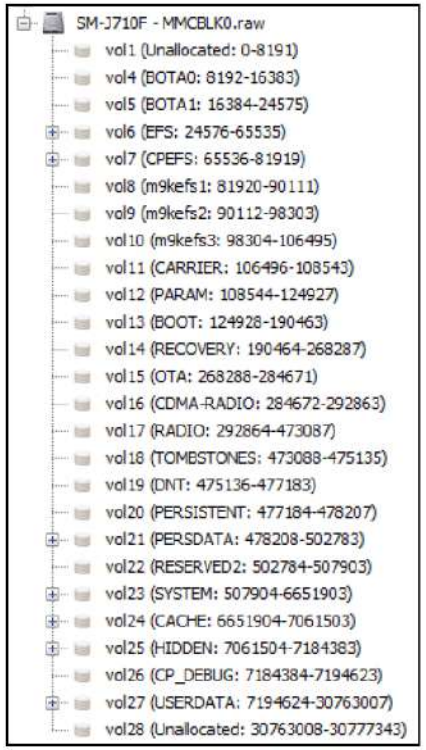


Figure 8. Obtaining root access of the Android device in Windows.

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Data Acquisition of *Internal* Device

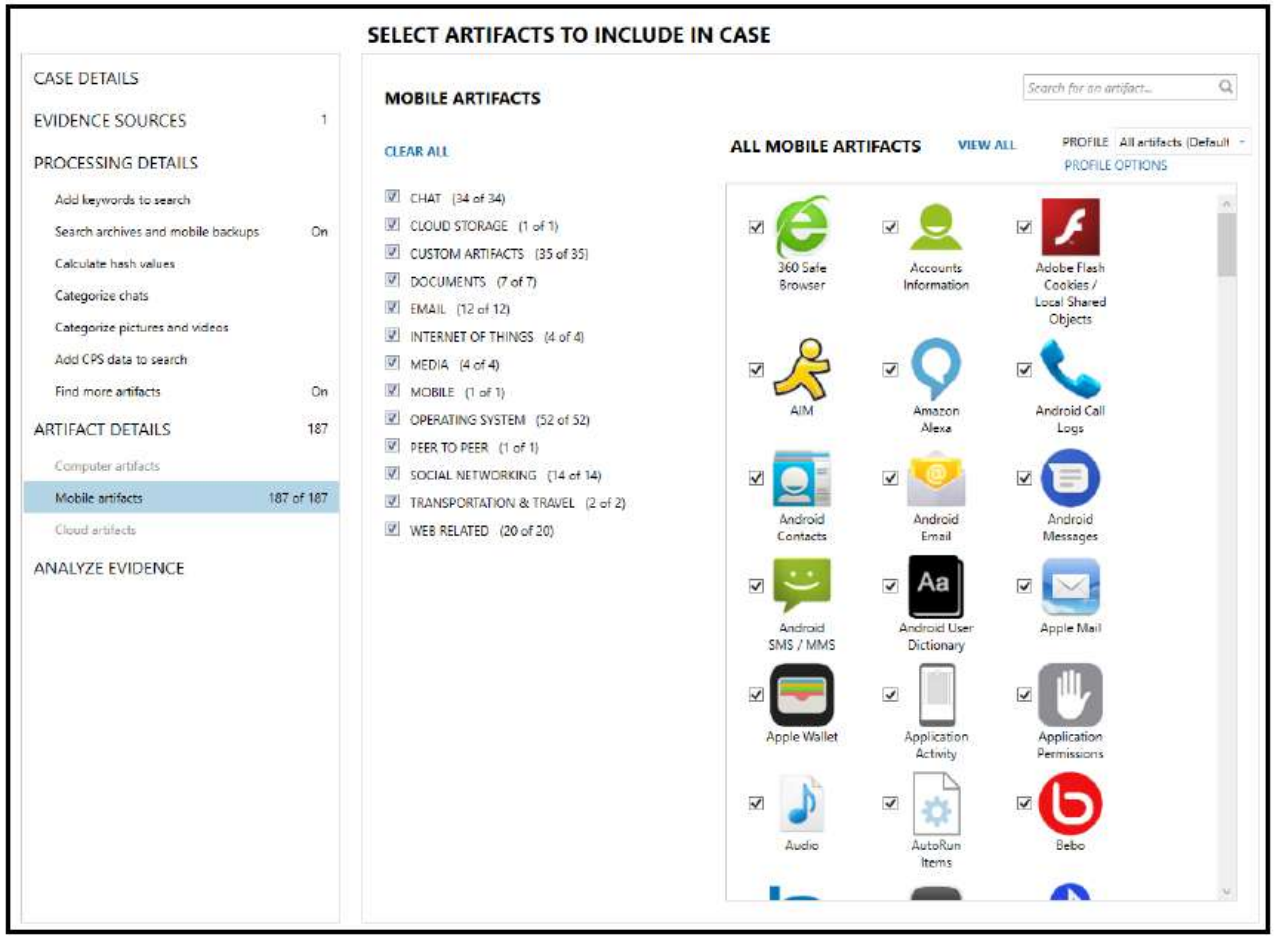
- Sample video on performing **non-volatile memory** acquisition using adb, **dd** and ncat: https://www.youtube.com/watch?v=KKkvkCgMeMA
- For **data analysis**, you can use FTK Imager or Autopsy:



From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Data Acquisition of *Internal* Device

- Alternatively, **mobile-device oriented forensics tools** can be used, e.g. Magnet AXIOM



From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Data Acquisition of *Internal* Device

- ***Volatile*** memory acquisition:
  - Use **LiME**: Android is based on Linux!
  - Please refer to acquisition using LiME & Volatility usage
  - Sample video on acquisition & analysis using LiME & Volatility: https://www.youtube.com/watch?v=enKqmD_8VWw
  - Another alternative RAM acquisition tool is **mem** (https://github.com/MobileForensicsResearch/mem)
  - Commonly used together with **netcat for Android** (https://github.com/MobileForensicsResearch/netcat) in order to write data out over abd and avoid writing to the device
  - Target PID can be set to **0**: all of RAM will be imaged

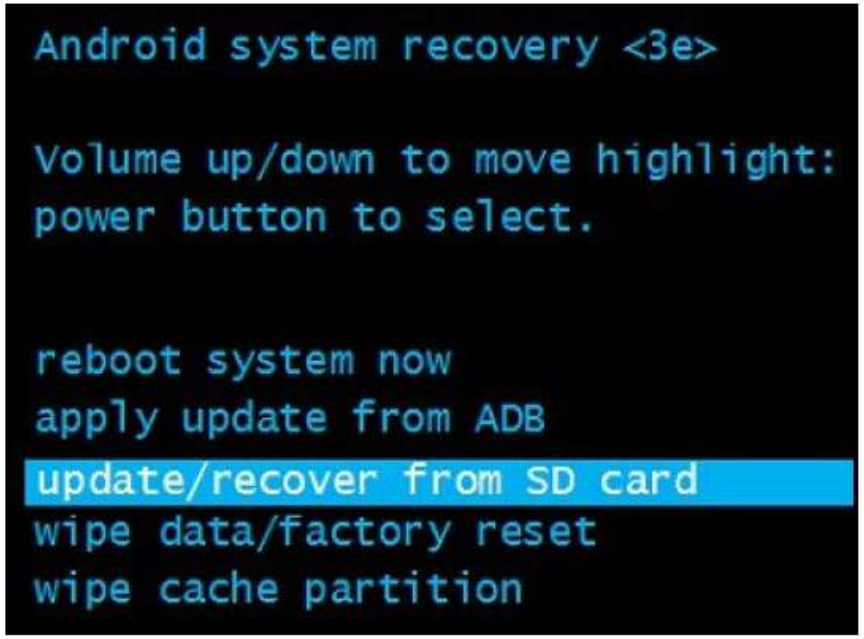
What if the Device *Cannot* be Rooted?

- It is still possible if a device has an ***unlocked bootloader***
- Android devices usually have a ***locked*** bootloader, which however **could be *unlockable***
- Unlocking a locked bootloader will **erase** user data partition!
- Some users have an ***unlocked* bootloader**: to install a ***custom recovery mode*** for unrestricted device access
- If a device has an ***unlocked* bootloader**: a **data acquisition** can be performed
  - Without rooting: root will be given via the recovery image
  - Without USB debugging: device interaction is via fastboot

Background: Recovery Mode

- Three main ***system partitions*** in an Android device:
  - **Boot loader**:
    - It is run when the device is powered on
    - Performs low-level HW initialization and boots into other partitions
  - **Android ROM**:
    - Contains all OS files that are necessary to run the device
  - ***Recovery***:
    - Originally contains the ***stock* recovery**
    - Used to factory-reset device (delete all user data & files), **applies system updates**
    - Can be flashed by user to contain a ***custom recovery***

Background: Example of *Stock* Recovery



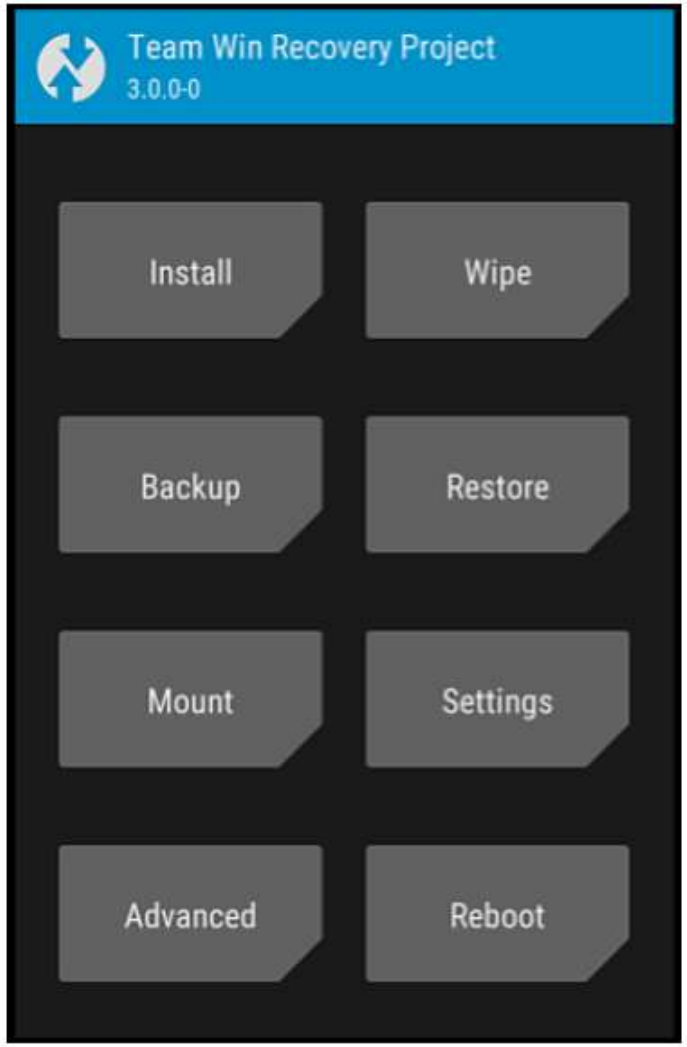
Android stock recovery

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Background: *Custom* Recovery

- What can a ***custom recovery*** do for the **device owner**?
  - Provides **full backup** & **restores** functionality (e.g. **NANDroid**)
  - Allows **unsigned** update packages or allows signed packages with custom keys
  - Selectively **mounts** device partitions and SD card
  - Provides USB mass storage **access** to SD card or data partitions
  - Provides **full ADB access,** with the ADB daemon running as root
  - A fully featured **BusyBox binary**, giving a collection of powerful command line tools in a single binary executable
- For the digital forensic **investigator**? Data acquisition!

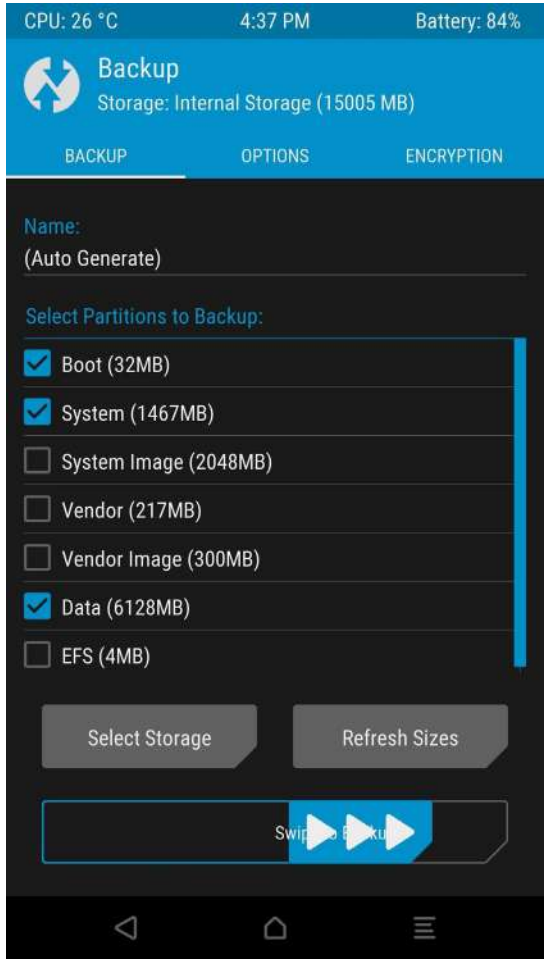
Background: *Custom* Recovery Example (TWRP)

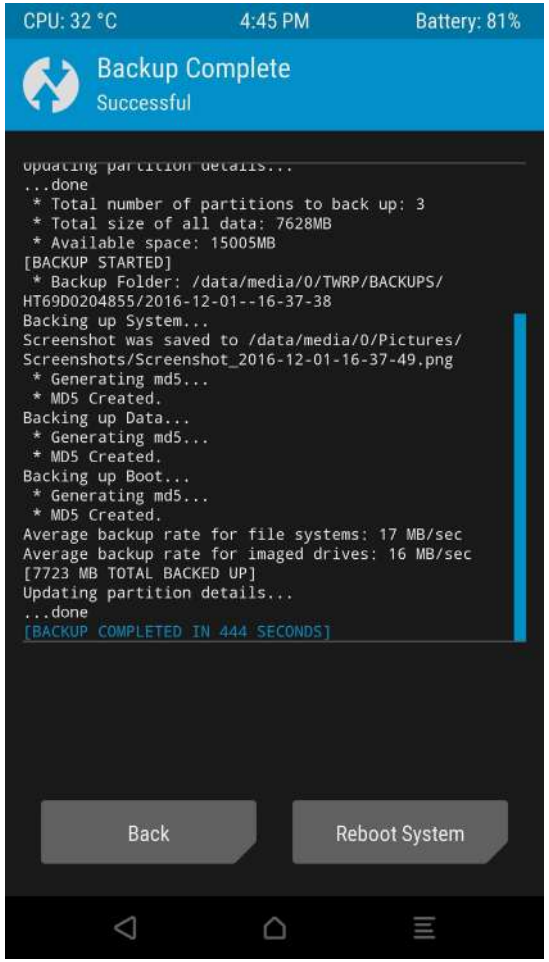


TWRP recovery

From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Background: TWRP & NANDroid Backup



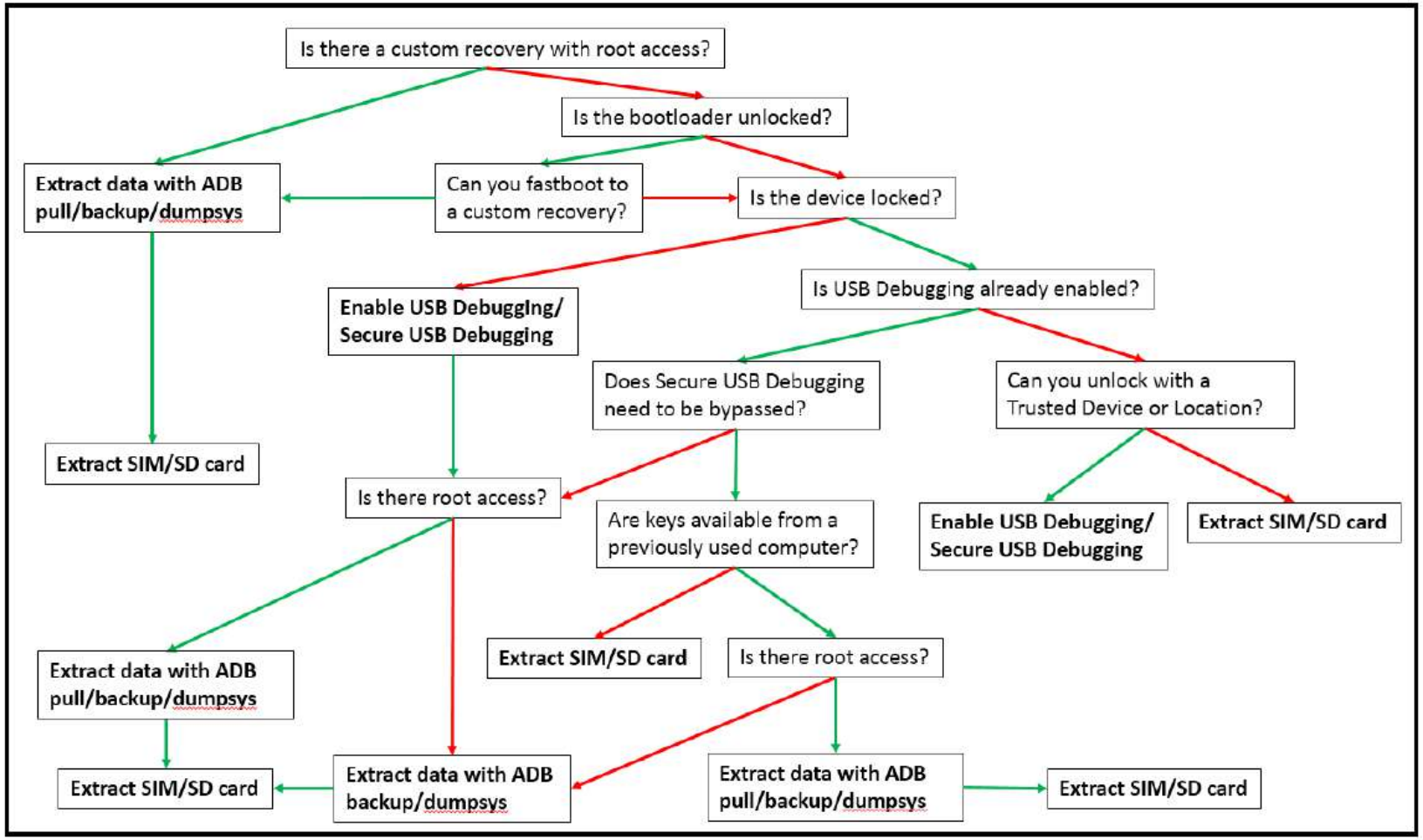


Source: https://android.gadgethacks.com/how-to/twrp-101-make-nandroid-backup-restore-your-entire-phone-0175300/

Bootloader Unlocking & Custom Recovery Installation

- Both are usually done using **Fastboot:**
  - A protocol utility built into the Android SDK
  - Direct interaction with a **device's bootloader** over a **USB** connection
  - A much **lower-level** version of ADB
  - It allows for modification/flashing of filesystem images, **unlocking** the boot loader
- Some Fastboot-related **commands**:
  - Bring a device into Fastboot mode: `adb reboot bootloader`
  - Unlock a locked but unlockable bootloader: `fastboot oem unlock` *Optional*
  - Flash the recovery partition: `fastboot flash recovery twrp.img`
  - Reboot: `fastboot reboot`

Android Logical Data Acquisition: Flowchart



From: Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition

Automated Tool-Based Data Acquisition

- Some **integrated tools** that can work in a **"plug-and-extract" fashion** to extract user data
- Example: **Cellebrite Universal Forensic Extraction Device (UFED)**
  - Has the ability to extract data from nearly **8,200 devices** as of June 2012
  - Retrieves subject data via **logical**, **file system**, **or physical extractions** (i.e.: hex dump, a bit-for-bit copy of a mobile device's entire storage)
  - Sold to government entities or corporate clients only
  - References:
    - https://en.wikipedia.org/wiki/Cellebrite#Mobile_forensics_products
    - https://en.wikipedia.org/wiki/Cellebrite_UFED
    - UFED Touch: https://www.youtube.com/watch?v=od0EjETlqjE
    - UFED 4PC: https://www.youtube.com/watch?v=5fEYqpJ6Mrw

Cellebrite UFED: Several Available Platforms

- **UFED 4PC**: for any user requiring access & collection capabilities on their existing **PC or laptop**
- **UFED Touch2**: fo data collection capabilities **anywhere**, whether in the lab, a remote location, or in the field
- **UFED Ruggedized Panasonic Laptop**: comes in a **ruggedized case** that can withstand drops, shocks & extreme temperatures

Source: https://cellebrite.com/en/ufed/

Cellebrite UFED: Extracted Evidence

Phone Examination Report Properties

Selected Manufacturer:	HTC
Selected Model:	HTC Hero CDMA (Android)
Detected Manufacturer:	sprint
Detected Model:	HERO200
Revision:	1.5 CUPCAKE eng.u70000.20090921.205629
MEID:	270113178313016459 (HEX: A1000007C69D8B)
IMSI:	310006032060645
Extraction start date/time:	11/06/09 16:39:45
Extraction end date/time:	11/06/09 16:51:23
Phone Date/Time:	11/06/09 20:38:51 (GMT)
Connection Type:	USB Cable
UFED Version:	Software: 1.1.2.4 UFED , Full Image: 1.0.2.4 , Tiny Image: 1.0.2.1
UFED S/N:	5518965

Figure 32. Phone identifying information from the UFED.

4	* Twitter	10/11/09 13:40:26 (GMT)	Read	Inbox	Phon
171658	* Shannon Maguire	10/11/09 06:18:47 (GMT)	Read	Inbox	Phon
052307	* Steve Charbonneau	10/11/09 04:19:44 (GMT)	Read	Inbox	Phon
052307	* Steve Charbonneau	10/11/09 03:49:45 (GMT)	Sent	Sent	Phon

Figure 33. Some of the SMS messages extracted by the UFED [Phone numbers truncated for publication].

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Cellebrite UFED: Extracted Evidence

104	Incoming		* Jan Lessard	11/05/
105	Incoming		* Shannon Maguire	11/06/
106	Incoming		* Brittney Langdon	11/06/
107	Incoming		N/A	11/06/
189	Outgoing	84	* Mike Anciello	11/06/
190	Outgoing	67	* Kathleen Wanner	11/06/
191	Outgoing	82	* Sara Manfield	11/06/
192	Outgoing	13	N/A	11/06/
46	Missed		* Andrew Cote	11/05/
47	Missed		* Jodi Lessard	11/05/
48	Missed		* Shannon Maguire	11/06/
49	Missed		N/A	11/06/

Figure 34. Some of the call history information extracted by the UFED;

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Cellebrite UFED: Extracted Evidence

15	File Name: IMAG0028.jpg File Size: 879981 Bytes File Date/Time: 10/20/09 23:48:50 MD5: 124E531F4EBCB1424135F47990F3FFA9 SHA256: D7F16EBA C29C90A 0995C7C 777F625 ED16C61 8515BEB 6DF00E6 03EA9D4 9AD59FA	Resolution: 72x72 (unit: inch) Pixel Resolution: 2560x1712 Camera Make: HTC Camera Model: HERO200 Date/Time: 2009:10:20 23:48:49	
7	File Name: imagejpeg_2.jpg File Size: 68872 Bytes File Date/Time: 10/15/09 23:59:38 MD5: 7B2B667F81DA33D01D2A3DED60486C7F SHA256: 492F43B7 F7553B3 0FA6BCB EA70C58 1FEEF8C B9A502C A86F0AF 11F9A2B E42E416	Resolution: 72x72 (unit: inch) Pixel Resolution: 1280x960 Camera Make: LG Electronics Inc Camera Model: LG-VX8560 Date/Time: 0000:00:00 00:00:00	

Figure 35. Two of the picture files extracted by the UFED.

#	File Name	File Size	File Date/Time	File Link
1	VIDEO0001.3gp MD5: 0569D5FE42A0AC9BB1AE797ED3BBC0F0 SHA256: 271A2C24 8A26480 2B536D8 0F46743 04DBC81 94398B3 B878EBD FA9524B 0E2949D	1250247 Bytes	10/20/09 23:47:13	VIDEO0001.3gp

Figure 36. Video file extracted by the UFED.

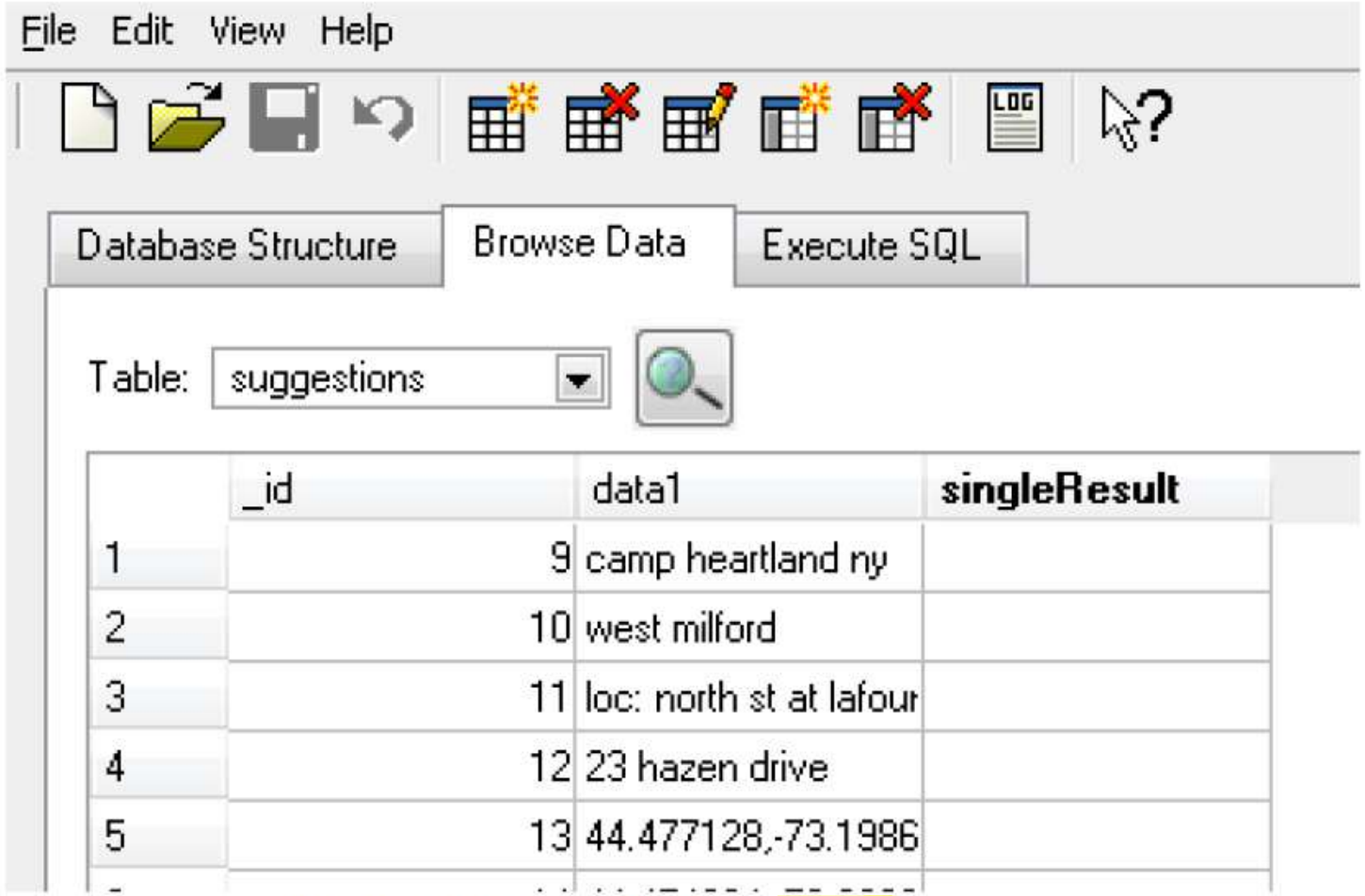
From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Sample Extraction Results

Figure 10. Recovered images: Corrupted image file (left) and intact image file (right).

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Sample Extraction Results



	_id	data1	singleResult
1	9	camp heartland ny	
2	10	west milford	
3	11	loc: north st at lafour	
4	12	23 hazen drive	
5	13	44.477128,-73.1986	

Figure 22. Google maps database.

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

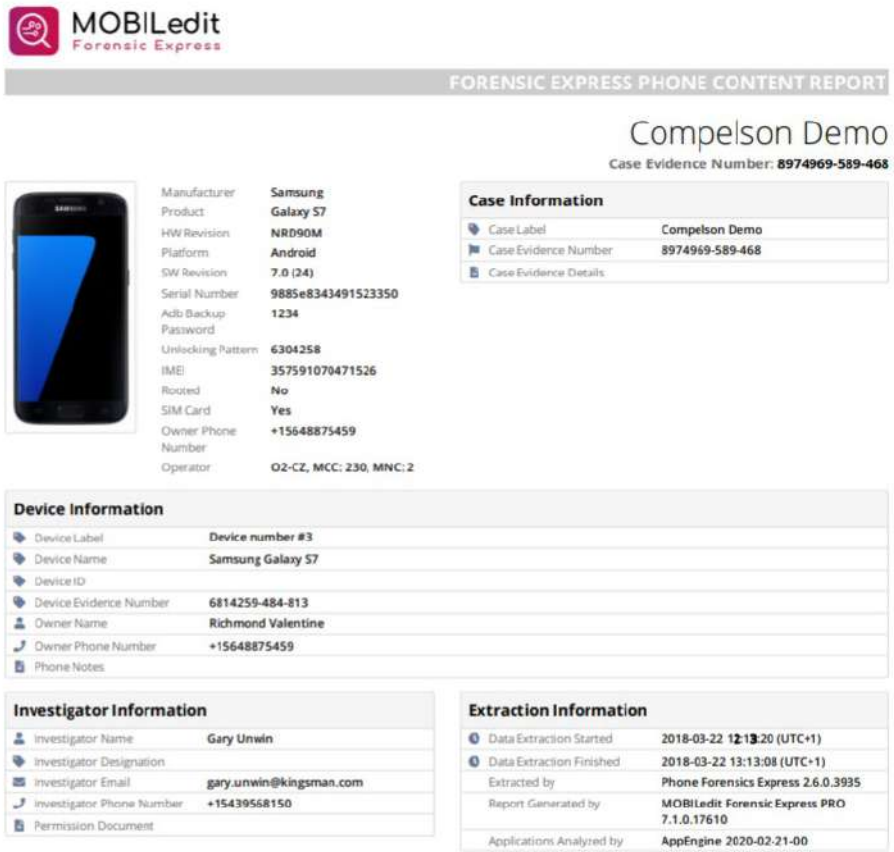
Sample Extraction Results

```
ls
UN-1C.AMR
UN-9.AMR
UN-B.AMR
UN-12.AMR
UN-1A.AMR
UN-1.AMR
UN-13.AMR
UN-7.AMR
UN-5.AMR
UN-11.AMR
UN-17.AMR
UN-F.AMR
UN-10.AMR
UN-6.AMR
UN-19.AMR
UN-16.AMR
UN-A.AMR
UN-D.AMR
UN-8.AMR
UN-C.AMR
# dd if=/data/data/com.coremobility.app.vnotes/files/UN-7.AMR of=/sdcard/UN-7.AM
R
dd if=/data/data/com.coremobility.app.vnotes/files/UN-7.AMR of=/sdcard/UN-7.AMR
73+1 records in
73+1 records out
37606 bytes transferred in 0.015 secs (2507066 bytes/sec)
# dd if=/data/data/com.coremobility.app.vnotes/files/UN-C.AMR of=/sdcard/UN-C.AM
R
dd if=/data/data/com.coremobility.app.vnotes/files/UN-C.AMR of=/sdcard/UN-C.AMR
37+1 records in
37+1 records out
19334 bytes transferred in 0.008 secs (2416750 bytes/sec)
# dd if=/data/data/com.coremobility.app.vnotes/files/UN-9.AMR of=/sdcard/UN-9.AM
R
dd if=/data/data/com.coremobility.app.vnotes/files/UN-9.AMR of=/sdcard/UN-9.AMR
64+1 records in
64+1 records out
32870 bytes transferred in 0.013 secs (2528461 bytes/sec)
#
```

Figure 25. Voice mail audio files.

From: Lessard & Kessler, "Android Forensics: Simplifying Cell Phone Examinations", 2010

Sample Produced Report



MOBILedit
Forensic Express

FORENSIC EXPRESS PHONE CONTENT REPORT

Compelson Demo

Case Evidence Number: 8974969-589-468

Manufacturer	Samsung
Product	Galaxy S7
HW Revision	NRD90M
Platform	Android
SW Revision	7.0 (24)
Serial Number	9885e8343491523350
Adb Backup Password	1234
Unlocking Pattern	6304258
IMEI	357591070471526
Rooted	No
SIM Card	Yes
Owner Phone Number	+15648875459
Operator	O2-CZ, MCC: 230, MNC: 2

Case Information

Case Label	Compelson Demo
Case Evidence Number	8974969-589-468
Case Evidence Details	

Device Information

Device Label	Device number #3
Device Name	Samsung Galaxy S7
Device ID	
Device Evidence Number	6814259-484-813
Owner Name	Richmond Valentine
Owner Phone Number	+15648875459
Phone Notes	

Investigator Information

Investigator Name	Gary Unwin
Investigator Designation	
Investigator Email	gary.unwin@kingsman.com
Investigator Phone Number	+15439568150
Permission Document	

Extraction Information

Data Extraction Started	2018-03-22 12:13:20 (UTC+1)
Data Extraction Finished	2018-03-22 13:13:08 (UTC+1)
Extracted by	Phone Forensics Express 2.6.0.3935
Report Generated by	MOBILedit Forensic Express PRO 7.1.0.17610
Applications Analyzed by	AppEngine 2020-02-21-00

https://www.mobiledit.com/s/MOBILedit-Forensic-Express-Demo-Report_710.pdf

Mobile Device & Android Forensics

- Some **references**:
  - Ayers et al., "*Guidelines on Mobile Device Forensics*", NIST Special Publication 800-101, Rev 1, 2014
  - Andrew Hoog, "*Android Forensics: Investigation, Analysis and Mobile Security for Google Android*", 2011
  - Oleg Skulkin et al., "*Learning Android Forensics: Analyze Android devices with the latest forensic tools and techniques*", 2nd edition, Packt Publishing, December 2018

iOS Forensics

Some Background on Apple, MacOS, iOS

- Cellebrite, "***Apple Computer and MacOS Basics***, ***Technical Guide***", July 2020, available from: https://cellebrite.com/en/apple-computer-and-macos-basics/ (has been uploaded to Canvas):
  - Windows: Registry files
  - **macOS**: No central location, but there are **property list (plist) files** in application bundles, user folders, other locations on the system
- iOS device forensics:
  - https://www.youtube.com/watch?v=5MJ2Iyot5Uw

Some Notes on iOS Forensics

- **Property lists** on iOS:
  - Use the filename extension **`.plist`**
  - Store **serialized objects**
  - Often used to store a **user's settings**, information about bundles & applications, etc.
  - Reading of a plist file: see Lab 9 Task 4
  - Reference: https://developer.apple.com/library/content/documentation/Cocoa/Conceptual/PropertyLists/Introduction/Introduction.html

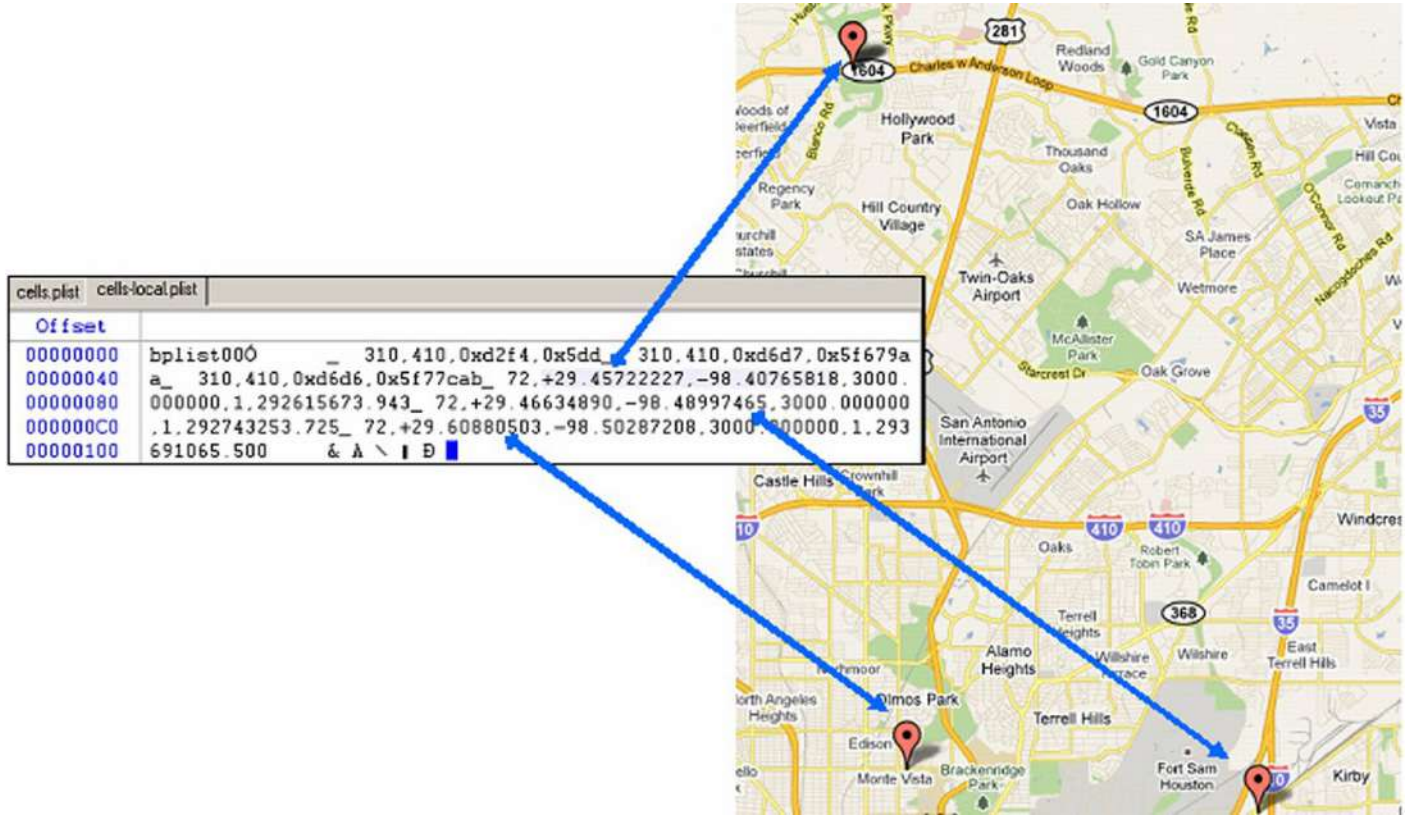


FIGURE 20.6 A file from an iPhone containing longitude and latitude of cellular tower locations used by the device.

Some Notes on iOS Forensics

- Following a logical acquisition, some **data items of interest** are:
  - `private\var\root\Library\Caches\Backup\`**`Manifest.plist`**:
    a list of **all apps** on the device
  - `wireless\Databases\`**`DataUsage.db`**:
    a database containing tables with the app's name (bundle name),
    processes associated to apps, timestamps of usage, and
    data (in/out) via the **WAN**
  - `com.`*<company><app-name>*`\`**`Cache.db`**:
    a database containing **data received** from an outside source,
    e.g., server or internet

Some Notes on iOS Forensics

- Many **concepts/tools** are similar to those in Android:
  - Iphone **jailbreak**: Android rooting
  - **Jailbreak tools**: LiberiOS, Electra, g0blin, Checkra1n, …
  - **Acquisition tools**: Oxygen Forensic Extractor, Cellebrite UFED, Elcomsoft iOS Forensic Toolkit, …
  - **Data viewing tools**: iBackup Viewer, iExplorer, …
  - …

Lab 9 Exercises

- Task 1: To examine **Packages.xml** of an Android device
- Task 2: To examine a **SQLite database file** of an Android app
- Task 3: To examine a **suspicious .apk** from an Android device
- Task 4: To examine a **.plist file** from an iOS device

Don’t forget: the **Graded Lab Tasks #6** (your last lab submission!)

Questions?
See you next week!
(*with Case-1 presentation*)