Mobile Threats Incident Handling (Part II)

Handbook, Document for teachers

1.0

SEPTEMBER 2015

About ENISA

Authors

This document was created by Yonas Leguesse, Christos Sidiropoulos, and Lauri Palkmets in consultation with S-CURE[1] (The Netherlands), ComCERT[2] (Poland), and DFN-CERT Services[3] (Germany).

Contact

For contacting the authors please use cert-relations@enisa.europa.eu.
For media enquires about this paper, please use press@enisa.europa.eu.

[1] Don Stikvoort, Michael Potter, and Alan Robinson
[2] Tomasz Chlebowski, Mirosław Maj, Piotr Szeptyński, and Michał Tatar
[3] Mirko Wollenberg

Table of Contents

Main Objective	This course will introduce concepts, tools, and techniques used for Mobile and Network Forensics. The students will familiarise themselves with the risks found on Mobile platforms and also ways of identifying and mitigating such risks, as well as techniques to analyse mobile related threats and malware.	
Targeted Audience	CSIRT staff involved in the process of incident handling, especially those responsible for detection of new threats related directly to the CERT customers.	
Total Duration	24 hours	
Theory Part Time Schedule	Introduction to mobile forensics	2 hours
	Threats and incidents handling	3 hours
	Mobile Forensics	3 hours
	Mobile forensic procedures	3 hours
	Mobile network forensic	0.5 hour
	Mobile malware reverse engineering	0.5 hour
	Recap of mobile forensic tools	1hour
	Countermeasures and protective measures	2 hours
Exercises Time Schedule	Task 2.1: Analysis of sample application's permissions on an Android device	1 hour
	Task 2.2: Analysis of sample application's Mach-o header on an iOS device	0.5 hour
	Task 3.1: A quick evaluation of knowledge regarding mobile	0.5 hour
	Task 4.1: Logical data extraction from Android devices	4 hours
	Task 4.2: File system extraction from Android devices	
	Task 4.3: Manual file carving	
	Task 4.4: RAM memory dump from Android device	
	Task 4.5: iOS – IPhone Backup Analyser 2	1 hour
	Task 4.6: Brute-forcing Android encryption mechanisms	1 hour
	Task 5.1: Analysing pcap data and proxy logs of Android.Trojan.SLocker.DZ	0.5 hour
	Task 5.2: Analysing pcap data and proxy logs of iOS.Oneclickfraud	0.5 hour
	Task 6.1: Analysing Android.Trojan.SLocker.DZ	0.5 hour
	Task 6.2: Analysing iOS.Oneclickfraud	0.5 hour
Frequency	Once per team member	

1. Introduction to mobile forensics

1.1 Mobile technologies

Mobile devices are the most widely used communication tools today. But they are not just limited to communications. Mobile technologies have created something like a "mobile space" which occupies (and sometimes fulfils) the life of many citizens. As of August 2015, according to GSMA Intelligence, there are more than 3.7 billion unique mobile subscribers with more than 7.2 billion mobile devices on Earth.[4] Everyone from children to senior citizens use mobile devices in different ways, depending on their preferences, hobbies or business purposes.

The primary role of a mobile phone is to provide an easy and fast means of communication thus enabling users to stay connected.

Secondly, there is an increasing want to stay connected to the Internet at all time. Thanks to modern smartphones this is possible. Mobile phones let users enjoy for example social media on the go or find the shortest route to a cafe or a bank.

One cannot forget to mention business: since businesses require constant communication, nowadays it is difficult to imagine any business functioning without the use of mobile phones. And with all the new devices available in the market these days, business people can organise their schedule and set reminders, so that they do not forget any important appointments. Mobile phones come packed with a lot of business applications (apps), which make the lives of workers a lot easier.

That being said, one must also consider the dangers associated with the ever growing use of mobile devices. Data security is a critical concern. Users must protect their private or business-related information stored on mobile devices by using suitable protection mechanisms. However, the most important is the people behaviour, since no technical security controls can replace common sense.

1.2 Historical evolution of mobile operating systems

The history of mobile operating systems is a short one but with many highlights over the last 45 years:[5] [6] [7] [8] [9]

- 1970–1979 First mobile phones using cellular technology.
- 1979–1992 Mobile phones use embedded systems to control operation.
- 1993 The first smartphone, IBM's Simon, has a touchscreen, e-mail and PDA features.
- 1996 Palm Pilot 1000 personal digital assistant (PDA) is introduced with the Palm OS mobile operating system.
- 1996 First Windows CE Handheld PC devices are introduced.
- 1999 Nokia S40 OS is officially introduced along with the Nokia 7110.
- 2000 Symbian becomes the first modern mobile OS on a smartphone with the launch of Ericsson R380.
- 2001 The Kyocera 6035 is the first smartphone with Palm OS.
- 2002 Microsoft's first Windows CE (Pocket PC) smartphones are introduced.

[4] GSMA Intelligence, https://gsmaintelligence.com/, last accessed on: 2015-09-14
[5] Mobile operating system, https://en.wikipedia.org/wiki/Mobile_operating_system, last accessed on: 2015-09-14
[6] iOS version history, https://en.wikipedia.org/wiki/IOS_version_history, last accessed on: 2015-09-14
[7] Android version history, https://en.wikipedia.org/wiki/Android_version_history, last accessed on: 2015-09-14
[8] Window Phone version history, https://en.wikipedia.org/wiki/Windows_Phone_version_history, last accessed on: 2015-09-14
[9] BlackBerry OS release history, https://en.wikipedia.org/wiki/BlackBerry_OS#Release_history, last accessed on: 2015-09-14

- 2002 BlackBerry releases its first smartphone.
- 2005 Nokia introduces Maemo OS on the first Internet tablet N770.
- 2007 Apple iPhone is introduced as a "mobile phone" and "internet communicator".
- 2007 Open Handset Alliance (OHA) formed by Google, HTC, Sony, Dell, Intel, Motorola, Samsung, LG, etc.
- 2008 OHA releases Android 1.0 with the HTC Dream (T-Mobile G1) as the first Android phone.
- 2009 Palm introduces webOS with the Palm Pre. By 2012 webOS devices were no longer sold.
- 2009 Samsung announces the Bada OS with the introduction of the Samsung S8500.
- 2010 Windows Phone OS phones are released but are not compatible with the previous Windows Mobile OS.
- 2011 MeeGo the first mobile Linux, combining Maemo and Moblin, is introduced with the Nokia N9, a collaboration of Nokia, Intel and Linux Foundation
- In September 2011 Samsung, Intel and the Linux Foundation announced that their efforts will shift from Bada, MeeGo to Tizen during 2011 and 2012.
- In October 2011 the Mer project was announced, centred around an ultra-portable Linux + HTML5/QML/JavaScript core for building products with, derived from the MeeGo codebase.
- 2012 Mozilla announced in July 2012 that the project previously known as "Boot to Gecko" was now Firefox OS and had several handset OEMs on board.
- 2013 Canonical announced Ubuntu Touch, a version of the Linux distribution expressly designed for smartphones. The OS is built on the Android Linux kernel, using Android drivers, but does not use any of the Java-like code of Android.
- 2013 BlackBerry released their new operating system for smartphones and tablets, BlackBerry 10.
- 2014 Microsoft released Windows Phone 8.1 in February 2014.
- 2014 Apple released iOS 8 in September 2014.
- 2014 BlackBerry released BlackBerry 10.3 with integration with the Amazon Appstore in September 2014.
- 2014 Google released Android 5.0 "Lollipop" in November 2014.
- 2015 Microsoft released Windows 10 Mobile developer preview in February 2015.
- 2015 Google released Android 5.1 which is an updated version of "Lollipop" in March 2015.
- 2015 Google released Android 6 developer preview in May 2015.
- 2015 Apple released iOS 9 Beta for public in June 2015.

1.3 Mobile forensics

Mobile forensics is a field of knowledge and practice aiming at the application of forensically sound techniques to provide high-quality, factual, relevant and authentic evidence for a legal case. Forensically sound techniques refer to techniques which ensure the proper code of conduct during the identification, seizure, storage, examination and analysis of data, resulting in evidence that can be successfully used in legal cases.

Due to the fact that mobile devices and technologies in general are becoming more and more diverse and complex, mobile forensics is a continuously developing field of study with constantly improved methods, tools, and devices.

Mobile forensic investigations (and digital forensic investigations in general) can be split into several phases:

- Identification of a target mobile device;
- Seizure and data acquisition;
- Examination and analysis;
- Reasoning and reporting;

Over the course of the investigation, all activities must be documented, and the gathered evidence must be stored properly and securely. This exercise will mainly focus on data acquisition (excluding physical approach), and on the examination and analysis of data (in terms of mobile device content, application-specific data, malware and network communications). However, all the phases will be explained in later chapters together with corresponding tasks and exercises. To better understand the next sections, basic concepts of data acquisition must be explained here. There are three different approaches for data acquisition:

- Logical acquisition

With a logical acquisition technique, an examiner makes a copy of files and folders located in the logical storage i.e. a partition, or the files/folders themselves. The advantage of this approach over those described below, is its simplicity: logical objects are easier to understand and the whole method is usually less time-consuming. On the other hand, the examiner will not be able to recover deleted items, in most cases he / she will not be aware that they even existed.

- File system acquisition

With this approach, an examiner makes a copy of the file system's structure, including the existing (logical) objects lying on top of it. This technique allows access to objects (e.g. SQLite database records) that were marked as deleted and are not available through logical extraction. The resulting image of the storage media is large but contains all information from a mobile device. The technique is more time-consuming and usually a bit more complex as it requires better understanding of operating system's architecture and file system's structure.

- Physical acquisition

With this technique, an examiner makes a bit-by-bit copy of the storage media (on which lies the file system as well as logical objects). This approach allows for the identification and analysis of data remnants on the actual storage media (e.g. SD card), but it requires direct access to flash memory inside of the device, hence it is the most complex way of data acquisition. In fact, this approach is very similar to hard drive imaging in the traditional computer forensic examination of PCs. Device manufacturers often secure their hardware from direct access, thus vendors of mobile forensic tools employ various techniques to by-pass the restrictions e.g. by uploading their own operating system's kernel to the device (a process briefly explained in later sections).

1.4 Historical evolution of mobile forensics

Mobile forensic technologies developed as the mobile devices and apps market grew. This field of study dates back to the late 1990s and is directly related to the popularity, availability and improved capabilities of mobile communications technologies. Over the last 20 years those technologies have undergone a massive evolution: from simple but rather huge cellular phones providing basic functionalities, to smart devices enabling their users not only to browse the Internet, but also run any application they want (not necessarily need).

While early mobile phones were not capable of doing much more than calling or sending SMS messages, mobile forensic technologies were not in high demand. Usually simple forensic tasks, such as browsing a device's contents or reviewing or photocopying its screen, were enough to gather necessary evidence. Another method of data extraction was to backup the content to a computer for further analysis. The analyses were often supported with information (e.g. call records, billing data) from mobile operators. Such an approach, however, had to change with advance of more complex and powerful mobile devices. It must be also noted, that the techniques mentioned above, were not forensically sound, as not all of them allowed read-only access to a device's content.

New mobile technologies brought new ways of data processing and exchange between users and services through wireless technologies (i.e. GSM/UMTS/LTE, WLAN, Bluetooth etc.) and the Internet. But this also brought a whole new arsenal of mobile forensic techniques allowing for forensically sound data extraction and powerful analysis. Many commercial and open-source tools were developed to address the demand for such solutions. A lot of them are mentioned in later chapters of this handbook and some of them are used during the exercises.

New tools are capable of forensically sound data extraction (logical and physical), automated analysis of data, keyword searches, analysing application-specific artifacts etc. But since new mobile devices are smarter than they were in 1990s, they are also harder to analyse. The complexity of mobile forensic investigation often can be much greater than that of a usual hard disk.

1.5 Latest trends in mobile forensics techniques

While analysing the latest trends in the field of mobile forensics it should be noted that examiners are challenged with a major problem, a lot of data which occupies a lot of memory. Analysis of large amounts of data is usually very time-consuming and in many cases time pressure makes it impossible. This is true for both mobile and "traditional" computer forensic investigations.

The solution to this problem comes with a technique called triage. Triage, in mobile forensics, is a simple but powerful approach. It allows to save a lot of memory as well as a lot of time. What is triage about? Let us imagine a situation in which a device with 64GB of memory is subject to investigation carried out in the field, the device's battery is running low, and there is heavy time pressure.

That is when and where the triage technique comes in handy. Let us assume that the initial analysis shows that only SMS messages are to be extracted due to their relevance to the case. In other words, all other data is unnecessary.

Secondly, it is necessary to prepare an "acquisition profile" which contains the instructions for the forensic software to only read SMS messages from the device, leaving all other data untouched. The resulting data will be only a few megabytes and the extraction will only take a few moments.

Yet another challenge for forensic examiners are automated device lock-out mechanisms, triggered after a given time period. When a device is locked, usually there is not much that can be done to access it (let alone extract the data stored within the device), and it has to be noted that quite often this is the case in investigations. There are some basic options to handle this situation:

- Try and guess the unlock code or pattern. Too many incorrect guesses may result in permanent lockout or wiping and making many guessing attempts will be very time-consuming,
- By-pass lock-out mechanisms by attaching a non-standard USB cable to the mobile device.

Attaching a service wire will put the device into a service state. There is no need to activate USB debugging mode. With a USB service wire plugged into the device, examiners can access physical memory and, unless it is encrypted, extract data without providing PINs, passwords or patterns protecting a locked-out device from access.

Mobile forensic investigators may be surprised when they find out that some mobile devices are equipped with so-called "Chinese chipsets".[10] Because of their low prices, such devices are flooding the market.

10 Chinese Chipsets - Physical Extraction from Mobile Devices with Chinese Chipsets, http://www.cellebrite.com/Pages/chinese-chipsets-mobile-forensics-for-chinese-chipsets, last accessed on: 2015-10-19

At first glance, their data interfaces look like mini or micro-USB but in fact they are something completely different and non-compliant. Vendors of commercial forensic software and hardware (like Micro Systemation or Cellebrite) design their own systems (e.g. Pin Point[11] and Chinex[12]) to help examiners perform memory extraction from these devices by using special kits with dedicated wires.

Since the mobile market is growing very fast, and mobile applications are installed (while not necessarily used) on almost every mobile phone, smartphone or tablet, forensic examiners will meet many new challenges, including software-based SIM cards, biometrics or NFC / HCE technologies.

1.6 Mobile Platforms and Versions

1.6.1 iOS 9[13]

The new version of Apple's mobile system must not only keep pace with the El Capitan version of Mac OS X, but also to keep pace with competitors, particularly Google's Android OS. These mobile operating systems emphasize contextual information in search results, device navigation and the capabilities of voice-activated virtual assistants like Siri.

The system will suggest applications depending on the time of day, and the search engine will understand questions or requests like "show me a picture of Michael's holiday in Croatia".

Applications can communicate with each other. While viewing an e-mail invitation, the user can tap an onscreen button to pass the appointment information to another application like the calendar app.

For many people, the Apple tablet is a tool that can be used not only for fun but also for work. If a user uses two fingers instead of one on the screen, the keyboard turns into a trackpad, which helps enhance usability. The most important new feature in iOS 9 on iPads is multitasking. The revised menu of open applications now allows two apps to open next to each other, simultaneously active, so user can use both at the same time. A new energy saving mode also extends battery life.

iOS 9 introduces a two-step verification and a series of new safeguards for iCloud, to enhance the safety of users and their data.

1.6.2 Android Marshmallow[14]

In late August 2015, Google announced that the newest version of Android will be given the name Marshmallow. The new operating system will present a lot of changes and new features.

Developers have focused primarily on better management of privileges. So far, during installation of an application, a user had to grant all the requested permissions, or not use the app at all. In Android 6.0, users will be able to grant permissions individually and on a regular basis. This will increase the safety of Android users. They will have more control over what data installed applications may access.

Android Marshmallow will have a new system to optimise battery life. When a user enables this function, all installed sensors detect if the user is actively using the smartphone, or if it is lying on the desk unused. When the smartphone does not detect activity, the system will reduce refresh rates for applications.

[11] XRY PinPoint: The Solution for Non-Standard Mobile Device Support, https://www.msab.com/products/pinpoint/, last accessed on: 2015-09-14
[12] UFED CHINEX is an end-to-end solution for the physical extraction and decoding of evidentiary data and passwords from phones manufactured with Chinese chipsets — MTK, Spreadtrum and Infineon, http://www.cellebrite.com/Pages/ufed-chinex, last accessed on: 2015-09-14
[13] iOS 9, http://www.apple.com/ios/whats-new/, last accessed on: 2015-10-19
[14] Android 6.0 Marshmallow, https://www.android.com/versions/marshmallow-6-0/, last accessed on: 2015-10-19

One of the novelties in the system is integration with Android Pay. This is Google's new payment system, which competes with solutions such as Apple Pay or Samsung Pay. It will be available on smartphones with NFC and will be used to perform quick payments. Authorisation will use a fingerprint reader, natively supported by the newest Android version.

1.6.3 Windows 10 Mobile[15]

Windows 10 Mobile is an edition of the Windows 10 operating system developed by Microsoft. It is the successor of Windows Phone 8.1 although it changed the name from Phone to Mobile.

Battery saving is one of the new important features. It allows for a quick and easy way to view which applications consume the most energy. The power saving function will be better in the new Microsoft mobile system. Users will receive more accurate statistics on Windows 10 Mobile and be able to disable some of the applications running in the background.

Microsoft Edge is a completely new web browser that Microsoft designed for Windows 10. Edge will be integrated with the voice assistant Cortana, which appeared in Windows Phone 8.1. This browser will be available only for Windows 10 systems.

Microsoft introduced a software tool for easy and, most importantly, rapid porting of applications written for Android and iOS to Windows 10 Mobile. It means that Windows 10 Mobile should be able to run applications designed for iOS and Android.

There are indications that the successor to Windows Phone 8.1 will allow users to record and store voice chat. It is very possible that this type of material will be linked to specific contacts, allowing to quickly find the selected conversation.

Windows 10 Mobile also includes an adaptation of the "Continuum" concept from its PC counterpart on supported devices, when attached to an external display, Windows 10 Mobile can scale its user interface and applications into a "PC-like" desktop interface with support for mouse and keyboard input.

1.7 Case studies on mobile threats for Android and iOS

Mobile threats change dynamically with vulnerabilities discovered every week. Two particular vulnerabilities deserve special attention and are explained later in the section. Mobile threats can be categorised in a number of ways:

- type of vulnerability,
- way of infection,
- depth and scope of infection,
- impact.

Threats can exploit vulnerabilities in the user-land (i.e. mobile applications) or the operating system components (i.e. kernel, system services). The latter are more dangerous but harder to find, the former are easier to identify but still can be very dangerous as in many cases mobile applications are granted extensive privileges over the operating system and device's components.

Infection can happen in a number of ways: through malware installed as a mobile application, via specially crafted SMS / MMS message, by exploiting vulnerability through a legitimate application (e.g. with crafted

[15] Windows 10 Mobile, http://www.microsoft.com/en-us/mobile/windows10/, last accessed on: 2015-10-19

input data). The most dangerous infections happen remotely without users' knowledge or interaction (e.g. Stagefright[16]).

Once infected, the device can be compromised up to a certain point. The worst infections enable attackers to take control over the whole device (they can also allow users to jailbreak or root their devices — those processes are explained in chapters 1.8 and 1.9). Exploitation of a mobile application's vulnerability can also be limited in scope e.g. allowing unauthorised access only to data processed within the application. However taking control over the behaviour of a mobile application may allow attackers to seek and exploit further vulnerabilities or just to extract interesting information.

Below we mention recent vulnerabilities and exploits affecting Android and iOS devices.

1.7.1 Android and Stagefright

Cybercriminals often target Android users. This is due to several factors: it is not only the most popular mobile system, but also the most fragmented one.

Fragmentation means that at the same time many different versions of Android systems are in use on the market. Even when a bug is detected and swiftly patched by Google in the next version of the system, most users will never see the patches or newer software versions.

As reported by bgr.com: early in 2015 a security researcher named Joshua Drake of Zimperium discovered a loophole that allows the device to be attacked by receiving a malicious multimedia message. This can lead to unauthorised remote code execution and privilege escalation. It is believed to have the biggest impact on Android users so far and leaves 95% of them vulnerable to the attack.[17]

This bug, known as Stagefright, is known to exist in version 2.2 (Froyo) and newer versions of Android. Its successful exploitation can be done remotely (only the target's mobile number is necessary), requires no end-user interaction and is possible due to a flaw in a core Android's component — a C++ library called "Stagefright". The software, developed as a part of Android Open Source Project, provides a back-end engine for opening and playing multimedia content.

1.7.2 CoreText vulnerability

The CoreText vulnerability in Apple iOS 8.0 through iOS 8.3 allows remote attackers to cause a denial of service to the device.[18]

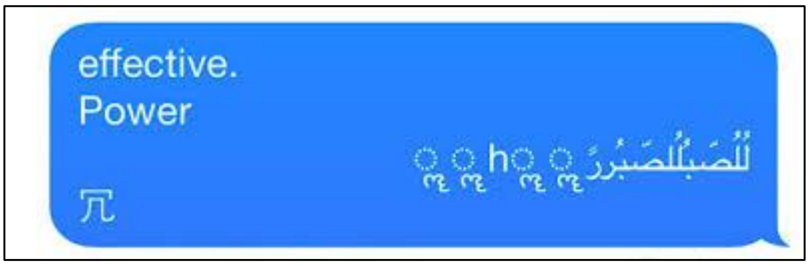

effective.
Power
لُلُصّبُلُلصّبُررً ॣ ॣh ॣ ॣ
冗

Figure 1: Text message is causing denial of service on vulnerable iOS devices

[16] 950 million Android users at risk as researcher uncovers massive security flaw, http://bgr.com/2015/07/27/android-security-flaw-mms-hack/, last accessed on: 2015-09-14
[17] How is Google fixing the Stagefright vulnerability that affects 95% of all Android phones?, http://www.androidauthority.com/how-is-google-fixing-the-stagefright-vulnerability-that-affects-95- of-all-android-phones-631560/, last accessed on: 2015-09-14
[18] CoreText in Apple iOS 8.x through 8.3 allows remote attackers to cause a denial of service, http://www.cvedetails.com/cve/CVE-2015-1157/, last accessed on: 2015-09-14

Many Apple's smartphone users from around the world reported that they received a message containing the string "effective. Power لُلُصّبُلُلصّبُررً ॣ h ॣ 冗" which completely blocked the Messaging application, and in some cases resulted in the restart of the device.

The error is related to the way that iOS presents incoming messages in the preview window. It does not show the whole body of the message, but only a portion thereof, which is located in a separate area. If it is longer then it is shortened by ellipsis. The problem occurs when the dots falls between the selected characters from non-Latin alphabets e.g. Arabic, Marathi or Chinese. In this case, the phone switches off and restarts.

It is worth mentioning a piece of malware, KeyRaider,[19] which was publically announced in August 2015, and which affects jailbroken devices only. According to Palo Alto Networks' researchers it steals "push notification service certificates and private keys, steals and shares App Store purchasing information, and disables local and remote unlocking functionalities on iPhones and iPads". The malware spreads through unofficial applications' repository — Weiphone's Cydia. Researchers found almost 225 000 valid Apple users' credentials stored on a Command & Control server. Details regarding the way that devices are infected and "robbed" of sensitive data are available on Palo Alto Networks website.[20]

1.8 Mobile technologies statistics

To imagine the scale of mobile devices usage, the following tables and charts may be helpful

OPERATING SYSTEM	1Q15 UNITS	1Q15 MARKET SHARE (%)	1Q14 UNITS	1Q14 MARKET SHARE (%)
Android	265,012	78.9	227,549	80.8
iOS	60,177	17.9	43,062	15.3
Windows	8,271	2.5	7,580	2.7
Blackberry	1,325	0.4	1,714	0.6
Other OS	1,268.7	0.4	1,731.0	0.6
Total	**336,054.4**	**100.0**	**281,636.9**	**100.0**

Table 1: Worldwide smartphone sales to end users by operating system in 1Q15 totalled 460.3 million units during the first quarter of 2015. (In the table below numbers show thousands of units). Source: Gartner (May 2015).

[19] Keyraider: iOS malware steals over 225,000 Apple accounts to create free app Utopia, http://researchcenter.paloaltonetworks.com/2015/08/keyraider-ios-malware-steals-over-225000-apple-accounts-to-create-free-app-utopia/, last accessed on: 2015-09-14

[20] Palo Alto Networks , https://www.paloaltonetworks.com/, last accessed on: 2015-10-19

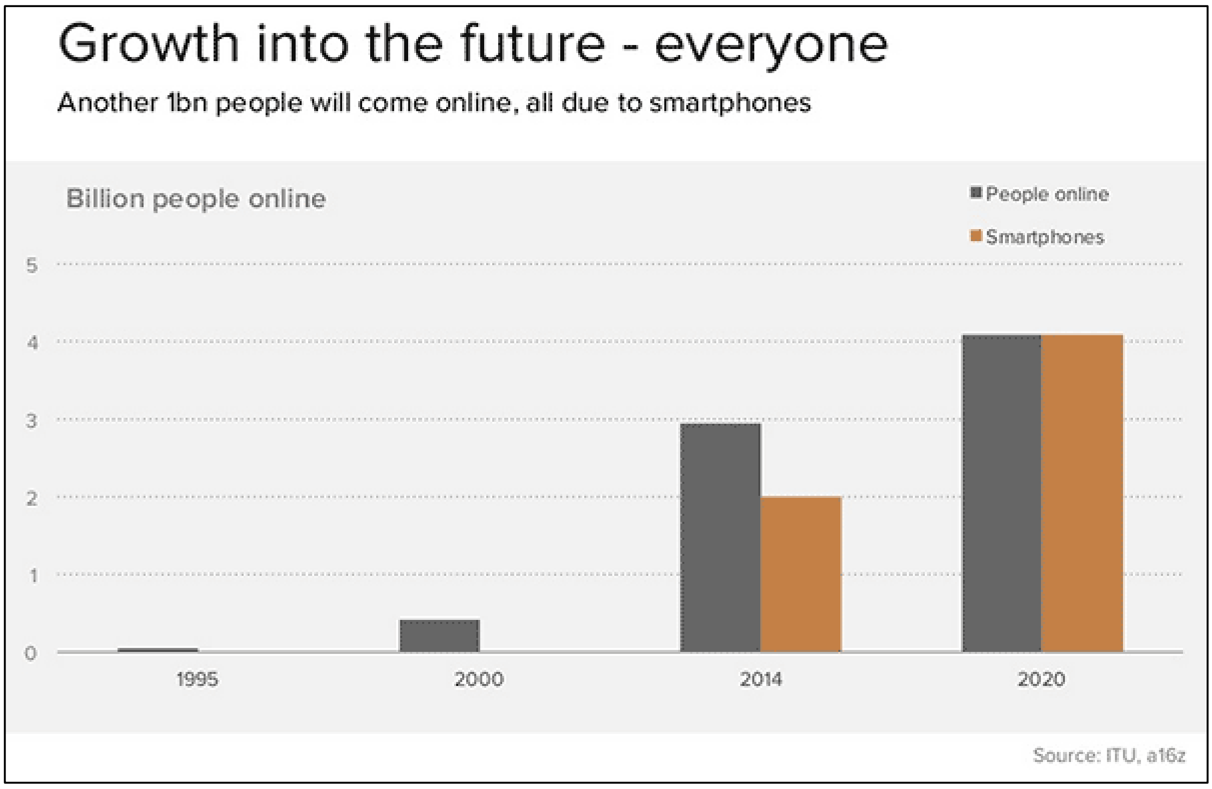


Figure 2: Worldwide comparison between people online and active smartphones. Source: ITU, a16z (2014).

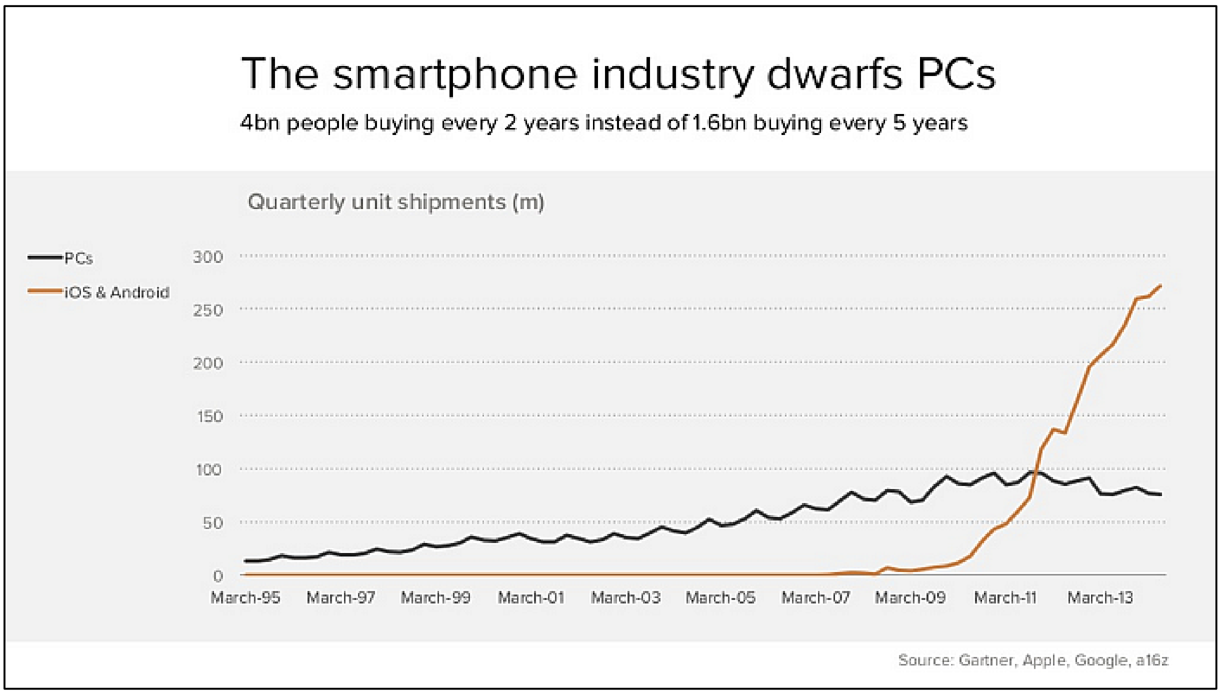


Figure 3: Worldwide comparison between selling desktop PC's and smartphones. Source: Gartner, Apple, Google, a16z (2014).

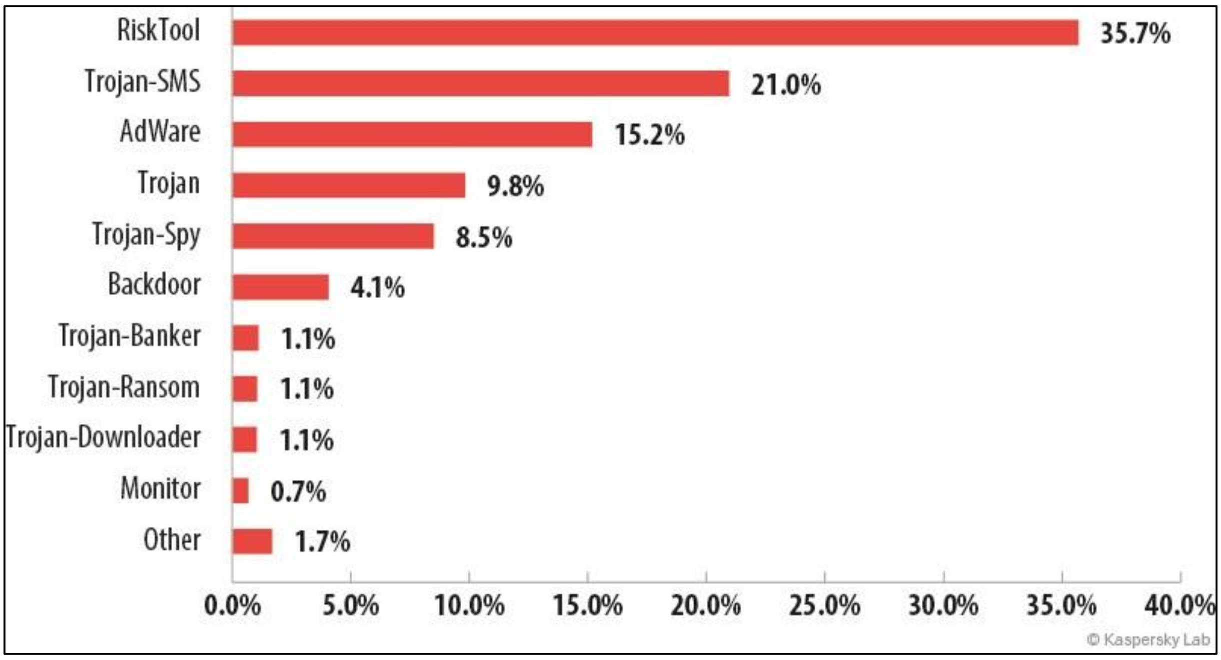


Figure 4: Distribution of new mobile malware by type, Q1 2015. Source: Kaspersky Lab (May 2015).

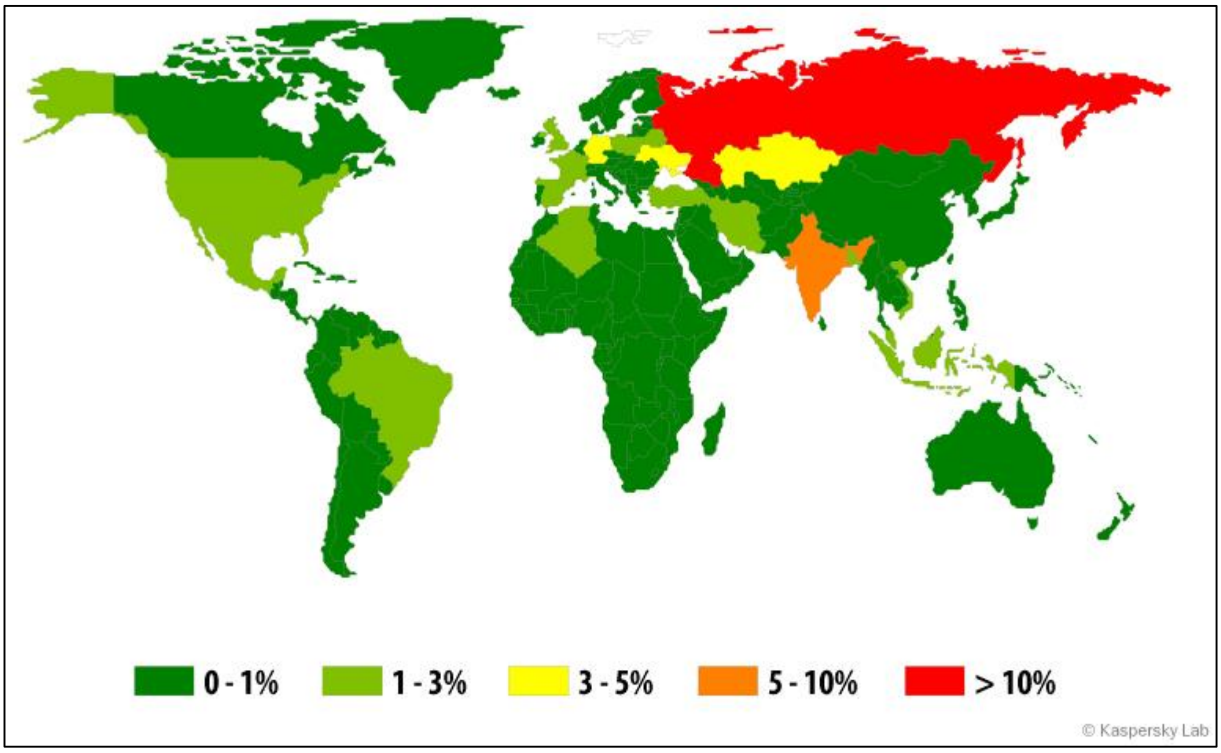


Figure 5: Geography of mobile threats. Source: Kaspersky Lab (May 2015).

COUNTRY	% OF ATTACKED USERS
Russia	45.7%
India	6.8%
Kazakhstan	4.1%
Germany	4.0%
Ukraine	3.0%
Vietnam	2.7%
Iran	2.3%
UK	2.2%
Malaysia	1.8%
Brazil	1.6%

Table 2: Top 10 countries by number of attacked users. Source: Kaspersky Lab (May 2015).

NAME	% OF ATTACKS
Trojan-SMS.AndroidOS.Stealer.a	18.0%
RiskTool.AndroidOS.MimobSMS.a	7.1%
DangerousObject.Multi.Generic	6.9%
RiskTool.AndroidOS.SMSreg.gc	6.7%
Trojan-SMS.AndroidOS.OpFake.bo	6.4%
AdWare.AndroidOS.Viser.a	5.9%
Trojan-SMS.AndroidOS.FakeInst.a	5.4%
Trojan-SMS.AndroidOS.OpFake.a	5.1%
Trojan-SMS.AndroidOS.FakeInst.fb	4.6%
Trojan-SMS.AndroidOS.Erop.a	4.0%
AdWare.AndroidOS.Ganlet.a	3.8%
Trojan-SMS.AndroidOS.Agent.u	3.4%
Trojan-SMS.AndroidOS.FakeInst.ff	3.0%
RiskTool.AndroidOS.Mobogen.a	3.0%
RiskTool.AndroidOS.CallPay.a	2.9%
Trojan-SMS.AndroidOS.Agent.ao	2.5%
Exploit.AndroidOS.Lotoor.be	2.5%
Trojan-SMS.AndroidOS.FakeInst.ei	2.4%
Backdoor.AndroidOS.Fobus.a	1.9%
Trojan-Banker.AndroidOS.Faketoken.a	1.7%

Table 3: Top 20 mobile threats of 2014. Source: Kaspersky Lab (May 2015).

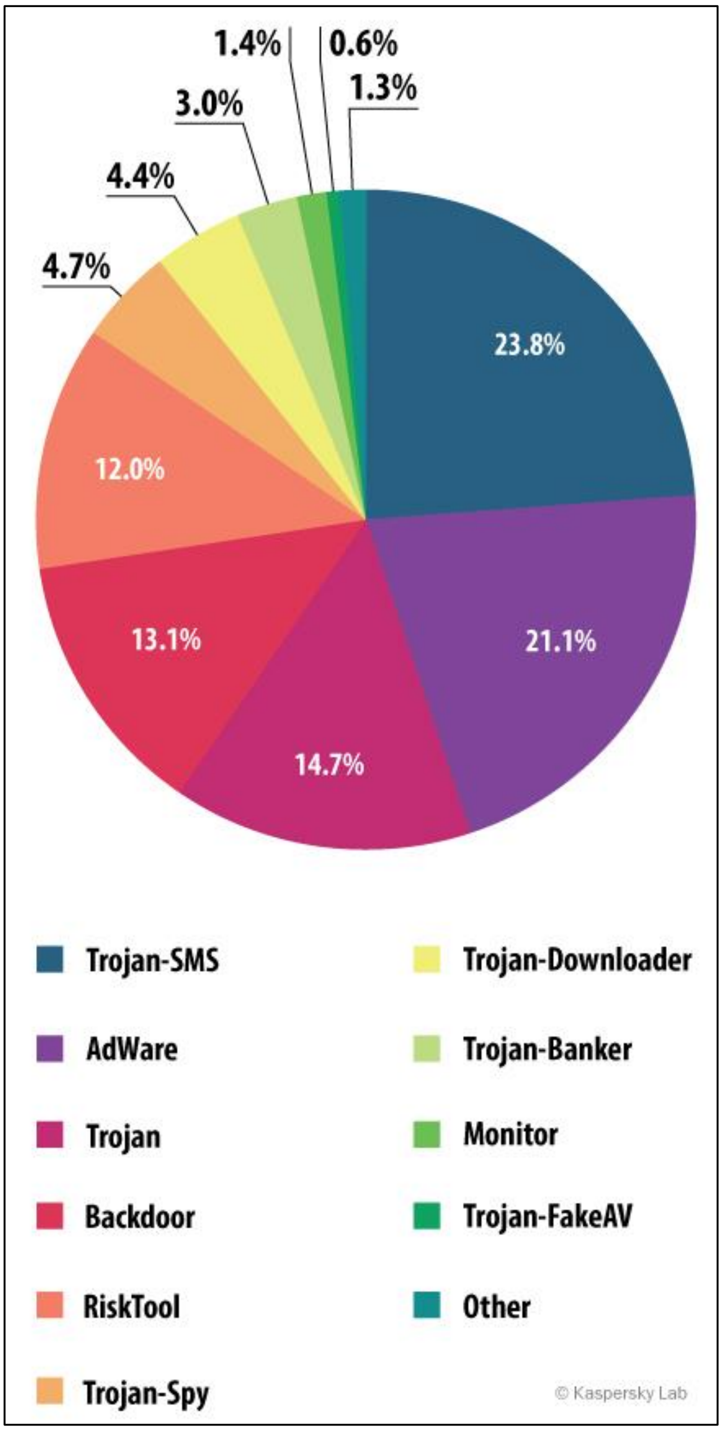


Figure 6: Distribution of mobile threats by type. Source: Kaspersky Lab (May 2015).

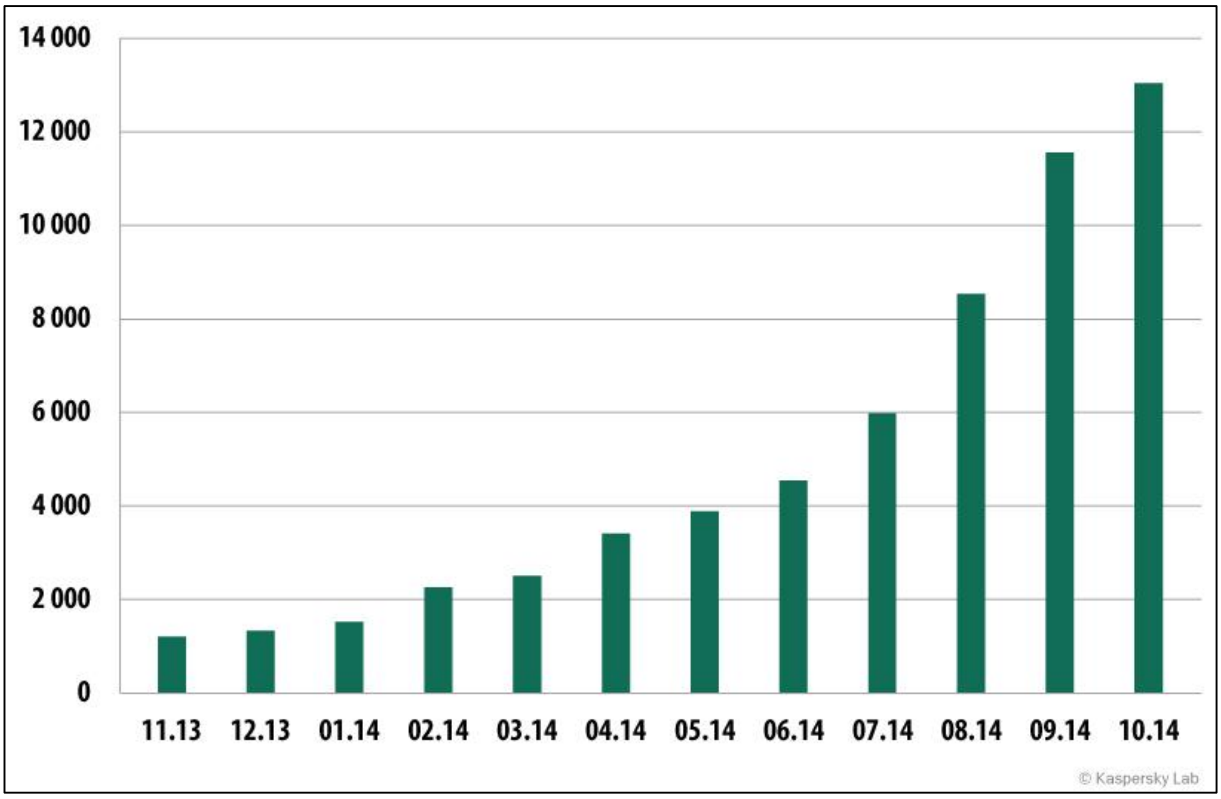


Figure 7: Number of mobile banking Trojans in the Kaspersky Lab collection (November 2013 — October 2014). Source: Kaspersky Lab (December 2014).

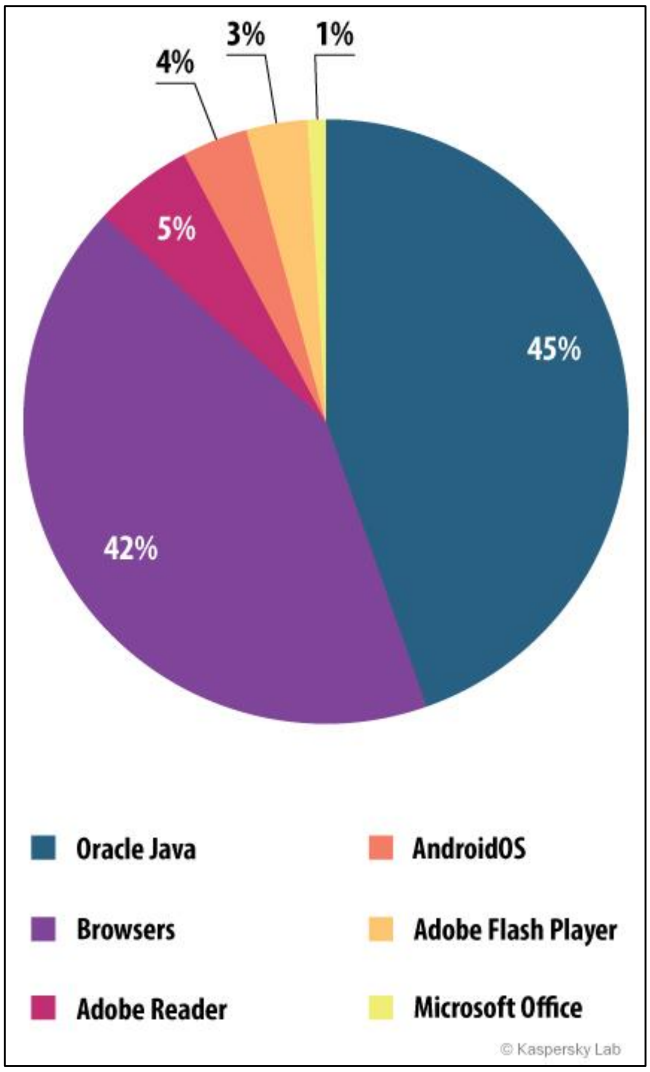


Figure 8: The distribution of exploits used by fraudsters, by type of application attacked. Source: Kaspersky Lab (December 2014).

COUNTRY	% OF UNIQUE USERS
Russia	53.81%
Kazakhstan	53.04%
Azerbaijan	49.64%
Vietnam	49.13%
Armenia	48.66%
Ukraine	46.70%
Mongolia	45.18%
Belarus	43.81%
Moldova	42.41%
Kyrgyzstan	40.06%
Germany	39.56%
Algeria	39.05%
Qatar	38.77%
Tajikistan	38.49%
Georgia	37.67%
Saudi Arabia	36.01%
Austria	35.58%
Lithuania	35.44%
Sri Lanka	35.42%
Turkey	35.40%

Table 4: The top 20 countries where users face the greatest risk of online infection. This statistic are based on the detection verdicts returned by the web antivirus module, received from users of Kaspersky Lab products who have consented to provide their statistical data. Source: Kaspersky Lab (December 2014).

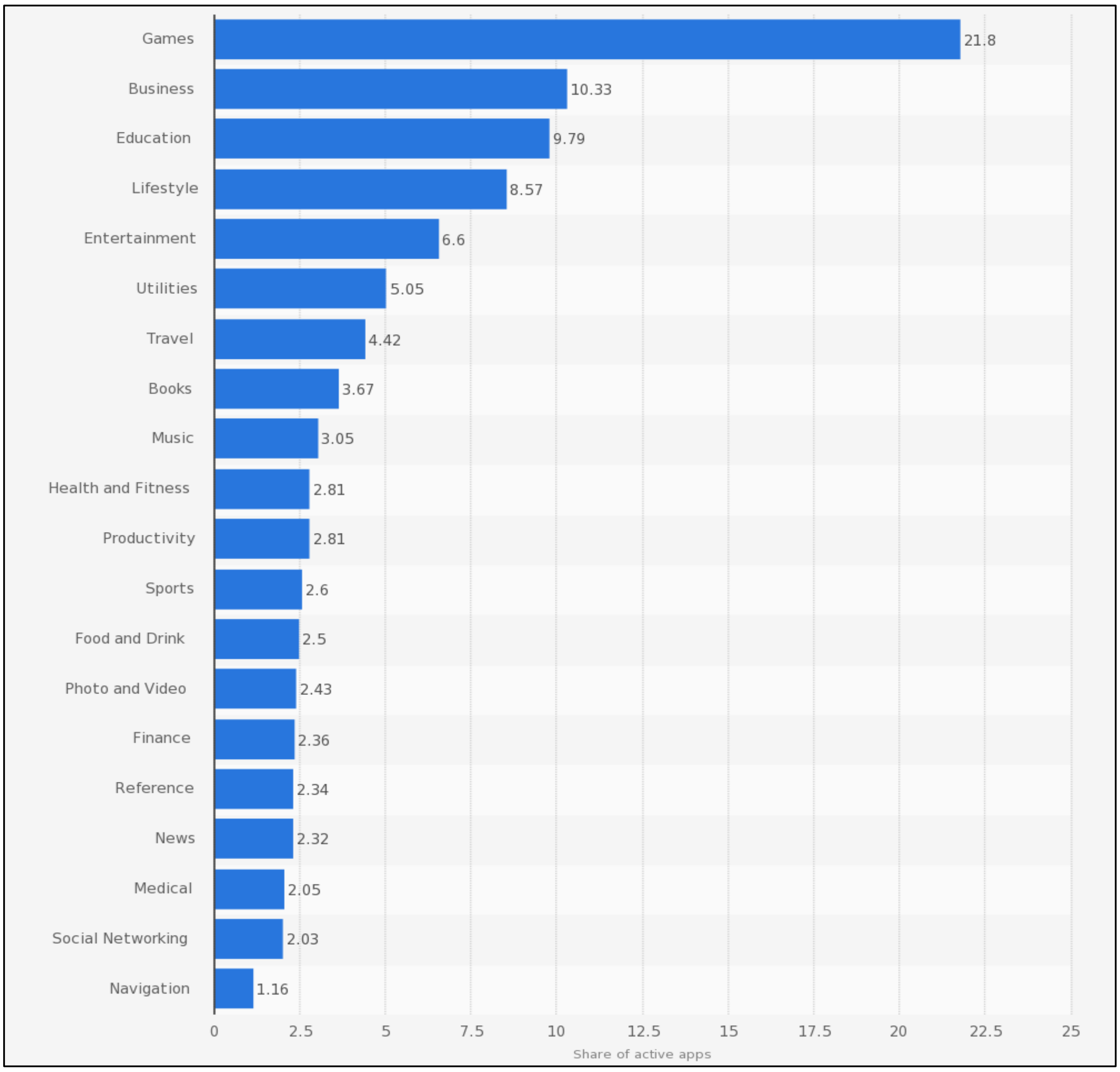


Figure 9: Most popular Apple App Store categories in June 2015, by share of available apps. Source: http://www.statista.com/statistics/270291/popular-categories-in-the-app-store/.

1.9 Rooting of Android-based devices

Rooting is the process that allows Android devices to overcome the limitations that manufacturers and distributors of equipment and software put on them. As a result of rooting, user of a device obtains the "super-user" privileges and the ability to change or replace system applications, load his or her own software (i.e. ROM), install custom or unofficial applications or enhance system performance. A user of a rooted device is able to perform operations that are not normally available to a usual user of a non-rooted device, such as dumping (i.e. creating) a full system image, overclocking processors (CPU or GPU) etc.

There are two methods of Android devices rooting:

- Permanent – used in most cases, aiming at replacing systems kernel to permanently take control over the device.
- Temporary – mostly used by commercial mobile forensics software, in which case a usual restart of a device will cause it to reverse to un-rooted state.

Instruction of how to root an Android device are easily available on the Internet. For now the simplest method to root an Android-based device is to use a "one-click" software.[21] User have to install it on their PC, connect an Android device to it and run software to perform root. There is even a website[22] allowing for remote rooting of Android-based devices.

1.10 Jail-breaking of iOS-based devices

Jailbreak is a combination of English words "jail" and "break" and as the name suggests, this process is associated with a breach of security. This generally means removing imposed limitations by uploading a modified kernel to devices such as iPhone, iPod touch, iPad and Apple TV. Jailbreak allows full access to the device and installation of applications or extensions that are not normally available through the official distribution source — Apple's App Store. The device after the jailbreak can still use the App Store, iTunes, and other normal functions however.

There are three methods of jailbreaking of iOS devices:

- Tethered jailbreak ("bound jailbreak") is an unlocking software for iOS, which retains its functionality until the iDevice is shut down. To reactivate the system, user must connect the iDevice to the computer and perform the jailbreak again using the appropriate program. Until this happens the device is not suitable for using.
- Semi-tethered jailbreak is a specific jailbreaking method. It means that when the previously jailbroken iDevice restarts, it will no longer have a modified kernel but it will still be usable for normal functions. The user will have to perform jailbreaking again to access the device without limitations.
- Untethered jailbreak ("resolved, released") is a fully functional modification of a systems' kernel that works even when user reboots the iDevice. Untethered jailbreak is not available for all versions and for all devices.

So far all Apple smartphones have vulnerabilities allowing for a jailbreak (e.g. identified and published by the following groups: Redsn0w,[23] Evasi0n,[24] Pangu,[25] TaiG[26]).

[21] One-click Android root software, http://www.kingoapp.com/android-root/feature-one.htm, last accessed on: 2015-09-14
[22] PerfectRoot’s Remote Android Rooting Services ensure increased security and higher functionality such as increased memory, battery life, system speeds, and custom app capability, http://perfectroot.com/, last accessed on: 2015-09-14
[23] iOS 8 jailbreak, http://www.redsn0w.us/, last accessed on: 2015-09-14
[24] evasi0n7 Jailbreak for iOS 7, http://evasi0n.com/, last accessed on: 2015-09-14
[25] Pangu Jailbreak for iOS 8, http://en.pangu.io/, last accessed on: 2015-09-14
[26] iOS 8.4 jailbreak, http://www.taig.com/en, last accessed on: 2015-09-14

2. Threats and incidents handling

Threats to mobile devices can affect mobile users without their knowledge and sometimes even without interaction. It is necessary to understand the nature of mobile-related threats that have impact on the most popular mobile platforms: iOS and Android. For this purpose, the following section explains security mechanisms implemented in those systems, followed by the identification of mobile applications properties and permissions. Mobile applications are the most common source of vulnerabilities and therefore the weakest point of any mobile device.

Once a vulnerability is exploited and the threat impacts the device and its user, special procedures like incident handling may be required. The nature of mobile threat incident handling is explained in another ENISA training exercise.[27] [28]

2.1 Threat analysis

Mobile devices, technically and generally, do not differ very much from their non-mobile counterparts. Although they use special hardware and dedicated software, the overall nature of vulnerabilities in both are similar as they result from design or implementation errors, misconfiguration, user negligence, and similar factors. Years ago, threats targeting mobile devices used to be different but with smartphones and tablets becoming more and more powerful and connected to the Internet, the difference starts to fade away. Hence, mobile devices and software is prone to malware and spyware attacks, authentication and authorisation bypass attempts, DoS attacks, and similar threats.

Due to the mobile nature of smartphones and tablets, they are filled with data of various types, including: pictures and videos, e-mail and contacts, call records and text messages, notes and task lists, location-related data, application-specific information (e.g. passwords, codes, and payment data). These phones are often used as a second factor of authentication or authorisation which makes them especially interesting for cybercriminals.

On the hardware level, mobile phones are prone to attacks targeting mobile network encryption, identification or access control mechanisms. With core parts of mobile networks still based on old SS7 protocol suite, they are prone to attacks exploiting vulnerabilities that just are not easy or even possible to patch: compare SS7 to SMTP, a three-decade old mail transfer protocol which by design has no counter-measures against spam or sender's address spoofing.

Some threats make use of the fact that mobile phones are still a bit less functional, less powerful, have limited GUI or battery life-span as compared to PCs and laptops. The simplified design of software architecture and its size sometimes require a compromise between security needs and user expectations. A good example is an anti-virus software which is a de facto standard for PCs and laptops, while many, if not most of mobile users have no AV installed on their smartphones. Quite possibly some of them are not even aware that AV software is available for some mobile devices.

Many security and supervision features must be implemented and controlled by external solutions such as Mobile Devices Management systems explained in section 8.3.

[27] Mobile threats incident handling — Toolset, Document for students, https://www.enisa.europa.eu/activities/cert/support/exercise/files/Mobilethreatsincidenthandlingtoolset.pdf, last accessed on: 2015-09-14
[28] Mobile threats incident handling — Handbook, Document for teachers, https://www.enisa.europa.eu/activities/cert/support/exercise/files/Mobileincidenthandlinghandbook.pdf, last accessed on: 2015-09-14

Once a threat exploits an existing vulnerability and a security incident happens, there is a need to analyse the root cause, impact and the threat itself. Basically there are two ways to approach this task:

- Static analysis of mobile applications code and configuration. For this purpose, applications need to be reverse-engineered i.e. decompiled and disassembled to understand what procedures and operations the application performs.
- Dynamic analysis through observation and assessment, in a controlled environment, of how the mobile application or the exploit behaves. Typically it means identification of network connections, inter-process communications, access to mobile device's assets and components, file and folders creation, modification or deletion.

The following are a few recommendations that will minimize the chance of falling victim to particular threats:

- Install applications that provide security functions on the mobile device.
- Do not download software from unauthorised or unofficial sources.
- Before installing an application, check the access permissions it is requesting.
- Upgrade firmware, operating system and applications when informed to do so.
- Turn on the phone lock code, if possible, and set complex password or pattern.
- Do not hold important data on your mobile device unencrypted.
- Check the popularity of an application and user reviews before downloading and installing it.
- Keep mobile devices in sight, do not leave them without supervision.
- Do not connect the mobile device to unknown computers or chargers.

2.2 Vulnerabilities

Vulnerabilities are software bugs that can be exploited by malware or attackers to take control of the system, to damage or get information.

According to Symantec, as of 2014,[29] the number of vulnerabilities in iOS is much higher than that of Android.

[29] Symantec's Internet Security Threat Report, https://www4.symantec.com/mktginfo/whitepaper/ISTR/21347932_GA-internet-security-threat-report-volume-20-2015-social_v2.pdf, last accessed on: 2015-09-14

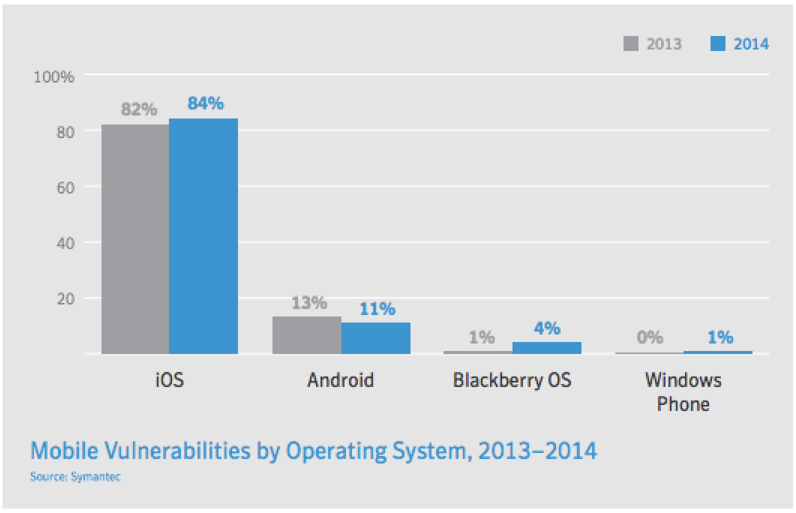


Figure 10: Statistics on vulnerabilities per mobile platform. Source: Symantec.

Apple apply strict policies to their application market and despite the large number of vulnerabilities in iOS, the number of applications that can exploit them is minimal (the control is however limited when an iOS Device is jailbroken).

While the chart above shows a huge difference in vulnerabilities per platform, Apple, through their App Store, can control and prevent spread of malicious applications. Google, on the other hand, is facing a challenge, with a more open applications installation policy and large fragmentation of devices and software markets.

2.3 Encryption mechanisms in Android and iOS

Preventing unauthorised access to a mobile device is an essential step in its protection, however this is often overlooked by the users. Different platforms offer different protection systems.

Android has six screen lock systems: swipe, Face Unlock (by camera), pattern, PIN (only digits), password and fingerprint-based system.

iOS provides three methods: complex password, PIN code (four to six digits) and fingerprint reader (TouchID).

Fingerprint authentication is an easy-to-use solution and more robust than most of the alternatives. However it used to have its security problems on iOS[30] [31] and Android[32] platforms.

The following chart shows a comparison of different access control means in terms of their convenience and security:

[30] Why I Hacked Apple's TouchID, And Still Think It Is Awesome, https://blog.lookout.com/blog/2013/09/23/why-i-hacked-apples-touchid-and-still-think-it-is-awesome/, last accessed on: 2015-09-14
[31] Why I hacked TouchID (again) and still think it's awesome, https://blog.lookout.com/blog/2014/09/23/iphone-6-touchid-hack/, last accessed on: 2015-09-14
[32] Fingerprints on mobile devices: abusing and leaking, https://www.blackhat.com/us-15/briefings.html#fingerprints-on-mobile-devices-abusing-and-leaking, last accessed on: 2015-09-14

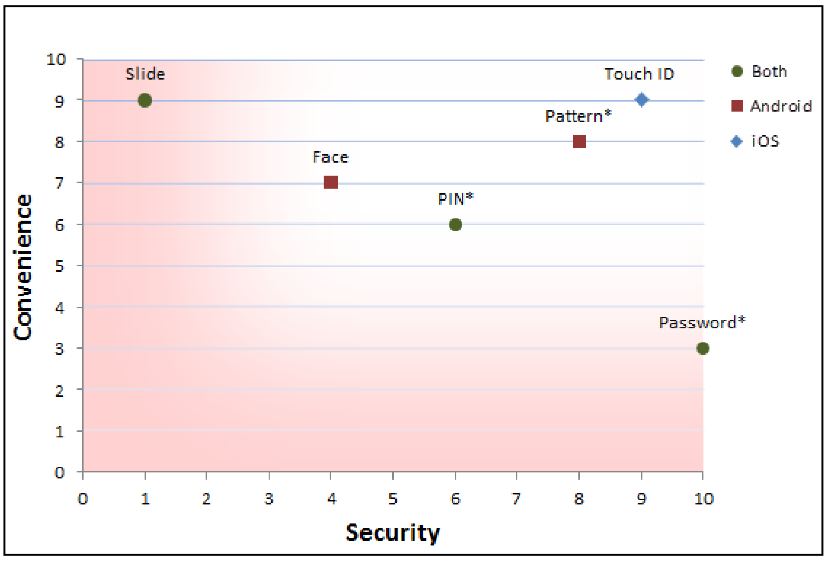


Figure 11: Source: SOFTONIC INTERNACIONAL S.A.

2.3.1 Encrypting user data

iOS-based devices have an encryption mechanism implemented on the hardware level, thus minimising negative impact on performance. The key is tied to the UID which is a 256-bit key that is fused into the application processor during manufacturing. This key is stored in a dedicated area of NAND storage called ‘effaceable storage’ which is used to store cryptographic keys.

Android, on the other hand, due to its openness and hardware / software fragmentation makes encryption implementation a challenge. Besides, users can choose if they want to have their devices encrypted. The impact on performance can be significant.[33]

Samsung devices with Android can include the KNOX system,[34] which is a piece of hardware and software that creates 'safe container' with its own security mechanisms besides standard Android security systems.

2.4 Threat analysis on iOS

To better understand iOS threats it is necessary to know more about iOS application security:[35]

- Every mobile application is installed in its own container. Should a successful exploitation happen, a compromised application and the attacker cannot get out of the container (unless the vulnerability allows for jailbreaking which is explained in section 8.5).
- Applications run as the "mobile" user.
- A keychain for outside the Sandbox is provided by Apple (for password or sensitive data storage).
- Applications can only access Keychain content for the application.
- Applications can only be downloaded from Apple's App Store.

The architecture is shown on a picture below:

[33] Google quietly backs away from encrypting new Lollipop devices by default, http://arstechnica.com/gadgets/2015/03/google-quietly-backs-away-from-encrypting-new-lollipop-devices-by-default/, last accessed on: 2015-09-14
[34] Samsung KNOX system , https://www.samsungknox.com/en, last accessed on: 2015-10-19
[35] iOS Security Guide, https://www.apple.com/business/docs/iOS_Security_Guide.pdf, last accessed on: 2015-09-14

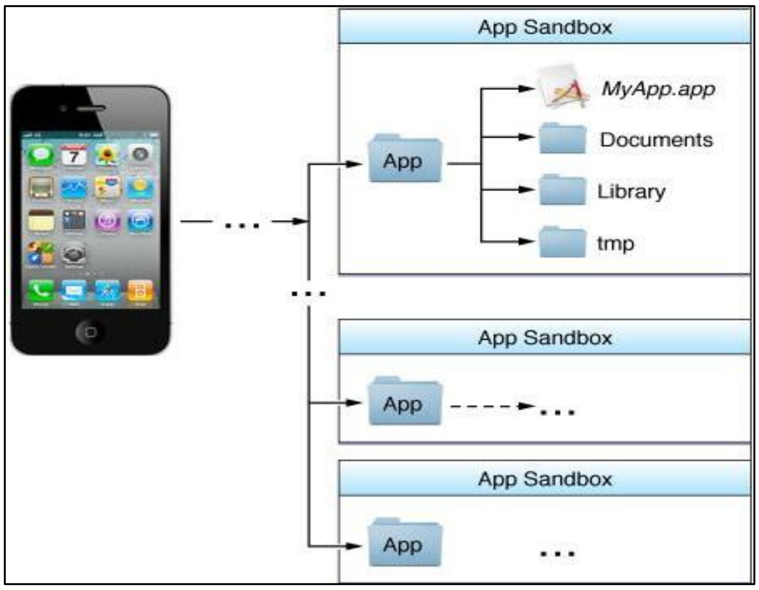


Figure 12: Source: https://crypto.stanford.edu/cs155/lectures/17-mobile-platforms.pdf

The App Store's walled garden model is very important because of its security implications. From the end-user's point of view it can be a bit cumbersome to download applications only from one source, but it has a positive impact on the security of the applications (and end-users):

- It is the only place for an unmodified iOS (i.e. non-jailbroken) to purchase and download applications.
- The App Store is controlled by Apple.
- All applications must be reviewed by Apple before public release.
- A released application can be removed from the App Store if it violates policies.

The above list is only applicable if the iOS device is not jailbroken. In general — to prevent iOS infections, only non-jailbroken systems should be used.

Several threats specifically target jailbroken devices. The biggest problem with jailbreaking is that it disables the sandboxing feature of iOS (explained in section 8.1). It means that applications have root (i.e. almost unlimited) privileges and hence can access other applications data or even deeper parts of iOS.

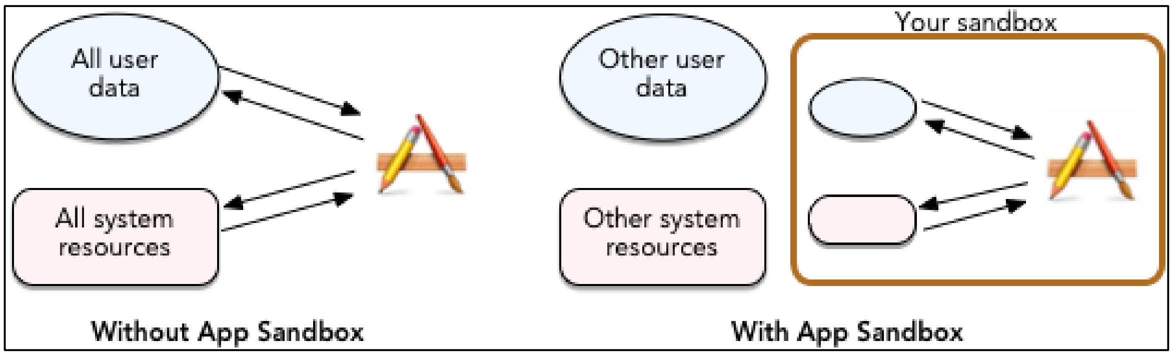


Figure 13: Source: http://macnews.tistory.com/233

One of the worst-kept secrets on the Internet is the default root password for iOS. A quick online search will reveal this password, and it is known that Apple has not changed it for several years. Therefore when a device is compromised a hacker will have a chance to connect via SSH and pull out all the user data due to almost unlimited privileges that root user has.

It is still possible to safely use a jailbroken iOS device. However, only when a user is experienced and realises the potential risks. The following are some security tips for users with jailbroken devices:

- Change the root password to something else than well-known default password. Most malware for jailbroken iPhones relies on the fact that only a few jailbreakers change the root password.
- Download and install anti-virus software on your iDevice because unofficial applications repositories like Cydia has many dangerous applications (e.g. KeyRaider[36] which stole 225 000 Apple users accounts).
- Be aware of what you are installing, since many applications trick users into installing malware.

2.5 Threat analysis on Android

Android is one of the most popular operating systems for smartphones and tablets. The popularity of the system is not without significance for the creators of malicious programs who use its considerable popularity. It is estimated that, just in the last year, hundreds of malicious software programs were created for Android operating system.

Malware installation can be prevented by following a few basic principles and taking care of installing the proven anti-virus application. Applications should be installed from an official store — Google Play. That being said, Android implements multiple layers of protection to keep malware at bay and prevent data or systems from being compromised.

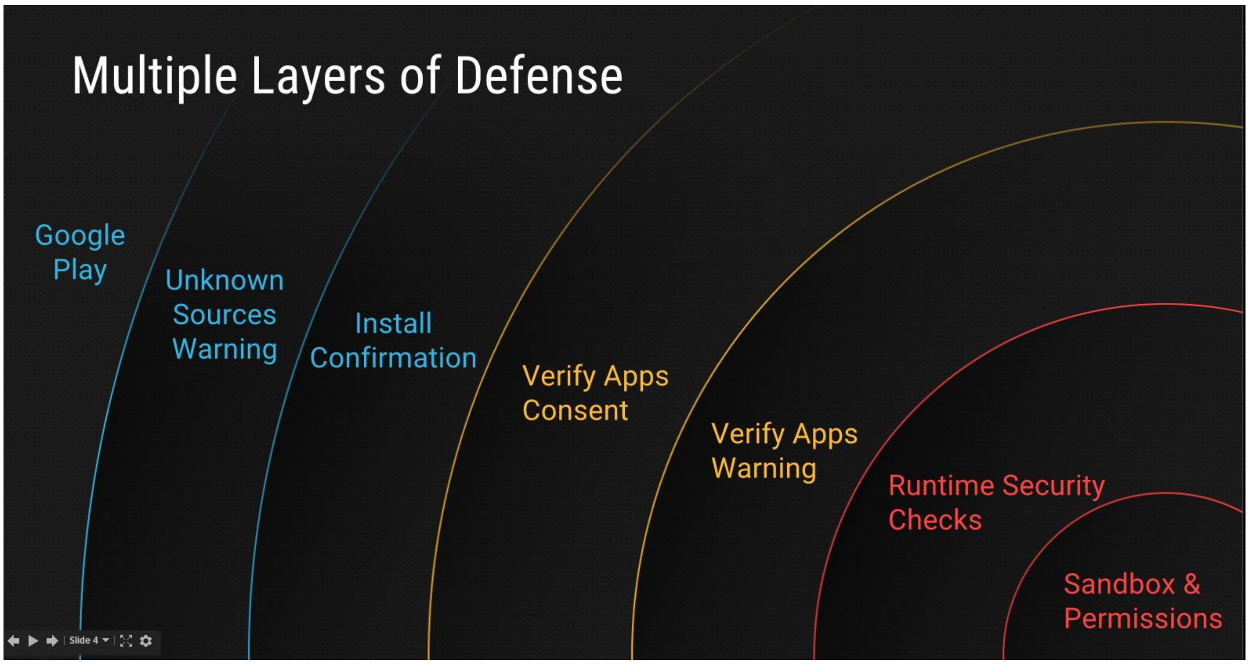


Figure 14: Source: https://docs.google.com/presentation/d/1YDYUrD22Xq12nKkhBfwoJBfw2Q-OReMr0BrDfHyfyPw/pub#slide=id.g1202bd8e5_073

[36] Keyraider: iOS malware steals over 225,000 Apple accounts to create free app Utopia, http://researchcenter.paloaltonetworks.com/2015/08/keyraider-ios-malware-steals-over-225000-apple-accounts-to-create-free-app-utopia/, last accessed on: 2015-09-14

One effective method of protection is KNOX, developed by Samsung, and made available for devices by this vendor only. KNOX is the solution for improving the security of Samsung devices running Android in the mid- to high-end. Factory installation of Android on these devices will allow access to a multi-tiered security architecture that make up the entire platform — Samsung KNOX. The term "tiered" here means nothing less than providing possibly the best protection for smartphone or tablet from the operating system's boot process to running and using an application by the user. The key elements of this multi-layered security model include: Secure Boot, Trusted Boot, Security Enhancements for Android, TIMA, and KNOX Container. The architecture is shown on a picture below:

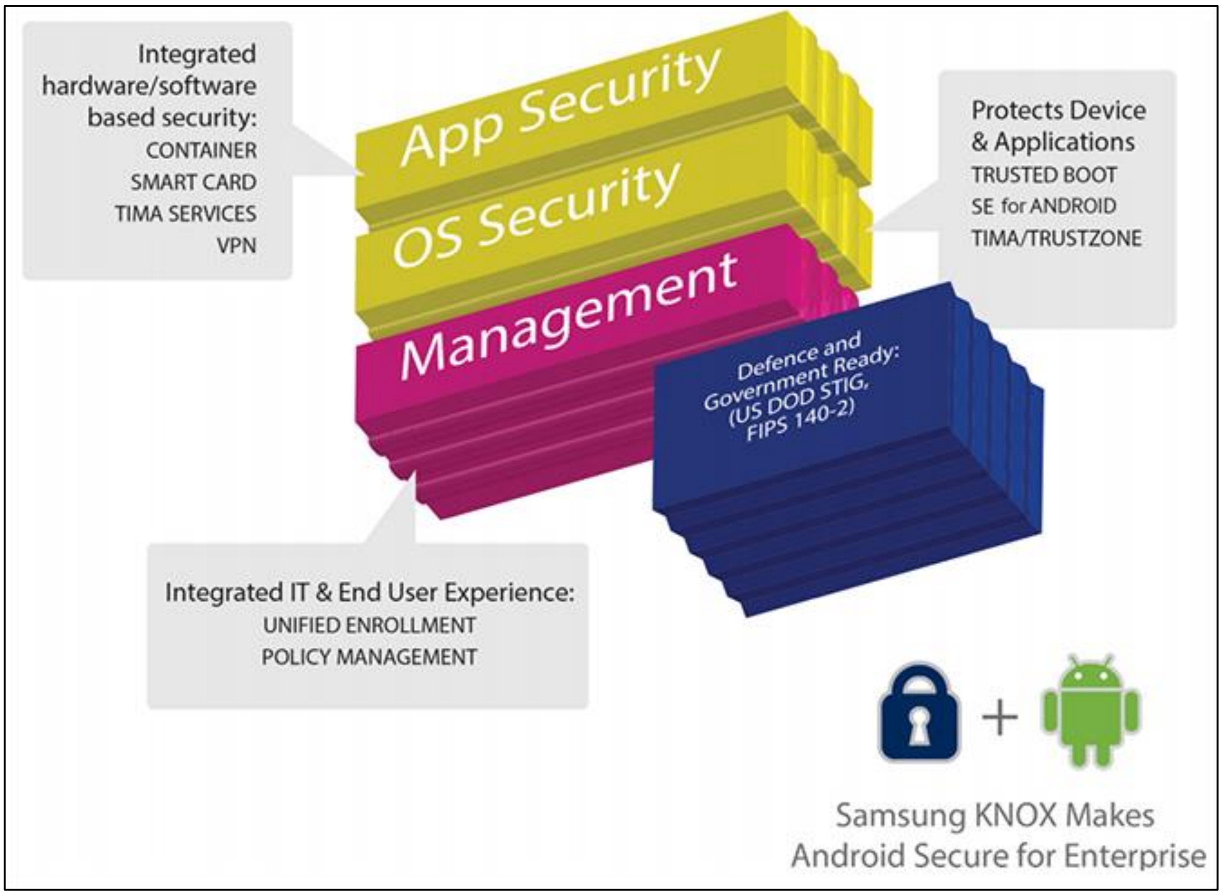

Integrated hardware/software based security:
CONTAINER
SMART CARD
TIMA SERVICES
VPN
App Security
OS Security
Management
Protects Device & Applications
TRUSTED BOOT
SE for ANDROID
TIMA/TRUSTZONE
Defence and Government Ready: (US DOD STIG, FIPS 140-2)
Integrated IT & End User Experience:
UNIFIED ENROLLMENT
POLICY MANAGEMENT
Samsung KNOX Makes Android Secure for Enterprise

Figure 15: Source: http://www.samsung.com/ca/business-images/resource/white-paper/2014/03/Samsung_KNOX_tech_whitepaper_Final_140220-0.pdf

The first protective layer in the three-layer security model of Samsung KNOX is called a Security Platform and includes: Secure Boot, Trusted Boot, Security Enhancements for Android and TIM.

Secure Boot, together with Trusted Boot is the primary security mechanism designed to protect against loading into RAM memory an unauthorised version of Android.

Two other functions are important for system integrity: Security Enhancements for Android and a solution called TIM. The former consists of specific access control lists for applications that run on Android. SE for Android was introduced into the system by Google with version 4.4 KitKat. TIMA (TrustZone-based Integrity Measurements Architecture) is used to verify the integrity of the Android kernel. Memory areas are verified in real-time against any modification of the kernel by malicious software. In addition to hardware security mechanisms, Samsung KNOX also provides data protection measures used by various applications running on the device. The first of these are services based on the TIMA architecture: CCM and the KeyStore. TIMA

CCM (Client Certificate Management) functions relates to encryption, decryption and signing data within KNOX. The second mechanism is the TIMA KeyStore. It enables key generation and storage needed to encrypt data for applications running on KNOX.

The final element in the KNOX application security model is the KNOX Container. Generally this is an additional, separate operating environment provided by the Android system and made available to the user. Once it starts, it provides a totally separate home screen, launcher, installed applications and widgets. Such isolation means that applications residing outside of the container by default are not able to access what is inside of it. Moreover, applications within the container cannot communicate with external processes and the general Android environment.

2.6 Task 2.1: Analysis of sample application's permissions on an Android device

2.6.1 Introduction

In this task, the students will use native Linux instrument called **AAPT**[37] which allows to take a look into permissions of the sample application. The AAPT tool can be used to list, add or remove resource files from apt packages (i.e. Android applications). If can also dump specific data from the packages.

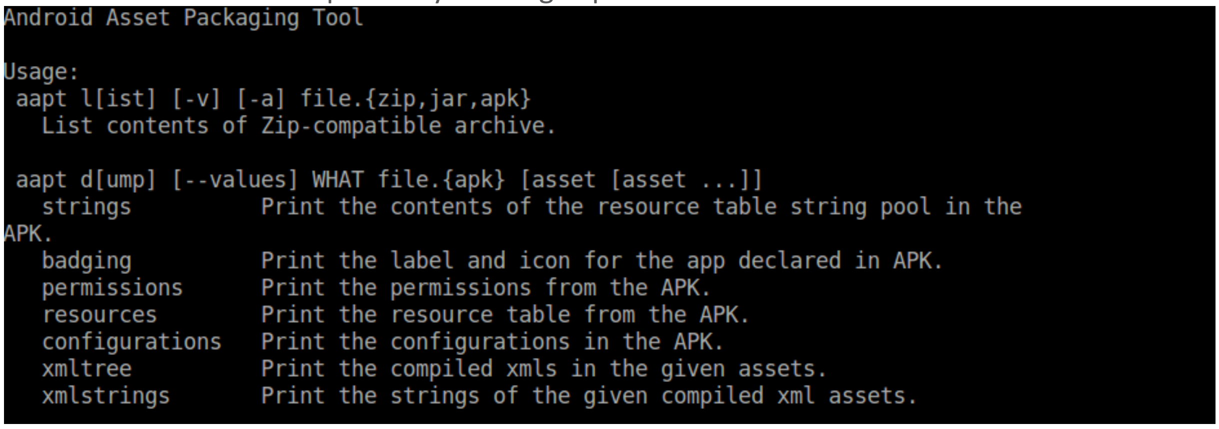
2.6.2 Details

In the exercise directory (/home/enisa/D2/2.6_T1) students will find an APK application file com.androidream.secretdiary.free.apk. For the analysis of this file students will have to use the pre-installed AAPT tool.

2.6.3 Task walk-through

In this section a possible approach to permissions' analysis is explained.

2.6.3.1 Take a look at the tool's options by running aapt.

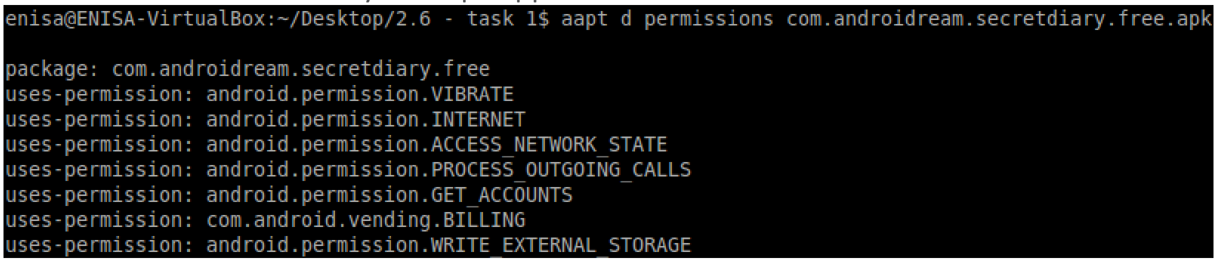


```
Android Asset Packaging Tool

Usage:
 aapt l[ist] [-v] [-a] file.{zip,jar,apk}
   List contents of Zip-compatible archive.

 aapt d[ump] [--values] WHAT file.{apk} [asset [asset ...]]
   strings          Print the contents of the resource table string pool in the
APK.
   badging          Print the label and icon for the app declared in APK.
   permissions      Print the permissions from the APK.
   resources        Print the resource table from the APK.
   configurations   Print the configurations in the APK.
   xmltree          Print the compiled xmls in the given assets.
   xmlstrings       Print the strings of the given compiled xml assets.
```

Figure 16

[37] Build System Overview, http://developer.android.com/sdk/installing/studio-build.html, last accessed on: 2015-09-14

2.6.3.2 Use command "aapt d permissions" to view the permissions of com.androidream.secretdiary.free.apk application.



```
enisa@ENISA-VirtualBox:~/Desktop/2.6 - task 1$ aapt d permissions com.androidream.secretdiary.free.apk

package: com.androidream.secretdiary.free
uses-permission: android.permission.VIBRATE
uses-permission: android.permission.INTERNET
uses-permission: android.permission.ACCESS_NETWORK_STATE
uses-permission: android.permission.PROCESS_OUTGOING_CALLS
uses-permission: android.permission.GET_ACCOUNTS
uses-permission: com.android.vending.BILLING
uses-permission: android.permission.WRITE_EXTERNAL_STORAGE
```

Figure 17

Compare permissions granted to this application to all available permissions for Android applications[38] and describe what this specific application can do.

2.7 Task 2.2: Analysis of sample application's Mach-o header on an iOS device

2.7.1 Introduction

In this simple task the students will use a tool called **OTOOL** which allows to take a look into Mach-O header of the sample iOS application. This application is available only for Mac OS X platform.

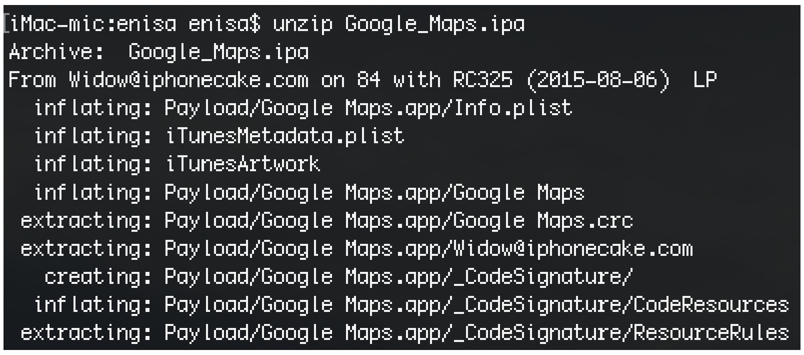
2.7.2 Details

You will have to use the otool on the sample iOS application. You will find information about the FAT header to identify the supported CPU architecture to run an application. It is important to know how to run the application if it is needed to put the application into a sandbox.

2.7.3 Task walk-through

Students will need to download .IPA file directly from an iPhone or from the Internet. After that they will need to unzip the .IPA file then check information from the FAT file header.

2.7.3.1 Unzip the .IPA file.

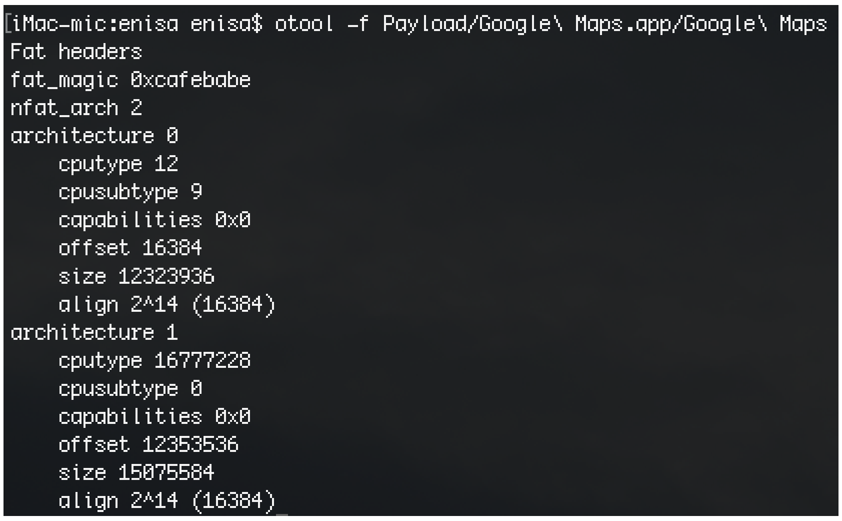


```
[iMac-mic:enisa enisa$ unzip Google_Maps.ipa
Archive:  Google_Maps.ipa
From Widow@iphonecake.com on 84 with RC325 (2015-08-06)  LP
  inflating: Payload/Google Maps.app/Info.plist
  inflating: iTunesMetadata.plist
  inflating: iTunesArtwork
  inflating: Payload/Google Maps.app/Google Maps
 extracting: Payload/Google Maps.app/Google Maps.crc
 extracting: Payload/Google Maps.app/Widow@iphonecake.com
   creating: Payload/Google Maps.app/_CodeSignature/
  inflating: Payload/Google Maps.app/_CodeSignature/CodeResources
 extracting: Payload/Google Maps.app/_CodeSignature/ResourceRules
```

Figure 18

[38] Android applications permissions, http://developer.android.com/preview/features/runtime-permissions.html, last accessed on: 2015-09-14

2.7.3.2 Use otool to check the FAT header.



```
[iMac-mic:enisa enisa$ otool -f Payload/Google\ Maps.app/Google\ Maps
Fat headers
fat_magic 0xcafebabe
nfat_arch 2
architecture 0
    cputype 12
    cpusubtype 9
    capabilities 0x0
    offset 16384
    size 12323936
    align 2^14 (16384)
architecture 1
    cputype 16777228
    cpusubtype 0
    capabilities 0x0
    offset 12353536
    size 15075584
    align 2^14 (16384)_
```

Figure 19

Compare the FAT_MAGIC value with the Mach-O documentation[39] and answer the question: is the binary for a 32-bit platform, 64-bit platform, or are the binaries universal?

- 32-bit (ARMv6, ARMv7) – 0xFEEDFACE
- 64-bit – 0xFEEDFACF
- Universal binaries – 0xCAFEBABE

[39] Universal Binaries and 32-bit/64-bit PowerPC Binaries, https://developer.apple.com/library/mac/documentation/DeveloperTools/Conceptual/MachORuntime/index.html#//apple_ref/c/tag/fat_header, last accessed on: 2015-09-14

3. Mobile Forensics

3.1 Concepts and principles

As explained earlier, mobile forensics is a set of complex techniques aiming at the delivery of digital evidence based on data extracted from mobile devices. As such, it utilises approaches, technologies and tools known from traditional computer forensics. Some of the concepts and solutions are common for both, while others are specific to mobile forensics. The two groups of principles and precautions are explained below.

3.1.1 Common principles

- Chain of custody

Chain of custody is a way of ensuring the integrity, authenticity and quality of evidence during the investigation. Even minor errors in how the evidence have been handled (in technical and physical terms) can result in a court of law rejecting the evidence. It is a good practice to protect evidence from the beginning of the investigation. This means adequate handling, usage of proper tools, techniques and solutions as well as detailed and complete documentation.

- Documentation

Every action or event related to the evidence should be documented, described or if necessary pictured or filmed. It helps investigators keep track of what has been done and allows independent experts to follow the same steps to verify results.

- Thinking ahead

It is very important to try to predict future actions, possible outcomes of the processes and behaviour of tools and devices, and be ready to respond to challenges. This comes with experience and sometimes with a bit of luck. Example of such situations includes: a new type of connector / interface, low battery level, encrypted device, on-going data transmission etc.

- Proper equipment

It is good to know beforehand what equipment will have to be prepared. This helps to make sure suitable hardware is ready, e.g. a PC with fully charged battery and free space for extractions, SIM card reader, empty SIM cards for cloning, memory card reader with write-blocking support, wires, Faraday bag, and software.

- Training

Mobile forensic investigators should be trained before they start performing any activity. It is necessary to understand not only mobile devices' architectures and the technologies behind them, but also the principles of forensic investigation.

3.1.2 Unique principles

Although mobile devices are yet another example of electronic equipment, mobile in their nature and complex in implementation, they impose certain principles upon those who perform forensic operations against them.

- Do not turn off powered-on devices

As opposed to an often used approach to computer forensics, in most cases mobile devices should be kept turned on. Due to locking mechanisms, a device once turned off, will require a PIN or a password to allow access (excluding unencrypted external storage media e.g. SD cards). If, upon device's seizure, it is up and running and unlocked, it is a good idea to disable any automatic screen lock-out mechanism. While this means tampering with the device itself, this step is necessary if one wishes to perform any other forensic actions on the device. This operation must be documented in detail.

- Isolate network connections

Devices under investigation, before the acquisition happens, should be prevented from communication with external devices (except for data acquisition) and over a wireless network. A suspect could, for example, remotely wipe a device's content. Enabling flight mode, in general, is a good idea, but in some cases this may prevent an investigator from using the wired interface to extract data. Instead a Faraday bag or a cloned SIM card without cellular connection can be used. Mobile phone users often add configuration for publically available or office wireless networks. This must be taken into consideration when planning the acquisition process and properly addressed e.g. by identifying locally available WiFi networks and preventing the device from connecting to them.

- Proper identification

Proper identification of a device is very important for a successful extraction. Many mobile phones vendors differentiate models with minor additions such as i8190 and i8190N, but in some cases those are completely different phones. Knowing the suspect device model will help to prepare for the investigation. Apart from that, proper identification and documentation is part of a chain of custody process.

- Overvoltage protection

Electronic devices, especially SIM cards, are very fragile and susceptible to damage. Always work in anti-static gloves and with extreme caution.

- Post-mortem forensics

Post mortem forensics is an acquisition technique employed when the device is turned off and no knowledge about codes, password, operating system or a (most recent) SIM card is available. There are mobile devices that detect insertion of a SIM card different from the previous one and which may delete or hide information e.g. call history. In such cases, if IMSI and ICCID numbers are known, the SIM card can be cloned (i.e. its duplicate can be made based on the IMSI and ICCID) and the device turned on for logical or file system acquisition (provided it is not protected by a password).

- Even better documentation

Even though, technically, mobile devices are computers, they cannot be treated the same way in terms of forensic examination. Apart from unique "features" of forensic activities explained above, a very important difference is that access to mobile devices content cannot be "write-blocked", in other words cannot be done read-only. While this is, in almost any case, a prerequisite in computer forensics, with mobile devices' logical extraction, it only makes sense to do it when a device is running. This leads to another caveat: all operations must be very, very well documented, since most of them result in tampering with the device (e.g. running mobile applications, changing files properties or modification dates, modification of RAM memory contents etc.) As for now, there are no ways to "write-block" access to the device or clone it for further investigation. While physical extraction allows for a bit-by-bit imaging of internal and external storage media, it is a complex task and with encrypted devices it will be of almost no use.

3.2 Mobile forensics tools

There are many commercial tools for the purposes of mobile forensics e.g. NowSecure Forensics,[40] AccessData's MPE+,[41] Cellebrite UFED,[42] EnCase Forensic,[43] Micro Systemation XRY,[44] FINALDATA

[40] NowSecure Forensics, https://www.nowsecure.com/forensics/, last accessed on: 2015-10-19
[41] AccessData's MPE+, http://accessdata.com/solutions/digital-forensics/mpe, last accessed on: 2015-10-19
[42] Cellebrite UFED, http://www.cellebrite.com/Mobile-Forensics, last accessed on: 2015-10-19
[43] EnCase Forensic, https://www.guidancesoftware.com/products/Pages/encase-forensic/overview.aspx, last accessed on: 2015-10-19
[44] Micro Systemation XRY, https://www.msab.com/products/xry/?gclid=CJ3I47uMzsgCFerpwgod6BED2g, last accessed on: 2015-10-19

FINALMobile Forensics,[45] Lantern 3,[46] Logicube CellXtract,[47] MOBILedit! Forensic,[48] Oxygen Forensic Suite,[49] Paraben Device Seizure,[50] Phone Forensics Express,[51] Radio Tactics' Athena,[52] Secure View 3.[53]

Some tools have been developed to explicitly address increasing criminal usage of phones manufactured based on Chinese chipsets (e.g. MediaTek (MTK), [54]Spreadtrum [55]and MStar[56]). Such tools include Cellebrite's CHINEX,[57] eDEC's Tarantula[58] and XRY PinPoint.[59] Some other vendors also have added support to some of Chinese phones to their software.

A number of open-source tools provide basic and complex functions supporting mobile device investigations. Most of them, however, are platform-specific, which means that they support certain phone types and in most cases focus on smartphones. Such tools include: iPhone Analyser,[60] TULP2G,[61] Open Source Android Forensics.[62] It is worth mentioning the Santoku Linux[63] distribution which offers a prepared Linux image with mobile forensics tools. In the later sections of this document, some of the most useful tools will be presented.

Another way of accessing mobile device content is the physical extraction which can be done with a flashing tool, used for programming (i.e. flashing) of the device memory e.g. EEPROM or flash memory. While most often used for servicing or debugging, some can also be used for forensic purposes. Explanation of the use of such tools is outside of the scope of this document, however in a later section, storage mechanisms are presented and described in detail.

3.3 Examples of data sources

The following section explains the sources of information coming directly from mobile devices and from mobile operators.

3.3.1 Mobile devices as a source of data

3.3.1.1 Digital evidence

Digital evidence is a piece of information, relevant to the case being investigated, which exists in a digital form. It can be an electronic document, picture, multimedia file, chat history, database, log record, executable (e.g. malware) file etc. as well as a bit-by-bit image of a hard drive, memory card or phone memory.

[45] FINALDATA FINALMobile Forensics, http://www.finaldata.com/, last accessed on: 2015-10-19
[46] Lantern 3, http://shop.osxforensics.com/, last accessed on: 2015-10-19
[47] Logicube CellXtract, http://www.logicube.com/shop/cellxtract/, last accessed on: 2015-10-19
[48] MOBILedit! Forensic, http://www.mobiledit.com/forensic, last accessed on: 2015-10-19
[49] Oxygen Forensic Suite, http://www.oxygen-forensic.com/en/, last accessed on: 2015-10-19
[50] Paraben Device Seizure, https://www.paraben.com/device-seizure.html, last accessed on: 2015-10-19
[51] Phone Forensics Express, http://www.mobiledit.com/forensic-express, last accessed on: 2015-10-19
[52] Radio Tactics' Athena, http://www.radio-tactics.com/, last accessed on: 2015-10-19
[53] Secure View 3, http://secureview.us/, last accessed on: 2015-10-19
[54] MediaTek (MTK), http://www.mediatek.com/, last accessed on: 2015-10-19
[55] Spreadtrum , www.spreadtrum.com/, last accessed on: 2015-10-19
[56] MStar, www.mstarsemi.com/, last accessed on: 2015-10-19
[57] Cellebrite's CHINEX, http://www.cellebrite.com/Pages/ufed-chinex, last accessed on: 2015-10-19
[58] eDEC's Tarantula, https://www.edecdf.com/product/tarantula-cell-phone-extraction-kit/ , last accessed on: 2015-10-19
[59] XRY PinPoint, https://www.msab.com/products/pinpoint/, last accessed on: 2015-10-19
[60] iPhone Analyser, http://www.crypticbit.com/zen/products/iphoneanalyzer, last accessed on: 2015-10-19
[61] TULP2G,http://tulp2g.sourceforge.net/, last accessed on: 2015-10-19
[62] Open Source Android Forensics, http://www.osaf-community.org/, last accessed on: 2015-10-19
[63] Santoku is dedicated to mobile forensics, analysis, and security, and packaged in an easy to use, Open Source platform, https://santoku-linux.com/, last accessed on: 2015-09-14

Digital (a.k.a. electronic) evidence must be authentic and relevant to the case under the investigation. Authenticity is ensured by following a command of chain of custody and employing forensically sound procedures to identify, seize, secure, store, analyse and present information being digital evidence.

3.3.1.2 IMEI (International Mobile Equipment Identity)

The IMEI number is an identifier of each mobile device and has a form of 15–16 digit code, the construction of which is as follows:

- 2 digits – manufacturer's code
- 6 digits – phone model code
- 6 digits – serial number
- The last digit is a checksum (based on Luhn algorithm)

The first 8 digits of an IMEI create a TAC (Type Allocation Code) which is used by a particular model of a device. The IMEI number can be read from every mobile phone by dialling the code *#06# in the dial pad.

3.3.1.3 ICCID (Integrated Circuit Card ID)

The ICCID number is a number that identifies the SIM card and is usually applied in whole or in part on its surface. ICCID number can be read from the SIM card without knowing the PIN or PUK code. The ICCID number has the form of a 19–20 digit code, with the following structure:

- 2 digits – industry identifier (e.g. for telecommunication it is 89)
- 1–3 digits – country code
- 1–4 digits – issuer code
- Variable length – the unique card number
- The last digit – checksum (based on Luhn algorithm)

3.3.1.4 IMSI (International Mobile Subscriber Identity)

The IMSI number identifies a specific subscriber and is provided in tandem with an IMEI number when the subscriber is registering into the GSM network. It is possible to read this number from a SIM card with the appropriate hardware and software, but only after entering the correct PIN or PUK code. The IMSI is in the form of a 15 digit code and has the following structure:

- 3 digits – the country code (MCC – Mobile Country Code)
- 2–3 digits – operator code (MNC – Mobile Network Code)
- Remaining digits – subscriber code

3.3.1.5 PIN code

The PIN code that protects access to the SIM card consists of 4 to 8 digits. The PIN code can be changed by the user. Three wrong PIN attempts block access to the SIM card. To unlock a SIM card, the PUK code must be entered.

3.3.1.6 PUK code

The PUK is an 8-digit security code to unlock a SIM card after three wrong PIN attempts. The PUK is known for a particular network operator issuing a SIM card. Once the code is entered incorrectly ten times, the SIM card is permanently blocked and even the issuer of the SIM card cannot unlock it.

3.3.2 Mobile device memory storage as a source of data

Mobile devices usually have three types of memory: internal (phone) memory, SIM card memory and external (e.g. SD card) — explained later in the document.

3.3.2.1 Phone memory

Older phones stored simple files (e.g. notes, pictures, videos, audio files). Smartphones and tables nowadays, store much more including, but not limited to:

- Contacts database
- SMS messages
- MMS messages
- E-Mail messages
- Recent calls list
- Pictures, video and audio files
- Schedules, memos, calendar, tasks
- Geolocation files
- Application-specific files

3.3.2.2 SIM card memory

Memory of SIM cards is very limited. Most SIM cards are sold in 32 or 64 kilobyte varieties, but it is possible to have more capacity on the SIM card. It can store the following type of information:

- Contacts database
- SMS messages
- SIM Card Identity Information, such as integrated circuit card identifier (ICCID), international mobile subscriber identity (IMSI), Authentication Key, Location Area Identity (LAI) and an Operator-Specific Emergency Number

When securing data off mobile devices, it is important to fetch information from all types of memory listed above (with the exception of a triage technique explained earlier in the document).

3.3.3 Mobile operator as a source of information

Mobile forensics make extensive use of data processed and provided by mobile operators. This may include billing and call data records (CDRs), subscriber's data, tariffs and services configuration, IP traffic records, roaming data, voicemail recordings, a device's geolocation and services-specific information. It is worth noting, that in most cases, mobile operators do not store text messages and call recordings with the exceptions being non-delivered text messages (for a period of time), voicemail recordings and communications of a subscriber under surveillance from law enforcement agencies.

Mobile forensic investigators must be careful when analysing data provided by mobile operators. Techniques such as Caller-ID spoofing, encrypted data connection carrying voice traffic, complex call routing, SS7 vulnerabilities (mentioned earlier in the document) and sim-boxing frauds may make data unreliable and require verification. In other words: the nature of a global telecommunication network, makes it susceptible to data manipulation thus resulting in possibly fake call data records, changed / spoofed source mobile numbers or hiding communications within other type of network traffic. Criminals often carry many old or simple mobile devices to avoid leaving traces or clues to investigators. Even though such devices and their subscribers still can be subject to surveillance or locating, the simplicity and low price of those devices can help increase the level of operational security of the criminals, which make forensic investigators' work more difficult.

Therefore, any analysis in a mobile devices-related investigation should make use (if possible) of both sources of information to allow for combination, verification and reasoning during the analysis.

3.4 Task 3.1: A quick evaluation of knowledge regarding mobile devices

Please answer the following questions. Only one answer is correct in each question.

1. What information is contained in the IMEI number? (Correct answer: a)
   a) The manufacturer's code
   b) Name of operator
   c) The number of home network
   d) None of the above

2. The ICCID number is: (Correct answer: d)
   a) The number identifying the SIM card
   b) The serial number of the SIM card
   c) Number, which can be read without knowing the PIN
   d) All of the above

3. The IMSI identifies: (Correct answer: a)
   a) Subscriber
   b) Phone
   c) The SIM card
   d) The telephone

4. To disable communication capabilities of a seized mobile device which is turned on: (Correct answer: d)
   a) Insert the device into the overvoltage bag
   b) Separate it from the network by pulling out SIM card
   c) Turn it off
   d) Put it in a Faraday's bag and analyse as soon as possible

5. How to check the IMEI of a device which is turned on? (Correct answer: a)
   a) By entering *#06#
   b) By entering *##06#
   c) By entering *#08#
   d) By entering *##08#

6. What does "post mortem" extraction mean? (Correct answer: b)
   a) Device is bricked
   b) Device is turned off
   c) Extraction will damage the device
   d) Device is turned on

7. How can the integrity of electronic evidence be ensured? (Correct answer: d)
   a) By burning extracted data to a read-only medium
   b) By a checksum
   c) By following chain of custody
   d) All of the above

4. Mobile forensic procedures

Mobile phones, like many other mobile devices, have an internal memory on which the operating system is installed and which allows the user to work with a mobile phone. The memory inside the appliance is also used to store private user data, such as list of contacts and text messages and multimedia messages, images and other types of media files. Apart from the internal memory, mobile devices often make use of an external one e.g. SD memory cards. Forensic activities focus on the extraction of data from a physical media: internal and external memory storage. The following section will explain how data is stored on a mobile device and how the extraction process works. The tasks at the end of this chapter will help students understand, how this knowledge can be applied to forensic analysis i.e. the purpose of data extraction, how this can be achieved and what are its outcomes.

4.1 Explanation of logical and physical extractions

Mobile devices use a flash memory, which is a type of an EEPROM (i.e. electrically erasable programmable read-only memory), which allows to store or delete multiple memory cells during a low-level programming operation. This memory is non-volatile which means stored data remains even after the battery is removed. Depending on the type of a logic gate used, there are two types of flash memories.

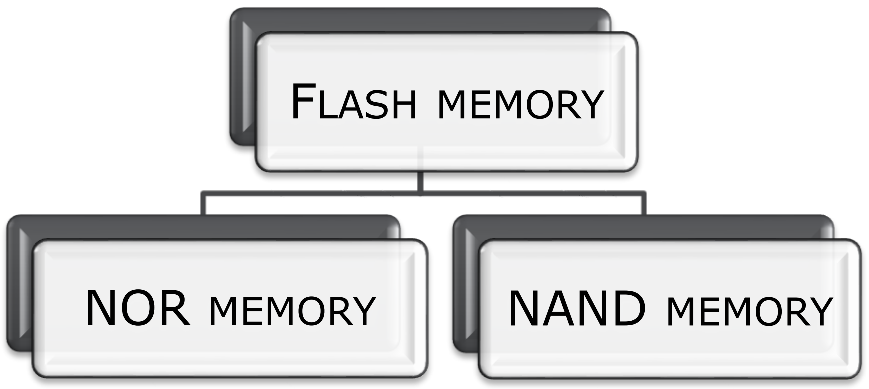


Figure 20: Types of flash memory

NOR-type memory allows a direct access to each memory cell but it needs a relatively long time for write and erase operations. For this reason NOR-type memory is good for storing data that do not require frequent updates, such as firmware.

For NAND-type memory the write and erase time is shorter on the other hand, and it has greater data density. The main feature of this type of a memory is the sequential access to data which limits its use mainly to storage.

In order to save data into the flash memory cell, it has to be deleted beforehand. It is not possible to write data to an already used cell. While it is possible to read and write any memory cell, the erasing operation is only possible for entire blocks of cells. It is not possible to delete an individual cell and for this reason data recording has limitations.

Flash memory records must be coordinated with the operation of erasing memory blocks. Usually if a file is updated or overwritten, the memory management system creates a new copy of the file in another location, marking its previous version as useless and the memory cells with old content as unused. However the old version of the file still occupies space. This is the reason, why in some cases it is possible to recover "deleted" data.

During the analysis of mobile devices for forensics purposes, data can be read twofold:

- From within the user-space (i.e. user accessible programs)
- By performing advanced analysis of the entire memory, taking into account the areas marked as free

While the former method does not usually seem to be forensically sound (as it tampers with evidence), the latter is preferred and for this reason explained in the following sections. It has to be noted however, that live methods (e.g. working from within the user-space) can be justified (they can be the only possible means of data extraction in given circumstances such as sophisticated encryption of the mobile device) and when done properly can provide useful and high-quality evidence in the case.

The following schema shows various types of extraction and analysis of data from mobile device's memory:

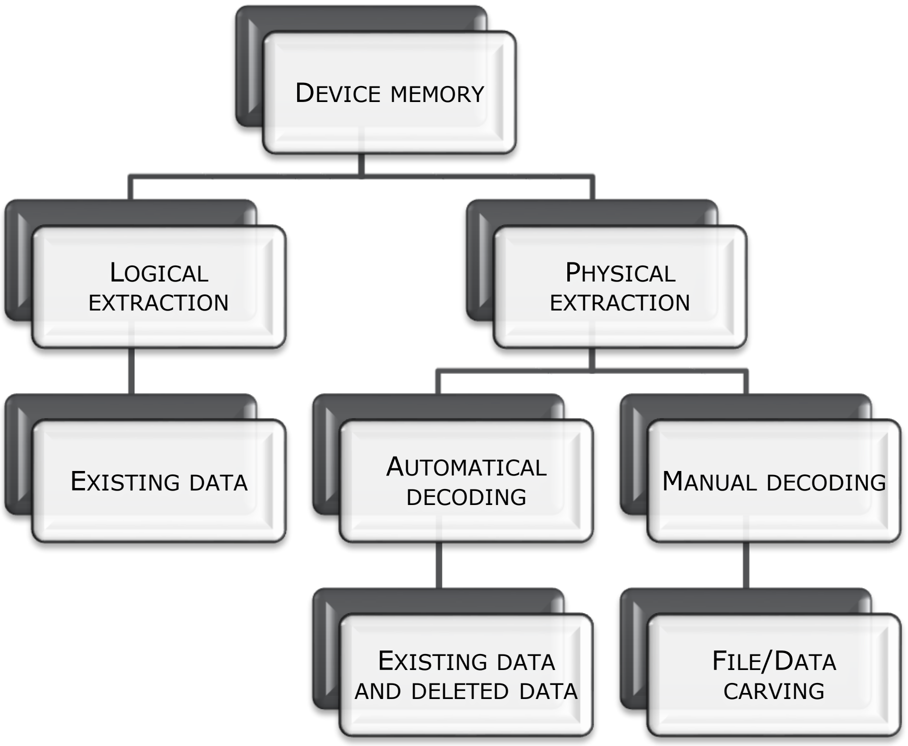


Figure 21: Types of data extraction and analysis

Depending on the case each method can prove useful. In some cases it is necessary to recover partially deleted or overwritten data, hence physical extraction will come in handy. When it is expected to review a certain set of information (e.g. current configuration), logical extraction may make more sense. However, there are many factors that help decide which method will be most efficient:

- Encryption
- Availability and compatibility of extraction software or devices
- Legal constraints
- User's "cooperation"
- Time constraints
- Scope and objectives of the analysis

Physical extraction is the first step in the analysis of data (including deleted and recovered files and folders). The next step is to use an appropriate decoding algorithm to enable the proper interpretation of extracted data. Those two steps are universal since any mobile platform makes use of data encoding (which is not the same as encryption!). The differences stem from the varied algorithms, techniques and technologies used by different vendors.

4.2 Best practices and techniques

It is very important to properly prepare the mobile device for data extraction and ensure optimal conditions. All components must be checked before the process starts to minimise the risk of any errors. It must also be noted that all steps (including pre-checks, exceptions and anomalies) must be well documented. The purpose of the documentation is to prove that the extraction process was performed in a forensically sound manner and given the same circumstances and by following the same procedure will provide same results every time.

However, in some cases the extraction can be done only once. That is why it is crucial to make sure all necessary equipment is ready, fully operational and at hand.

4.2.1 Battery and power supply

When securing mobile devices it is very important to ensure the appropriate level of battery or provide external power supply. The best solution is to have own, proven battery pack providing stable reading conditions. When the mobile device is collected at a different place than the extraction process will take place, it is important to make sure that the device does not run out of battery in the meantime. However, charging should be done from a power supply connected to a grid (e.g. mains socket), and not from the computer's USB port.

4.2.2 Communication interfaces

Regardless of the type of a communication interface, it should always be kept clean and dry. In case of any contamination the connector must be cleaned from dust, dirt or any signs of corrosion. A clean connector provides a stable connection and eliminates most errors occurring during the process of data reading. Mobile devices collected at a remote site should be placed in special bags protecting them from dust and damp.

However, mobile devices are also able to communicate wirelessly. If such communication is not desired, but the device cannot be tampered with (e.g. by disabling communications or turning flight-mode on), the device should be placed in a bag that is a Faraday cage blocking any electric fields — the wireless communication in particular.

4.2.3 Communication cables

Equally important is the use of a proper cable during the extraction. The equipment must be of a high quality, perfectly fit to the connector (original parts are preferred to "original copies"), to ensure a stable connection.

4.2.4 Premises

In some cases a long-running data extraction process has to be restarted due to an accidental unplugging of either the power supply (e.g. for write-blocking devices), or a cable connected to mobile devices. To avoid such situations, desks where data extraction takes place must be clean, and offices should be only attended by the authorised staff.

4.2.5 Software

A forensic analyst needs to be prepared for any type of data extraction. Apart from the necessary hardware, it is equally important to have an adequate software. The software should have automatic data decoding functionality which significantly accelerates whole analysis. An important element is the module that gives the ability to manually search for information based on keywords, as well as group and sort certain types of artefacts (e.g. by file or application type, by date, by size, by sender or receiver etc.)

4.3 Physical analysis

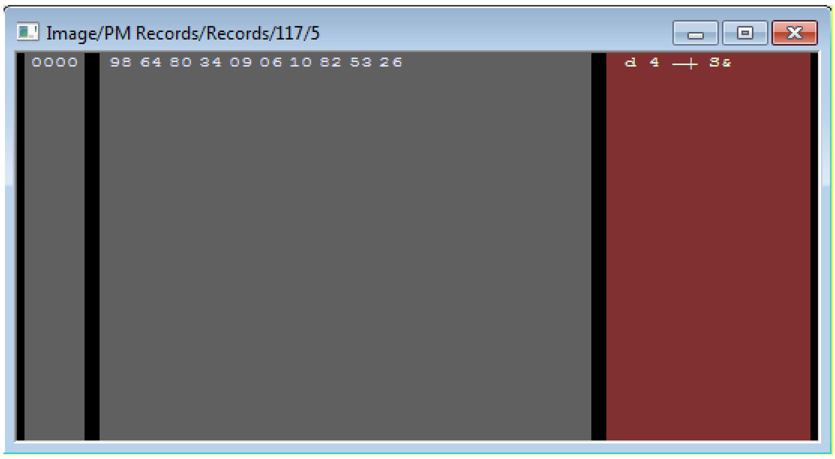
Physical extraction allows for analysis of the whole memory of a device. Therefore there are some differences from a logical extraction approach not only in terms of actual techniques or tools, but also the scope and form of data available for extraction.

4.3.1 Unique techniques in physical analysis

A SIM card, as explained earlier, stores certain types of information. Its form, however, needs some explanation.

4.3.1.1 SIM card history and reverse nibbling method

Some data on a SIM card is stored using a reverse nibbling method in which a byte (8 bits) consists of two nibbles (both 4 bits long) and is saved in an inverted form shown on a picture below. This method is most commonly encountered in the GSM-network related records storage, such as IMSI, IMEI or ICCID. The picture below shows an ICCID number saved in a reverse nibble order.



```
0000    98 64 80 34 09 06 10 82 53 26        d 4 ─┼ 3&
```

Figure 22: ICCID number is saved in a reverse nibble order

The saved ICCID number is:

98 64 80 34 09 06 10 82 53 26

But, as mentioned earlier, first two digits of ICCID number for telecommunications are 89. Hence, when reversed, the proper ICCID number is:

89 46 08 43 90 60 01 28 35 62

4.3.1.2 Location history

A device's location can also be established, to some extent and detail, without GPS. A LAC (Location Area Code) is a 2-bytes unique code assigned to the Location Area (LA) i.e. geographical division of a mobile network. Location areas, on the other hand, are split into cells (hence cellular network) distinguished by a Cell ID (CID) number. A SIM card of a mobile device is used to store a Location Area Identifier which consists of MCC (Mobile Country Code), MNC (Mobile Network Code) and LAC. The LAI number is used to update the core of the mobile network (i.e. VLR – Visitor Location Register and MSC – Mobile Switching Centre) about the subscriber device's location to properly route and establish calls.

Each mobile operator determines the division of the country into areas of LA and LAC numbers assigned to them. There are websites[64] on the Internet, allowing a search for cellular base stations for GSM and UMTS based on the actual location, LAC and CID.

4.3.1.3 Active key-logger

Android and iOS devices both have a cache memory (i.e. a database) containing phrases of text entered by their users. This text is cached for the purpose of predictive text function. Often the text is saved into the cache in the same order as it was typed, therefore enabling investigators to piece together phrases or sentences. But since users can switch between the applications, the cache may contain data from all of them without actual information of the source application. Hence, investigators should be wary of misinterpretation of the cache content.

4.3.1.4 7-bit strings as a GSM alphabet

The GSM alphabet[65] is in the standard encoding uses 7-bits (which is different from widely used 8-bit encoding e.g. UTF-8). Using XACT[66] tool, 7-bit strings can be easily decoded.

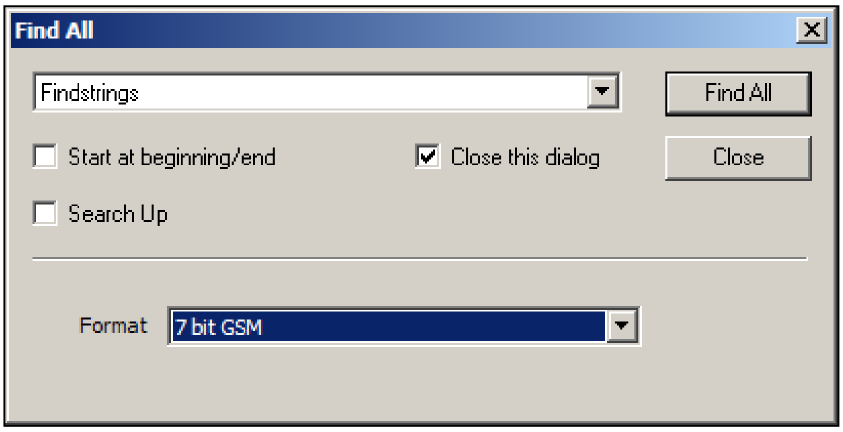
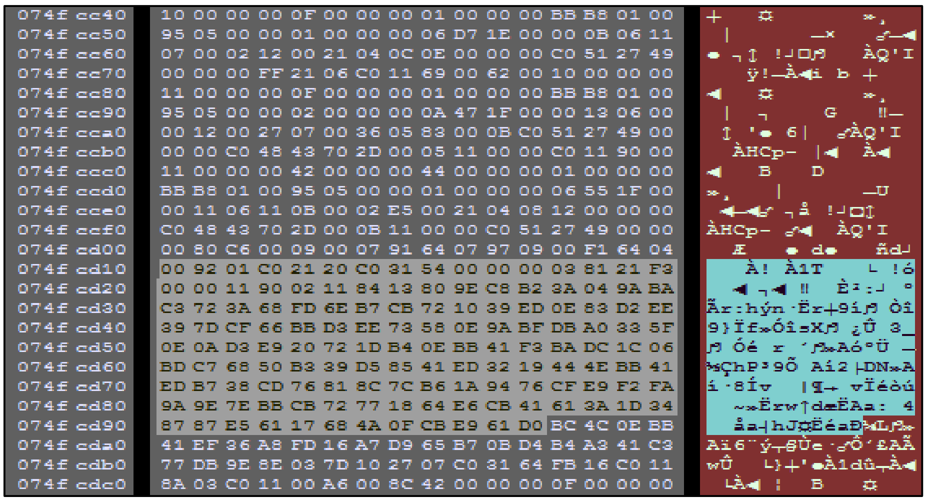

Find All
Findstrings
Start at beginning/end: [ ]
Close this dialog: [✓]
Search Up: [ ]
Format: 7 bit GSM

Figure 23: Searching for 7-bit encoded strings

[64] OpenCellID is the world's largest collaborative community project that collects GPS positions of cell towers, used free of charge, for a multitude of commercial and private purposes., http://opencellid.org/, last accessed on: 2015-09-14
[65] The 7 bit default alphabet, http://holman.id.au/apps/ipsms/default_alphabet.html, last accessed on: 2015-09-14
[66] XACT: Mobile phone investigations go deeper, http://www.veille.ma/IMG/pdf/xact-datasheet.pdf, last accessed on: 2015-09-14

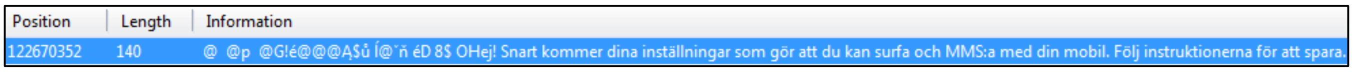
Original content in a hexadecimal format is shown on a picture below:



```
074f cc40  10 00 00 00 0F 00 00 00 01 00 00 00 BB B8 01 00
074f cc50  95 05 00 00 01 00 00 00 06 D7 1E 00 00 0B 06 11
074f cc60  07 00 02 12 00 21 04 0C 0E 00 00 00 C0 51 27 49
074f cc70  00 00 00 FF 21 06 C0 11 69 00 62 00 10 00 00 00
074f cc80  11 00 00 00 0F 00 00 00 01 00 00 00 BB B8 01 00
074f cc90  95 05 00 00 02 00 00 00 0A 47 1F 00 00 13 06 00
074f cca0  00 12 00 27 07 00 36 05 83 00 0B C0 51 27 49 00
074f ccb0  00 00 C0 48 43 70 2D 00 05 11 00 00 C0 11 90 00
074f ccc0  11 00 00 00 42 00 00 00 44 00 00 00 01 00 00 00
074f ccd0  BB B8 01 00 95 05 00 00 01 00 00 00 06 55 1F 00
074f cce0  00 11 06 11 0B 00 02 E5 00 21 04 08 12 00 00 00
074f ccf0  C0 48 43 70 2D 00 0B 11 00 00 C0 51 27 49 00 00
074f cd00  00 80 C6 00 09 00 07 91 64 07 97 09 00 F1 64 04
074f cd10  00 92 01 C0 21 20 C0 31 54 00 00 00 03 81 21 F3
074f cd20  00 00 11 90 02 11 84 13 80 9E C8 B2 3A 04 9A BA
074f cd30  C3 72 3A 68 FD 6E B7 CB 72 10 39 ED 0E 83 D2 EE
074f cd40  39 7D CF 66 BB D3 EE 73 58 0E 9A BF DB A0 33 5F
074f cd50  0E 0A D3 E9 20 72 1D B4 0E BB 41 F3 BA DC 1C 06
074f cd60  BD C7 68 50 B3 39 D5 85 41 ED 32 19 44 4E BB 41
074f cd70  ED B7 38 CD 76 81 8C 7C B6 1A 94 76 CF E9 F2 FA
074f cd80  9A 9E 7E BB CB 72 77 18 64 E6 CB 41 61 3A 1D 34
074f cd90  87 87 E5 61 17 68 4A 0F CB E9 61 D0 BC 4C 0E BB
074f cda0  41 EF 36 A8 FD 16 A7 D9 65 B7 0B D4 B4 A3 41 C3
074f cdb0  77 DB 9E 8E 03 7D 10 27 07 C0 31 64 FB 16 C0 11
074f cdc0  8A 03 C0 11 00 A6 00 8C 42 00 00 00 0F 00 00 00
```

Figure 24: Hexadecimal representation of 7-bit encoded strings

After decoding, a human-readable content is visible:



Position	Length	Information
122670352	140	@ @p @G!é@@@Ą$ů Í@˘ň éD 8$ OHej! Snart kommer dina inställningar som gör att du kan surfa och MMS:a med din mobil. Följ instruktionerna för att spara.

Figure 25: Human-readable string after decoding

4.3.2 Tools and devices for physical forensics

There are a number of dedicated tools and hardware devices aimed at providing physical access to investigated mobile devices.

4.3.2.1 Mobile Unlock Clips

Some mobile unlock clips allow investigators to get user passwords from many smartphones. The clips do not require resetting the devices which in many cases revert them to factory settings. Unlock clips keep device's content and allows to get the passcode easily. Examples of such devices are:

- MFC Dongle[67]
- XPIN Clip[68]
- Genie Universal 2010 Clip[69]

4.3.2.2 Service boxes

A service box is a device enabling an advanced access to mobile phones and their maintenance e.g. to disable SIM-locks, flash a dead phone, reconstruct certificates etc. But many boxes also can download the whole memory of a memory chip, thus performing its full physical extraction to a raw data format. Examples of such devices are:

- Octopus Box[70]

[67] MFC Dongle, http://www.mfcbox.com/, last accessed on: 2015-10-19
[68] XPIN Clip, http://xpinclip.com/, last accessed on: 2015-10-19
[69] Genie Universal 2010 Clip, http://multi-com.eu/,details,id_pr,6998,key,genie-universal-2010-clip.html, last accessed on: 2015-10-19
[70] Octopus Box, http://octopusbox.com/, last accessed on: 2015-10-19

- RiFF Box[71]
- Pegasus Box[72]

4.3.3 JTAG as a backup interface for physical forensics

In many cases, physical extraction is only possible by the use of a JTAG interface. According to Wikipedia (http://en.wikipedia.org/wiki/Joint_Test_Action_Group):

Joint Test Action Group (JTAG) is the common name for what was later standardised as the IEEE 1149.1 Standard Test Access Port and Boundary-Scan Architecture. It was initially devised for testing printed circuit boards using a boundary scan and is still widely used for this application. Today JTAG is also widely used for IC debug ports. In the embedded processor market, essentially all modern processors support JTAG when they have enough pins. Embedded systems development relies on debuggers talking to chips with JTAG to perform operations like single stepping and breakpointing. Digital electronics products such as cell phones or a wireless access point generally have no other debug or test interfaces.

JTAG forensics is an acquisition procedure which involves connecting to the Standard Test Access Port (TAPs) on a device and instructing the processor to transfer the raw data stored on connected memory chips. Jtagging supported phones can be an extremely effective technique to extract a full physical image from devices that cannot be acquired by other means. JTAG in conjunction with the appropriate service box gives you very powerful tool to extract physical image of the device's memory.

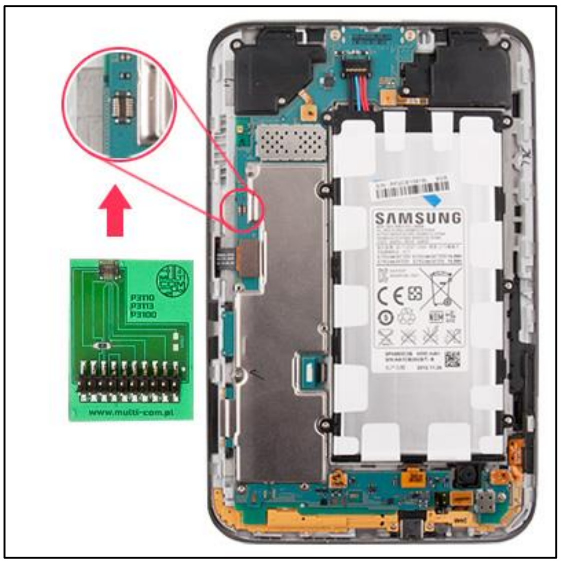


Figure 26: A JTAG connector. Source: http://multi-com.eu/multicom/pimg/jtag_p3100_400.jpg

4.4 Logical analysis

4.4.1 Android partitions

Android is the operating system based on Linux for mobile devices such as smartphones, tablets and netbooks. Currently, it is the most popular mobile system in the world. Android, although it is not a typical Linux distribution, is a Linux system, and was based on the Linux kernel.

[71] RiFF Box, http://www.riffbox.org/, last accessed on: 2015-10-19
[72] Pegasus Box, http://pegasusbox.com/, last accessed on: 2015-10-19

Android uses a number of different partitions and folders, and each of them having a separate function. The system partition contains the following directories:

- `/boot`

As the name suggests, this partition is associated with booting the device. This is where you store the kernel and bootloader, which is responsible for the proper running of smartphone / tablet. Without this partition device is not able to run the operating system.

- `/system`

In the /system directory, the operating system is stored, except for the kernel and bootloader. There are interface binaries and libraries, default system applications and many other things directly related to the Android operating system.

- `/recovery`

Partition /recovery is an alternative to the /boot partition. This partition is an additional software for service repairs and maintenance purposes.

- `/data`

Partition /data (or /userdata) is the partition, which collects user data such as contacts, text messages, personal settings and installed applications. Clearing this partition is the same as a reset to factory defaults.

- `/cache`

The built-in browser uses the cache memory for storage of saved passwords, searches, automatic fill-ins etc. Temporary settings and the results of the work that is currently doing by the user are also stored here.

- `/mics`

The contents of the partition /mics is system settings, operator data, configuration data, USB configuration etc.

Memory cards also have their own partitions and assigned directories:

- `/sdcard`

Partition /sdcard is a location of all data stored on the SD card. Here are the installed applications and user documents and multimedia which the user decided to save on the SD card. This partition can be unmounted (i.e. detached) from the operating system.

- `/sd-ext`

While this partition is not normally used in Android environment, it is worth mentioning. If the device is loaded with custom software, the /sd-ext partition works like a /data partition. This is often used as extension of built-in memory.

Android is an open system, which on the one hand, when we look at the resistance of device manufacturers and carriers to release updates in a timely fashion is a curse. On the other hand thanks to the enthusiasm of many people, openness is a big advantage. With a variety of modified software, multiple users can enjoy the faster system, newer and more powerful than those offered by the manufacturer. Some fairly well known Android modifications include:

- CyanogenMod[73]
- Android Open Kang Project[74]
- MIUI[75]

4.4.2 iOS partitions

All iPhones use a non-volatile NAND memory. It is divided into two partitions:

- `/dev/disk0s1s1 – mounted at /`

This partition is mounted at root (i.e. /) of the file system. Directories located on this partition remind us of a UNIX-like file system hierarchy.

- `/dev/disk0s1s2` (/dev/disk0s2 in older iOS versions) – mounted at `/private/var`

It is a device's user data partition which stores: App Store applications, all media files, settings etc.

It must be noted here, that jailbroken devices must have the partition configuration changed in /private/etc/fstab file, enabling read-write access. On iDevices that are not jailbroken, the entries look like:

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

On a jailbroken iDevice the entries are:

```
/dev/disk0s1s1 / hfs rw 0 1
/dev/disk0s1s2 /private/var hfs,nosuid,nodev rw 0 2
```

4.5 Task 4.1: Logical data extraction from Android devices

4.5.1 Introduction

In this task the students will use the AF Logical OSE tool to make a logical extraction from Android device. The trainer will give a short introduction to the usage of the Android AVD's and AF Logical OSE tool.

4.5.2 Tools used

- AVD[76]
- AF Logical OSE[77]

4.5.3 Details

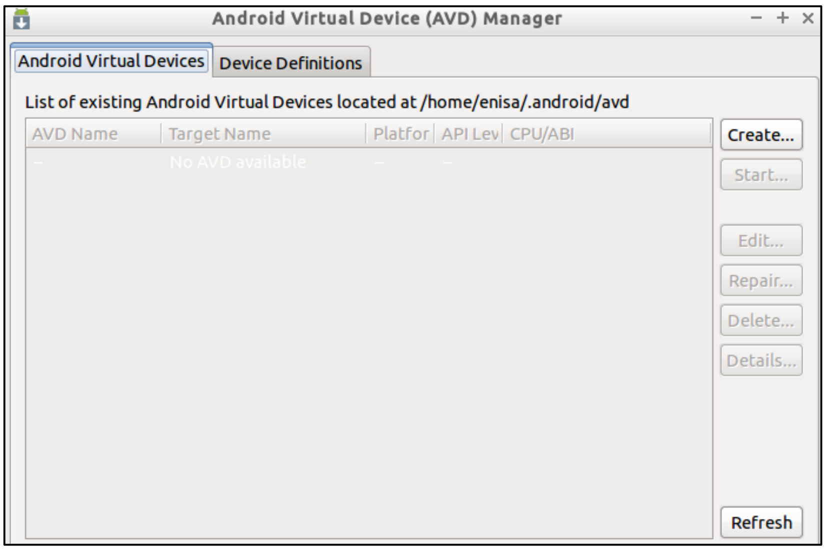
Students have to prepare the Android Virtual Machine with the Android AVD tool. After that, they'll have to fill some data into Android Virtual Machine and once that is done, students can make a logical extraction. If they are any problems with creation of Virtual Machine and / or populating it with sample data students can use AVD called *Android_VM_ENISA*.

4.5.4 Task walk-through

The following steps explain how to make a logical extraction of an Android device.

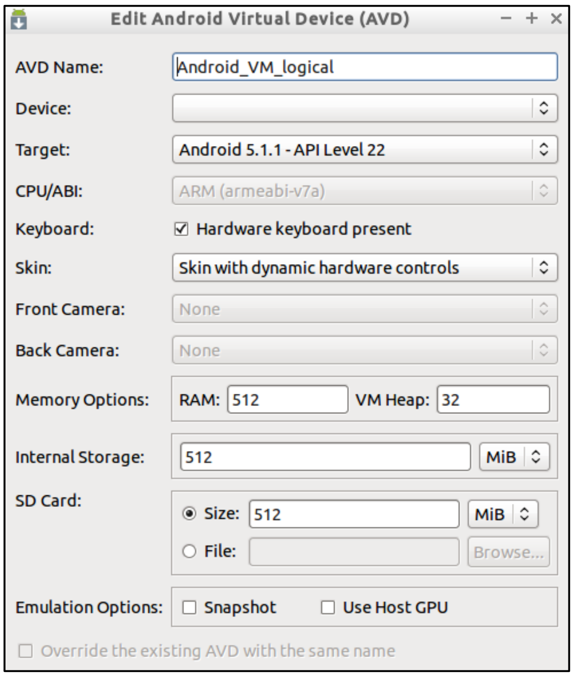
[73] CyanogenMod is an enhanced open source firmware distribution for smartphones and tablet computers based on the Android mobile operating system, http://www.cyanogenmod.org/, last accessed on: 2015-09-14
[74] Android Open Kang Project, http://aokp.co/, last accessed on: 2015-09-14
[75] MIUI ROM, http://en.miui.com/, last accessed on: 2015-09-14
[76] AVD, http://developer.android.com/tools/help/avd-manager.html, last accessed on: 2015-10-19
[77] AF Logical OSE, https://santoku-linux.com/howto/howto-use-aflogical-ose-logical-forensics-android/, last accessed on: 2015-10-19

4.5.4.1 Create new AVD machine. Open Linux Terminal and type android. You will see the SDK Manager window. Navigate to Tools and go to Manage AVDs. Click on Create and create a new AVD as shown in the picture below.



List of existing Android Virtual Devices located at /home/enisa/.android/avd

Figure 27 AVD Manager

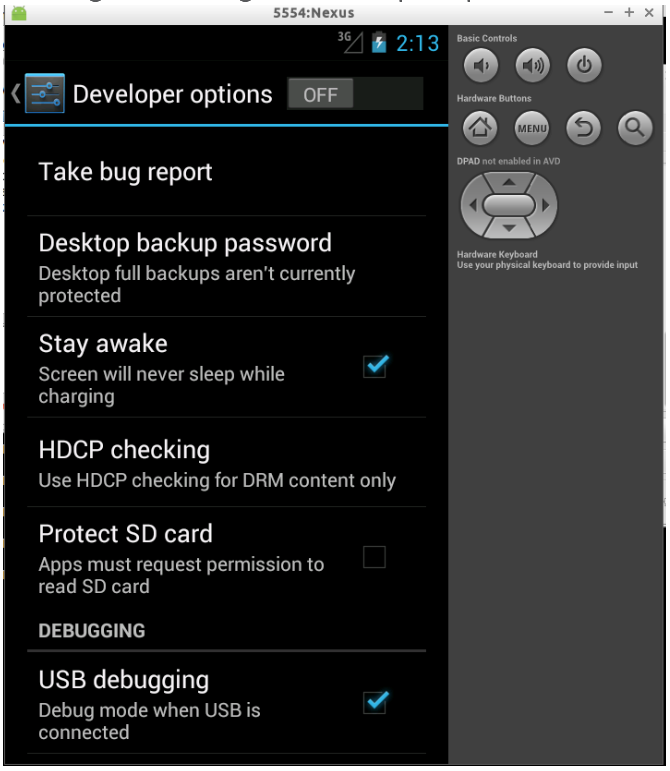

Edit Android Virtual Device (AVD)
Browse...
Override the existing AVD with the same name

AVD Name:	Android_VM_logical
Device:	
Target:	Android 5.1.1 - API Level 22
CPU/ABI:	ARM (armeabi-v7a)
Keyboard:	☑ Hardware keyboard present
Skin:	Skin with dynamic hardware controls
Front Camera:	None
Back Camera:	None
Memory Options:	RAM: 512 VM Heap: 32
Internal Storage:	512 MiB
SD Card:	● Size: 512 MiB; ○ File:
Emulation Options:	☐ Snapshot ☐ Use Host GPU

Figure 28 Create new AVD

4.5.4.2 Fill in sample data (e.g. add some contacts, try to send few SMS messages, try to call any number, open Internet browser, save some images on the internal memory and send some images by MMS message).

4.5.4.3 On Android Virtual Machine go to Settings -> Developer options and turn on USB debugging.



Take bug report

Desktop backup password
Desktop full backups aren't currently protected

Stay awake ✓
Screen will never sleep while charging

HDCP checking
Use HDCP checking for DRM content only

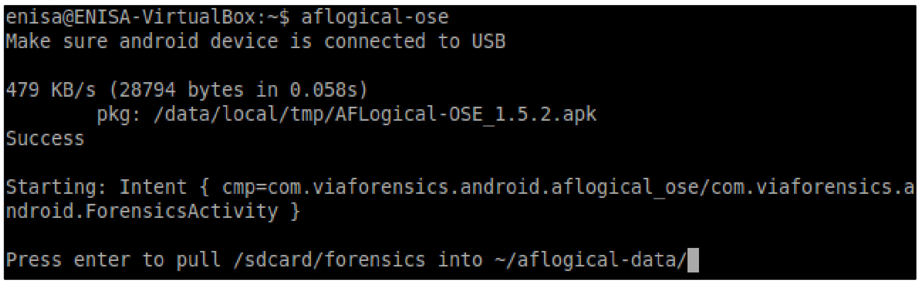
Protect SD card ☐
Apps must request permission to read SD card

DEBUGGING

USB debugging ✓
Debug mode when USB is connected

Figure 29 Enable USB debugging

4.5.4.4 Run the aflogical-ose command via terminal. By running this command you will push to the device a small application which tries to download data from the device.



```
enisa@ENISA-VirtualBox:~$ aflogical-ose
Make sure android device is connected to USB

479 KB/s (28794 bytes in 0.058s)
        pkg: /data/local/tmp/AFLogical-OSE_1.5.2.apk
Success

Starting: Intent { cmp=com.viaforensics.android.aflogical_ose/com.viaforensics.a
ndroid.ForensicsActivity }

Press enter to pull /sdcard/forensics into ~/aflogical-data/
```

Figure 30 aflogical-ose command

4.5.4.5 On the device's screen it can be seen types of information for downloading from the device memory. Click on capture and wait until you "Data extraction completed" message appears.

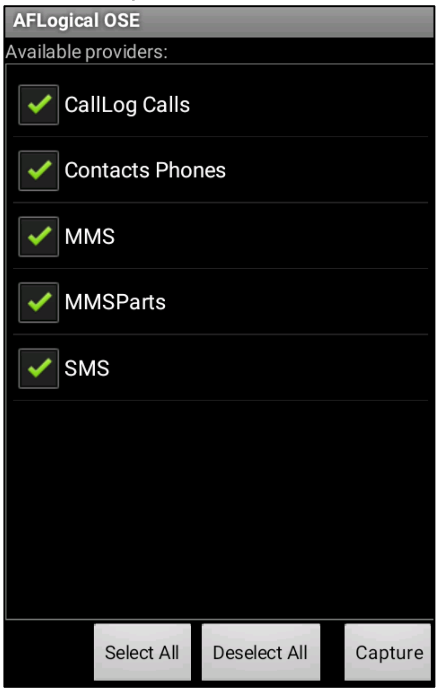
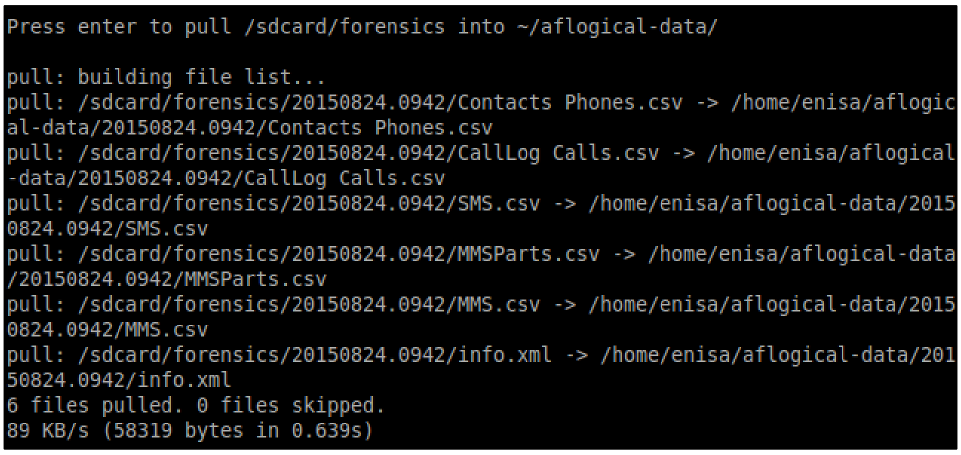


Figure 31

4.5.4.6 Now you need to get back into the terminal window and press enter to download data to the local hard disk.



```
Press enter to pull /sdcard/forensics into ~/aflogical-data/

pull: building file list...
pull: /sdcard/forensics/20150824.0942/Contacts Phones.csv -> /home/enisa/aflogic
al-data/20150824.0942/Contacts Phones.csv
pull: /sdcard/forensics/20150824.0942/CallLog Calls.csv -> /home/enisa/aflogical
-data/20150824.0942/CallLog Calls.csv
pull: /sdcard/forensics/20150824.0942/SMS.csv -> /home/enisa/aflogical-data/2015
0824.0942/SMS.csv
pull: /sdcard/forensics/20150824.0942/MMSParts.csv -> /home/enisa/aflogical-data
/20150824.0942/MMSParts.csv
pull: /sdcard/forensics/20150824.0942/MMS.csv -> /home/enisa/aflogical-data/2015
0824.0942/MMS.csv
pull: /sdcard/forensics/20150824.0942/info.xml -> /home/enisa/aflogical-data/201
50824.0942/info.xml
6 files pulled. 0 files skipped.
89 KB/s (58319 bytes in 0.639s)
```

Figure 32 aflogical-ose pull data from device

4.5.4.7 After that you have to find a folder called aflogical-data on your hard drive. In this folder you will find another folder named by the date and time of the extraction and when you will go deeper then you will find *.csv files with resources downloaded from the phone.

4.6 Task 4.2: File system extraction from Android devices

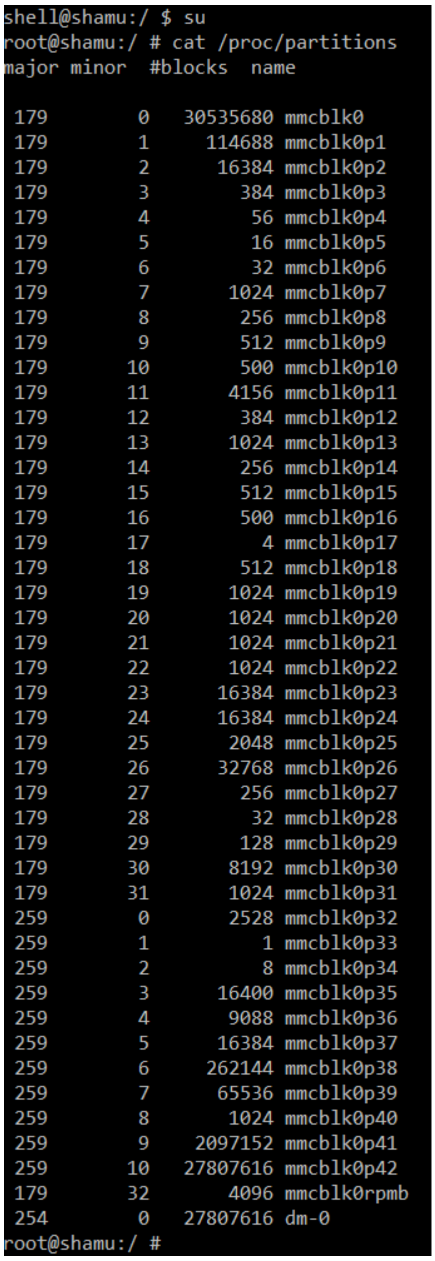
4.6.1 Introduction

This task demonstrates how to extract the file system from an actual device. The device used is an unlocked-rooted LG Nexus 6 (shamu).

4.6.2 Task walk-through

4.6.2.1 Firstly we connect the device through usb and enable usb debugging in the phone settings.

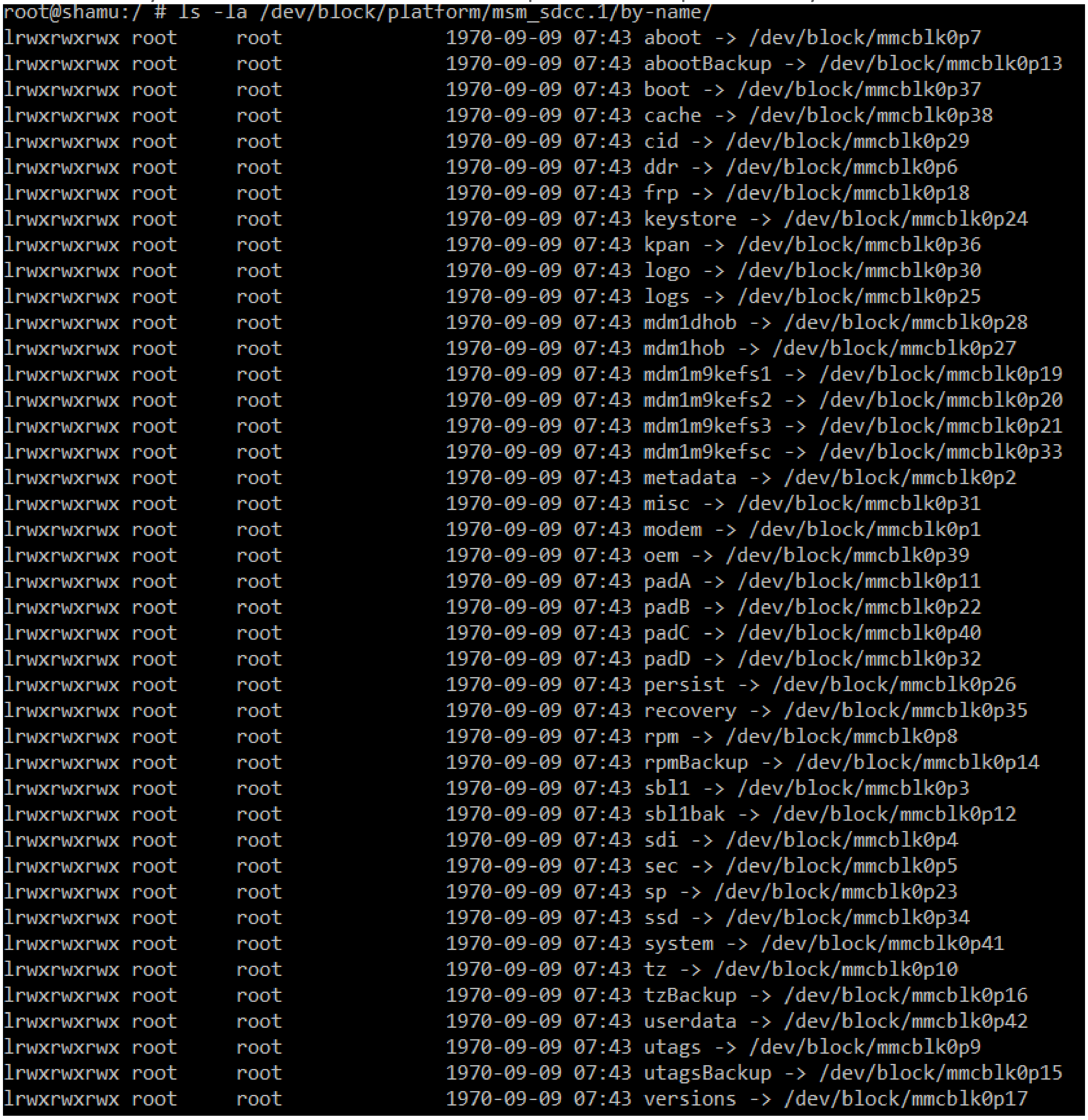
4.6.2.2 To identify the partition layout we connect to the device through adb shell and list partitions through /proc/partitions.



```
shell@shamu:/ $ su
root@shamu:/ # cat /proc/partitions
major minor  #blocks  name

 179        0   30535680 mmcblk0
 179        1     114688 mmcblk0p1
 179        2      16384 mmcblk0p2
 179        3        384 mmcblk0p3
 179        4         56 mmcblk0p4
 179        5         16 mmcblk0p5
 179        6         32 mmcblk0p6
 179        7       1024 mmcblk0p7
 179        8        256 mmcblk0p8
 179        9        512 mmcblk0p9
 179       10        500 mmcblk0p10
 179       11       4156 mmcblk0p11
 179       12        384 mmcblk0p12
 179       13       1024 mmcblk0p13
 179       14        256 mmcblk0p14
 179       15        512 mmcblk0p15
 179       16        500 mmcblk0p16
 179       17          4 mmcblk0p17
 179       18        512 mmcblk0p18
 179       19       1024 mmcblk0p19
 179       20       1024 mmcblk0p20
 179       21       1024 mmcblk0p21
 179       22       1024 mmcblk0p22
 179       23      16384 mmcblk0p23
 179       24      16384 mmcblk0p24
 179       25       2048 mmcblk0p25
 179       26      32768 mmcblk0p26
 179       27        256 mmcblk0p27
 179       28         32 mmcblk0p28
 179       29        128 mmcblk0p29
 179       30       8192 mmcblk0p30
 179       31       1024 mmcblk0p31
 259        0       2528 mmcblk0p32
 259        1          1 mmcblk0p33
 259        2          8 mmcblk0p34
 259        3      16400 mmcblk0p35
 259        4       9088 mmcblk0p36
 259        5      16384 mmcblk0p37
 259        6     262144 mmcblk0p38
 259        7      65536 mmcblk0p39
 259        8       1024 mmcblk0p40
 259        9    2097152 mmcblk0p41
 259       10   27807616 mmcblk0p42
 179       32       4096 mmcblk0rpmb
 254        0   27807616 dm-0
root@shamu:/ #
```

Figure 33 /proc/partitions

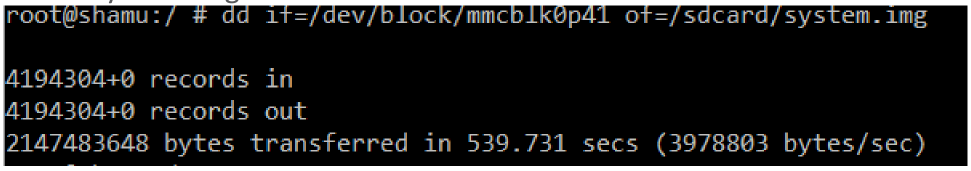
4.6.2.3 Alternatively to better understand the mount points we list partitions by name.



```
root@shamu:/ # ls -la /dev/block/platform/msm_sdcc.1/by-name/
lrwxrwxrwx root     root                 1970-09-09 07:43 aboot -> /dev/block/mmcblk0p7
lrwxrwxrwx root     root                 1970-09-09 07:43 abootBackup -> /dev/block/mmcblk0p13
lrwxrwxrwx root     root                 1970-09-09 07:43 boot -> /dev/block/mmcblk0p37
lrwxrwxrwx root     root                 1970-09-09 07:43 cache -> /dev/block/mmcblk0p38
lrwxrwxrwx root     root                 1970-09-09 07:43 cid -> /dev/block/mmcblk0p29
lrwxrwxrwx root     root                 1970-09-09 07:43 ddr -> /dev/block/mmcblk0p6
lrwxrwxrwx root     root                 1970-09-09 07:43 frp -> /dev/block/mmcblk0p18
lrwxrwxrwx root     root                 1970-09-09 07:43 keystore -> /dev/block/mmcblk0p24
lrwxrwxrwx root     root                 1970-09-09 07:43 kpan -> /dev/block/mmcblk0p36
lrwxrwxrwx root     root                 1970-09-09 07:43 logo -> /dev/block/mmcblk0p30
lrwxrwxrwx root     root                 1970-09-09 07:43 logs -> /dev/block/mmcblk0p25
lrwxrwxrwx root     root                 1970-09-09 07:43 mdm1dhob -> /dev/block/mmcblk0p28
lrwxrwxrwx root     root                 1970-09-09 07:43 mdm1hob -> /dev/block/mmcblk0p27
lrwxrwxrwx root     root                 1970-09-09 07:43 mdm1m9kefs1 -> /dev/block/mmcblk0p19
lrwxrwxrwx root     root                 1970-09-09 07:43 mdm1m9kefs2 -> /dev/block/mmcblk0p20
lrwxrwxrwx root     root                 1970-09-09 07:43 mdm1m9kefs3 -> /dev/block/mmcblk0p21
lrwxrwxrwx root     root                 1970-09-09 07:43 mdm1m9kefsc -> /dev/block/mmcblk0p33
lrwxrwxrwx root     root                 1970-09-09 07:43 metadata -> /dev/block/mmcblk0p2
lrwxrwxrwx root     root                 1970-09-09 07:43 misc -> /dev/block/mmcblk0p31
lrwxrwxrwx root     root                 1970-09-09 07:43 modem -> /dev/block/mmcblk0p1
lrwxrwxrwx root     root                 1970-09-09 07:43 oem -> /dev/block/mmcblk0p39
lrwxrwxrwx root     root                 1970-09-09 07:43 padA -> /dev/block/mmcblk0p11
lrwxrwxrwx root     root                 1970-09-09 07:43 padB -> /dev/block/mmcblk0p22
lrwxrwxrwx root     root                 1970-09-09 07:43 padC -> /dev/block/mmcblk0p40
lrwxrwxrwx root     root                 1970-09-09 07:43 padD -> /dev/block/mmcblk0p32
lrwxrwxrwx root     root                 1970-09-09 07:43 persist -> /dev/block/mmcblk0p26
lrwxrwxrwx root     root                 1970-09-09 07:43 recovery -> /dev/block/mmcblk0p35
lrwxrwxrwx root     root                 1970-09-09 07:43 rpm -> /dev/block/mmcblk0p8
lrwxrwxrwx root     root                 1970-09-09 07:43 rpmBackup -> /dev/block/mmcblk0p14
lrwxrwxrwx root     root                 1970-09-09 07:43 sbl1 -> /dev/block/mmcblk0p3
lrwxrwxrwx root     root                 1970-09-09 07:43 sbl1bak -> /dev/block/mmcblk0p12
lrwxrwxrwx root     root                 1970-09-09 07:43 sdi -> /dev/block/mmcblk0p4
lrwxrwxrwx root     root                 1970-09-09 07:43 sec -> /dev/block/mmcblk0p5
lrwxrwxrwx root     root                 1970-09-09 07:43 sp -> /dev/block/mmcblk0p23
lrwxrwxrwx root     root                 1970-09-09 07:43 ssd -> /dev/block/mmcblk0p34
lrwxrwxrwx root     root                 1970-09-09 07:43 system -> /dev/block/mmcblk0p41
lrwxrwxrwx root     root                 1970-09-09 07:43 tz -> /dev/block/mmcblk0p10
lrwxrwxrwx root     root                 1970-09-09 07:43 tzBackup -> /dev/block/mmcblk0p16
lrwxrwxrwx root     root                 1970-09-09 07:43 userdata -> /dev/block/mmcblk0p42
lrwxrwxrwx root     root                 1970-09-09 07:43 utags -> /dev/block/mmcblk0p9
lrwxrwxrwx root     root                 1970-09-09 07:43 utagsBackup -> /dev/block/mmcblk0p15
lrwxrwxrwx root     root                 1970-09-09 07:43 versions -> /dev/block/mmcblk0p17
```

Figure 34 Partitions By Name

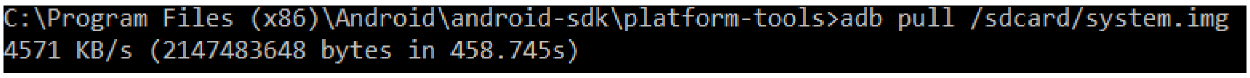
4.6.2.4 Next, extract the system.img to the root of the sdcard.



```
root@shamu:/ # dd if=/dev/block/mmcblk0p41 of=/sdcard/system.img

4194304+0 records in
4194304+0 records out
2147483648 bytes transferred in 539.731 secs (3978803 bytes/sec)
```

Figure 35 Backup of system.img to file

4.6.2.5 Finally we pull the system.img locally to further investigate.



```
C:\Program Files (x86)\Android\android-sdk\platform-tools>adb pull /sdcard/system.img
4571 KB/s (2147483648 bytes in 458.745s)
```

Figure 36

4.7 Task 4.3: Manual file carving

4.7.1 Introduction

In this task, the students will use the wxHexEditor tool to perform file carving from a file system imaged of an Android device. The trainer will give a short introduction into the usage of the wxHexEditor and tells something about file signatures.

4.7.2 Tools used

- wxHexEditor[78]

4.7.3 Details

This exercise refers to Task 2 above. Now that students have dumped a partition from Android Virtual Machine, the goal is to find some JPG file in the partition image.

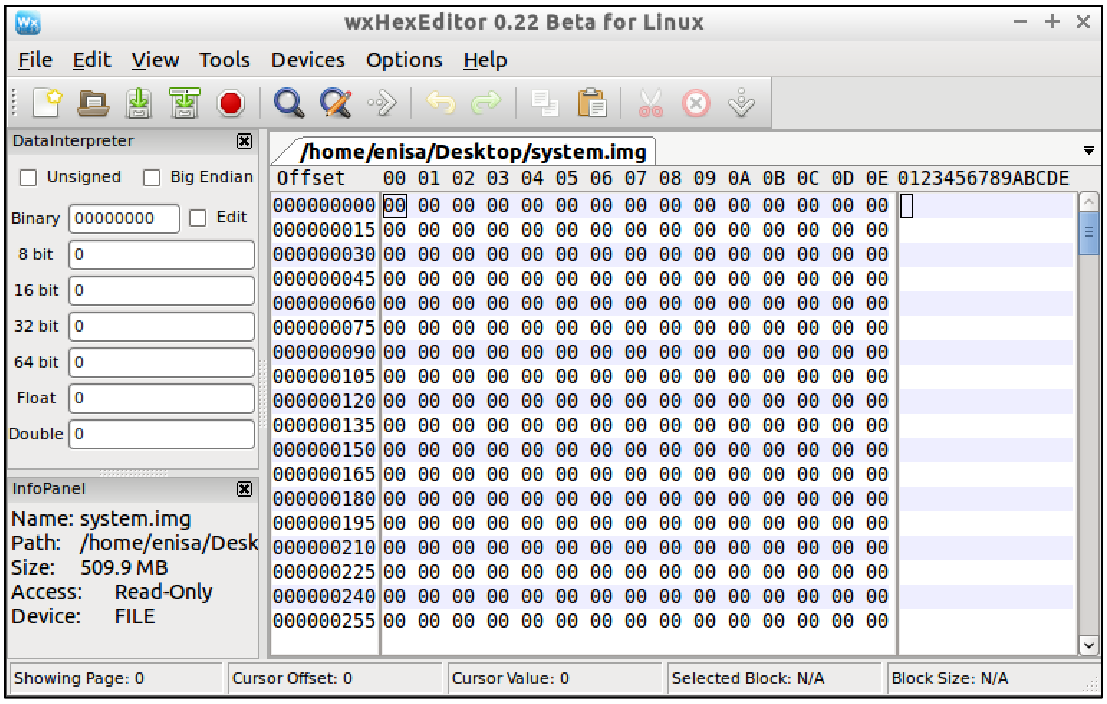
4.7.4 Task walk-through

A file signature is data used to identify or verify the content of a file. In particular, it may be a so called magic number which identifies the format of the file. Generally a short sequence of bytes (most magic numbers are 2–4 bytes long) is placed at the beginning of the file.

Manual file carving is the process of reconstructing files by scanning the raw image of the disk looking for file signatures and its contents, and reassembling them. This is usually done by examining the header (the first few bytes) and footer (the last few bytes) of a file.

[78] wxHexEditor, http://www.wxhexeditor.org/, last accessed on: 2015-10-19

4.7.4.1 Open wxHexEditor by typing in Linux Terminal command wxHexEditor and open one of partition file by clicking on File -> Open.


wxHexEditor 0.22 Beta for Linux
File Edit View Tools Devices Options Help
DataInterpreter
InfoPanel

Unsigned [ ] Big Endian [ ]
Binary 00000000 Edit [ ]
8 bit 0
16 bit 0
32 bit 0
64 bit 0
Float 0
Double 0

Name: system.img
Path: /home/enisa/Desk
Size: 509.9 MB
Access: Read-Only
Device: FILE

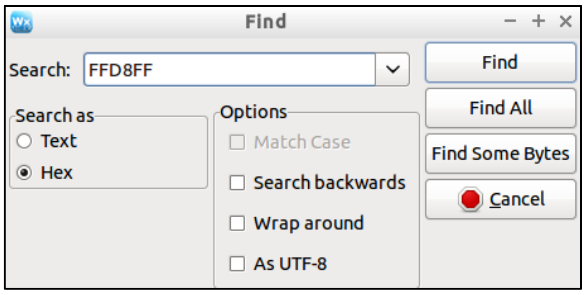
/home/enisa/Desktop/system.img

```
Offset    00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0123456789ABCDE
000000000 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000015 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000045 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000060 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000075 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000090 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000105 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000135 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000150 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000165 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000180 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000195 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000210 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000225 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000240 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
000000255 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Showing Page: 0 | Cursor Offset: 0 | Cursor Value: 0 | Selected Block: N/A | Block Size: N/A

Figure 37

4.7.4.2 You need to find a header of a JPG file which is FF D8 FF (list of all known file signatures can be found on the Internet[79]). It is important, that the header is a few bytes ahead of an ASCII string "JFIF". Use search tool ("Edit -> Find -> Find All") directly from wxHexEditor.



Search: FFD8FF

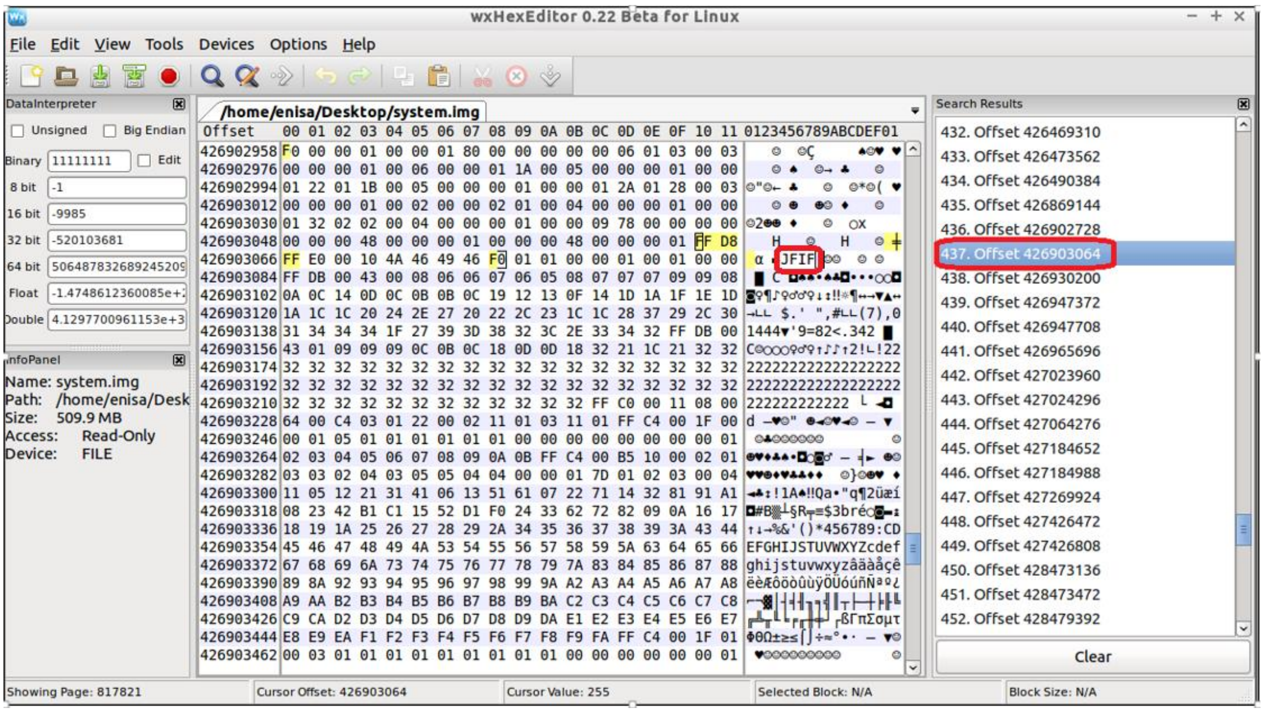
Search as: Text [✓ Hex]

Options: Match Case [ ], Search backwards [ ], Wrap around [ ], As UTF-8 [ ]

Figure 38

[79] File signatures table, http://www.garykessler.net/library/file_sigs.html, last accessed on: 2015-09-14

4.7.4.3 Save the offset address of a JPG file header.


wxHexEditor 0.22 Beta for Linux
File Edit View Tools Devices Options Help
DataInterpreter
InfoPanel
Search Results
Clear

/home/enisa/Desktop/system.img

```
Offset     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 0123456789ABCDEF01
426902958 F0 00 00 01 00 00 01 80 00 00 00 00 00 06 01 03 00 03
426902976 00 00 00 01 00 06 00 00 01 1A 00 05 00 00 00 01 00 00
426902994 01 22 01 1B 00 05 00 00 00 01 00 00 01 2A 01 28 00 03
426903012 00 00 00 01 00 02 00 00 02 01 00 04 00 00 00 01 00 00
426903030 01 32 02 02 00 04 00 00 00 01 00 00 09 78 00 00 00 00   2        x
426903048 00 00 00 48 00 00 00 01 00 00 00 48 00 00 00 01 FF D8    H       H
426903066 FF E0 00 10 4A 46 49 46 F0 01 01 00 00 01 00 01 00 00  α JFIF
426903084 FF DB 00 43 00 08 06 06 07 06 05 08 07 07 07 09 09 08   C
426903102 0A 0C 14 0D 0C 0B 0B 0C 19 12 13 0F 14 1D 1A 1F 1E 1D
426903120 1A 1C 1C 20 24 2E 27 20 22 2C 23 1C 1C 28 37 29 2C 30    $.' ",#  (7),0
426903138 31 34 34 34 1F 27 39 3D 38 32 3C 2E 33 34 32 FF DB 00 1444 '9=82<.342
426903156 43 01 09 09 09 0C 0B 0C 18 0D 0D 18 32 21 1C 21 32 32 C           2! !22
426903174 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 222222222222222222
426903192 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 32 222222222222222222
426903210 32 32 32 32 32 32 32 32 32 32 32 32 FF C0 00 11 08 00 222222222222
426903228 64 00 C4 03 01 22 00 02 11 01 03 11 01 FF C4 00 1F 00 d     "
426903246 00 01 05 01 01 01 01 01 01 00 00 00 00 00 00 00 00 01
426903264 02 03 04 05 06 07 08 09 0A 0B FF C4 00 B5 10 00 02 01
426903282 03 03 02 04 03 05 05 04 04 00 00 01 7D 01 02 03 00 04             }
426903300 11 05 12 21 31 41 06 13 51 61 07 22 71 14 32 81 91 A1   !1A  Qa "q 2üæí
426903318 08 23 42 B1 C1 15 52 D1 F0 24 33 62 72 82 09 0A 16 17  #B   §R  $3bré
426903336 18 19 1A 25 26 27 28 29 2A 34 35 36 37 38 39 3A 43 44    %&'()*456789:CD
426903354 45 46 47 48 49 4A 53 54 55 56 57 58 59 5A 63 64 65 66 EFGHIJSTUVWXYZcdef
426903372 67 68 69 6A 73 74 75 76 77 78 79 7A 83 84 85 86 87 88 ghijstuvwxyzâäàåçê
426903390 89 8A 92 93 94 95 96 97 98 99 9A A2 A3 A4 A5 A6 A7 A8 ëèÆôöòûùÿÖÜóúñÑªº¿
426903408 A9 AA B2 B3 B4 B5 B6 B7 B8 B9 BA C2 C3 C4 C5 C6 C7 C8
426903426 C9 CA D2 D3 D4 D5 D6 D7 D8 D9 DA E1 E2 E3 E4 E5 E6 E7             ßΓπΣσµτ
426903444 E8 E9 EA F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FF C4 00 1F 01 ΦΘΩ±≥≤⌠⌡÷≈°•·
426903462 00 03 01 01 01 01 01 01 01 01 01 00 00 00 00 00 00 01
```

DataInterpreter: ☐ Unsigned ☐ Big Endian
Binary 11111111 ☐ Edit
8 bit -1
16 bit -9985
32 bit -520103681
64 bit 506487832689245209
Float -1.4748612360085e+
Double 4.1297700961153e+3

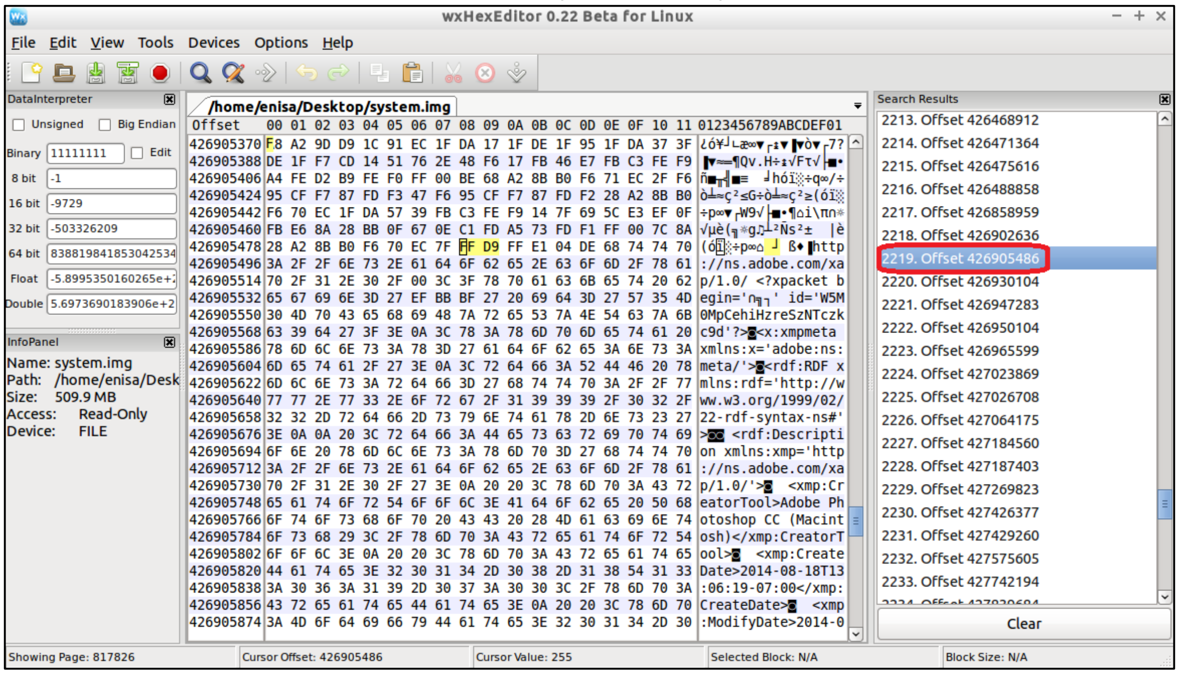
InfoPanel:
Name: system.img
Path: /home/enisa/Desk
Size: 509.9 MB
Access: Read-Only
Device: FILE

Search Results:
432. Offset 426469310
433. Offset 426473562
434. Offset 426490384
435. Offset 426869144
436. Offset 426902728
437. Offset 426903064
438. Offset 426930200
439. Offset 426947372
440. Offset 426947708
441. Offset 426965696
442. Offset 427023960
443. Offset 427024296
444. Offset 427064276
445. Offset 427184652
446. Offset 427184988
447. Offset 427269924
448. Offset 427426472
449. Offset 427426808
450. Offset 428473136
451. Offset 428473472
452. Offset 428479392

Showing Page: 817821 | Cursor Offset: 426903064 | Cursor Value: 255 | Selected Block: N/A | Block Size: N/A

Figure 39

4.7.4.4 Now you need to find a hex value of FF D9 which is a JPEG files' footer. Try to locate the nearest occurrence after the header identified in the previous step.


wxHexEditor 0.22 Beta for Linux
File Edit View Tools Devices Options Help
DataInterpreter
InfoPanel
Search Results
Clear

/home/enisa/Desktop/system.img

```
Offset     00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 0123456789ABCDEF01
426905370 F8 A2 9D D9 1C 91 EC 1F DA 17 1F DE 1F 95 1F DA 37 3F ¿ó¥    æ∞       ò   7?
426905388 DE 1F F7 CD 14 51 76 2E 48 F6 17 FB 46 E7 FB C3 FE F9    ≈ ¶Qv.H÷ √Fτ√
426905406 A4 FE D2 B9 FE F0 FF 00 BE 68 A2 8B B0 F6 71 EC 2F F6 ñ          hóï ÷q∞/÷
426905424 95 CF F7 87 FD F3 47 F6 95 CF F7 87 FD F2 28 A2 8B B0 ò ≈ç²≤G÷ò ≈ç²≥(óï
426905442 F6 70 EC 1F DA 57 39 FB C3 FE F9 14 7F 69 5C E3 EF 0F ÷p∞  W9√    ¶ i\πn
426905460 FB E6 8A 28 BB 0F 67 0E C1 FD A5 73 FD F1 FF 00 7C 8A √µè(   g   ²Ñs²±  |è
426905478 28 A2 8B B0 F6 70 EC 7F FF D9 FF E1 04 DE 68 74 74 70 (óï ÷p∞      ß  http
426905496 3A 2F 2F 6E 73 2E 61 64 6F 62 65 2E 63 6F 6D 2F 78 61 ://ns.adobe.com/xa
426905514 70 2F 31 2E 30 2F 00 3C 3F 78 70 61 63 6B 65 74 20 62 p/1.0/ <?xpacket b
426905532 65 67 69 6E 3D 27 EF BB BF 27 20 69 64 3D 27 57 35 4D egin='∩  ' id='W5M
426905550 30 4D 70 43 65 68 69 48 7A 72 65 53 7A 4E 54 63 7A 6B 0MpCehiHzreSzNTczk
426905568 63 39 64 27 3F 3E 0A 3C 78 3A 78 6D 70 6D 65 74 61 20 c9d'?> <x:xmpmeta
426905586 78 6D 6C 6E 73 3A 78 3D 27 61 64 6F 62 65 3A 6E 73 3A xmlns:x='adobe:ns:
426905604 6D 65 74 61 2F 27 3E 0A 3C 72 64 66 3A 52 44 46 20 78 meta/'> <rdf:RDF x
426905622 6D 6C 6E 73 3A 72 64 66 3D 27 68 74 74 70 3A 2F 2F 77 mlns:rdf='http://w
426905640 77 77 2E 77 33 2E 6F 72 67 2F 31 39 39 39 2F 30 32 2F ww.w3.org/1999/02/
426905658 32 32 2D 72 64 66 2D 73 79 6E 74 61 78 2D 6E 73 23 27 22-rdf-syntax-ns#'
426905676 3E 0A 0A 20 3C 72 64 66 3A 44 65 73 63 72 69 70 74 69 >   <rdf:Descripti
426905694 6F 6E 20 78 6D 6C 6E 73 3A 78 6D 70 3D 27 68 74 74 70 on xmlns:xmp='http
426905712 3A 2F 2F 6E 73 2E 61 64 6F 62 65 2E 63 6F 6D 2F 78 61 ://ns.adobe.com/xa
426905730 70 2F 31 2E 30 2F 27 3E 0A 20 20 3C 78 6D 70 3A 43 72 p/1.0/'>   <xmp:Cr
426905748 65 61 74 6F 72 54 6F 6F 6C 3E 41 64 6F 62 65 20 50 68 eatorTool>Adobe Ph
426905766 6F 74 6F 73 68 6F 70 20 43 43 20 28 4D 61 63 69 6E 74 otoshop CC (Macint
426905784 6F 73 68 29 3C 2F 78 6D 70 3A 43 72 65 61 74 6F 72 54 osh)</xmp:CreatorT
426905802 6F 6F 6C 3E 0A 20 20 3C 78 6D 70 3A 43 72 65 61 74 65 ool>   <xmp:Create
426905820 44 61 74 65 3E 32 30 31 34 2D 30 38 2D 31 38 54 31 33 Date>2014-08-18T13
426905838 3A 30 36 3A 31 39 2D 30 37 3A 30 30 3C 2F 78 6D 70 3A :06:19-07:00</xmp:
426905856 43 72 65 61 74 65 44 61 74 65 3E 0A 20 20 3C 78 6D 70 CreateDate>   <xmp
426905874 3A 4D 6F 64 69 66 79 44 61 74 65 3E 32 30 31 34 2D 30 :ModifyDate>2014-0
```

DataInterpreter: ☐ Unsigned ☐ Big Endian
Binary 11111111 ☐ Edit
8 bit -1
16 bit -9729
32 bit -503326209
64 bit 838819841853042534
Float -5.8995350160265e+
Double 5.6973690183906e+2

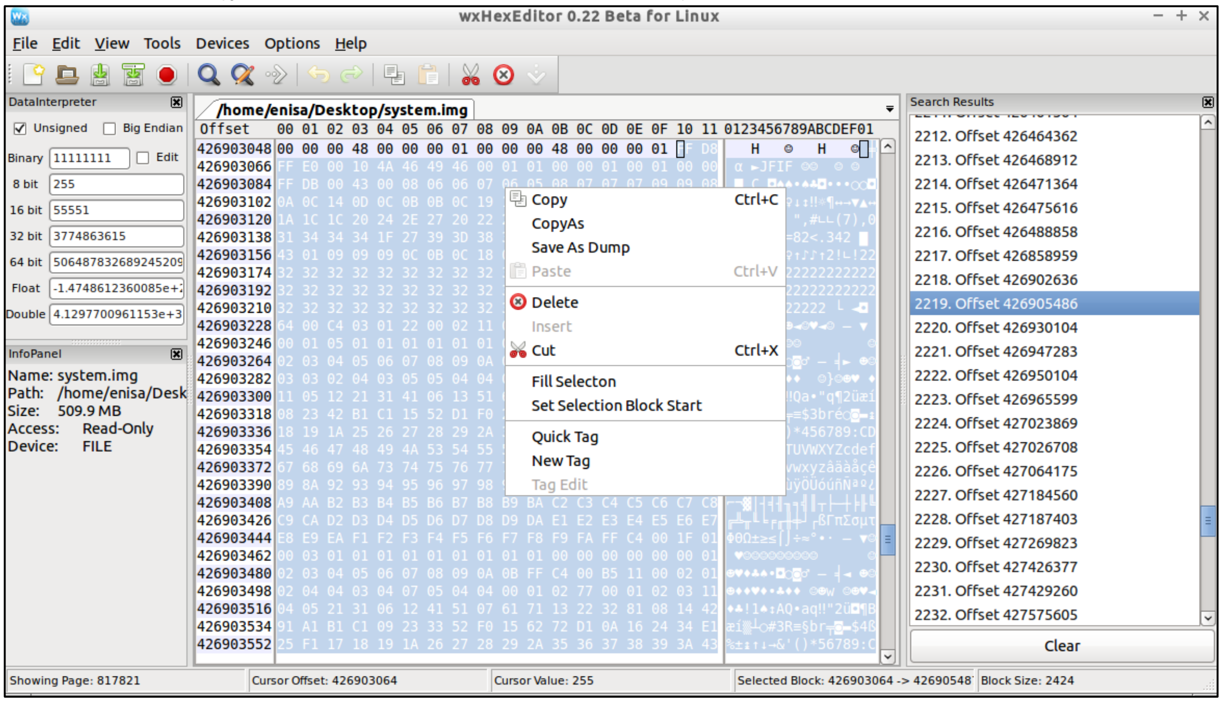
InfoPanel:
Name: system.img
Path: /home/enisa/Desk
Size: 509.9 MB
Access: Read-Only
Device: FILE

Search Results:
2213. Offset 426468912
2214. Offset 426471364
2215. Offset 426475616
2216. Offset 426488858
2217. Offset 426858959
2218. Offset 426902636
2219. Offset 426905486
2220. Offset 426930104
2221. Offset 426947283
2222. Offset 426950104
2223. Offset 426965599
2224. Offset 427023869
2225. Offset 427026708
2226. Offset 427064175
2227. Offset 427184560
2228. Offset 427187403
2229. Offset 427269823
2230. Offset 427426377
2231. Offset 427429260
2232. Offset 427575605
2233. Offset 427742194

Showing Page: 817826 | Cursor Offset: 426905486 | Cursor Value: 255 | Selected Block: N/A | Block Size: N/A

Figure 40

4.7.4.5 When you find the header and trailer of JPG file you have to mark hexadecimal code from header to footer (you need to know offset addresses).



```
Binary  11111111
8 bit   255
16 bit  55551
32 bit  3774863615
64 bit  506487832689245209
Float   -1.4748612360085e+2
Double  4.1297700961153e+3

Name:    system.img
Path:    /home/enisa/Desk
Size:    509.9 MB
Access:  Read-Only
Device:  FILE
```

```
/home/enisa/Desktop/system.img
Offset    00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 0123456789ABCDEF01
426903048 00 00 00 48 00 00 00 01 00 00 00 48 00 00 00 01 FF D8
426903066 FF E0 00 10 4A 46 49 46 00 01 01 00 00 01 00 01 00 00
426903084 FF DB 00 43 00 08 06 06 07 06 05 08 07 07 07 09 09 08
426903102 0A 0C 14 0D 0C 0B 0B 0C 19
426903120 1A 1C 1C 20 24 2E 27 20 22
426903138 31 34 34 34 1F 27 39 3D 38
426903156 43 01 09 09 09 0C 0B 0C 18
426903174 32 32 32 32 32 32 32 32 32
426903192 32 32 32 32 32 32 32 32 32
426903210 32 32 32 32 32 32 32 32 32
426903228 64 00 C4 03 01 22 00 02 11
426903246 00 01 05 01 01 01 01 01 01
426903264 02 03 04 05 06 07 08 09 0A
426903282 03 03 02 04 03 05 05 04 04
426903300 11 05 12 21 31 41 06 13 51
426903318 08 23 42 B1 C1 15 52 D1 F0
426903336 18 19 1A 25 26 27 28 29 2A
426903354 45 46 47 48 49 4A 53 54 55
426903372 67 68 69 6A 73 74 75 76 77
426903390 89 8A 92 93 94 95 96 97 98
426903408 A9 AA B2 B3 B4 B5 B6 B7 B8 B9 BA C2 C3 C4 C5 C6 C7 C8
426903426 C9 CA D2 D3 D4 D5 D6 D7 D8 D9 DA E1 E2 E3 E4 E5 E6 E7
426903444 E8 E9 EA F1 F2 F3 F4 F5 F6 F7 F8 F9 FA FF C4 00 1F 01
426903462 00 03 01 01 01 01 01 01 01 01 01 00 00 00 00 00 00 01
426903480 02 03 04 05 06 07 08 09 0A 0B FF C4 00 B5 11 00 02 01
426903498 02 04 04 03 04 07 05 04 04 00 01 02 77 00 01 02 03 11
426903516 04 05 21 31 06 12 41 51 07 61 71 13 22 32 81 08 14 42
426903534 91 A1 B1 C1 09 23 33 52 F0 15 62 72 D1 0A 16 24 34 E1
426903552 25 F1 17 18 19 1A 26 27 28 29 2A 35 36 37 38 39 3A 43
```

```
2212. Offset 426464362
2213. Offset 426468912
2214. Offset 426471364
2215. Offset 426475616
2216. Offset 426488858
2217. Offset 426858959
2218. Offset 426902636
2219. Offset 426905486
2220. Offset 426930104
2221. Offset 426947283
2222. Offset 426950104
2223. Offset 426965599
2224. Offset 427023869
2225. Offset 427026708
2226. Offset 427064175
2227. Offset 427184560
2228. Offset 427187403
2229. Offset 427269823
2230. Offset 427426377
2231. Offset 427429260
2232. Offset 427575605
```

```
Showing Page: 817821   Cursor Offset: 426903064   Cursor Value: 255   Selected Block: 426903064 -> 42690548   Block Size: 2424
```

Figure 41

4.7.4.6 Now you choose "Save As Dump" and save this file as TEST.JPG. After that you will be able to see the picture.

It is worth mentioning that file carving can be done automatically with commercial tools, such as Micro Systemation XRY or XACT,[80] or open-source ones, e.g. foremost.[81]

4.8 Task 4.4: RAM memory dump from Android device

4.8.1 Introduction

In this task the students will use tools to make a RAM memory extraction from Android device. After that students will use Volatility and Autopsy tools to analyse RAM dump file.

4.8.2 Tools used

- Android SDK[82]
- Android NDK[83]
- LiME[84]
- Dwarfdump[85]
- Volatility[86]

[80] Micro Systemation XRY/XACT, https://www.msab.com/, last accessed on: 2015-10-19
[81] Foremost, https://en.wikipedia.org/wiki/Foremost_%28software%29, last accessed on: 2015-10-19
[82] Android SDK, http://developer.android.com/sdk/index.html, last accessed on: 2015-09-14
[83] Android NDK, http://developer.android.com/tools/sdk/ndk/index.html, last accessed on: 2015-09-14
[84] LiME: Linux Memory Extractor, https://github.com/504ensicslabs/lime, last accessed on: 2015-09-14
[85] Libdwarf and Dwarfdump, http://wiki.dwarfstd.org/index.php?title=Libdwarf_And_Dwarfdump, last accessed on: 2015-09-14
[86] Volatility: RAM dump analyser, https://code.google.com/p/volatility/wiki/, last accessed on: 2015-09-14

- Autopsy[87]

4.8.3 Details

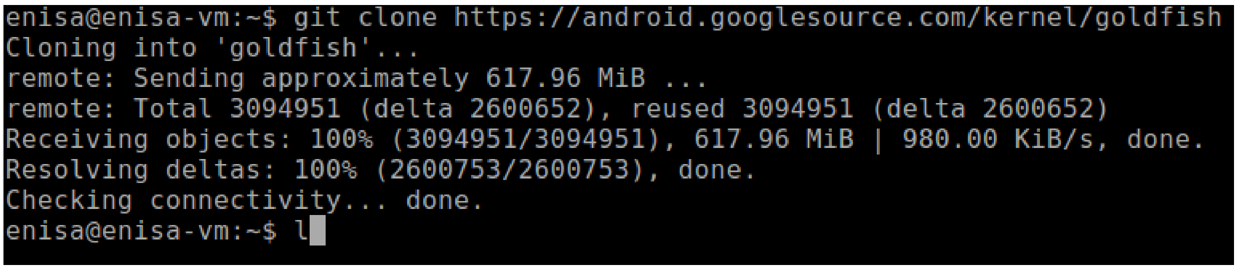
This exercise will explain how to:

- Perform Android device memory forensics with Volatility,
- Set up Android build environment,
- Cross-compile of Android kernel,
- Use the Android Emulator,
- Acquire memory from Android devices with LiME module,
- Build Volatility profile for Android,
- Run Volatility commands against Android memory dumps.

4.8.4 Task walk-through - Dumping RAM memory

4.8.4.1 Students will use the same emulator device they used in the previous task. Firstly will download the source code of the emulator kernel, named goldfish, from the official google repository

```
git clone https://android.googlesource.com/kernel/goldfish
```

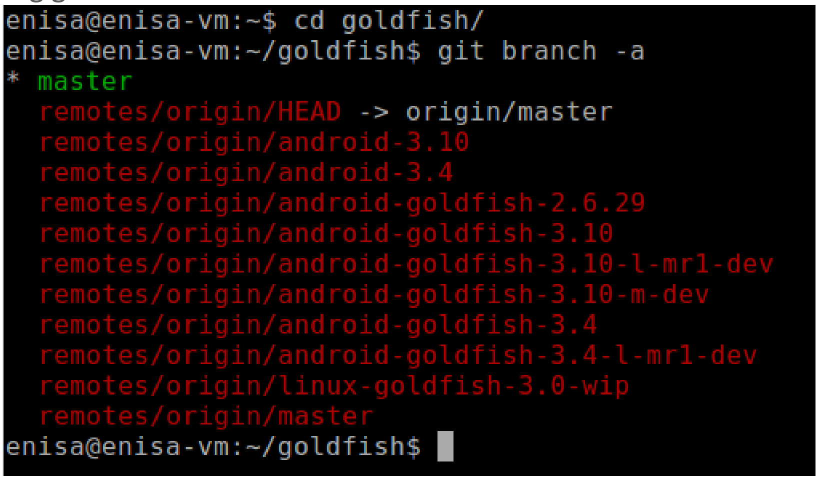


```
enisa@enisa-vm:~$ git clone https://android.googlesource.com/kernel/goldfish
Cloning into 'goldfish'...
remote: Sending approximately 617.96 MiB ...
remote: Total 3094951 (delta 2600652), reused 3094951 (delta 2600652)
Receiving objects: 100% (3094951/3094951), 617.96 MiB | 980.00 KiB/s, done.
Resolving deltas: 100% (2600753/2600753), done.
Checking connectivity... done.
enisa@enisa-vm:~$ l
```

Figure 42 Download of kernel source code

[87] Autopsy® is a digital forensics platform and graphical interface to The Sleuth Kit® and other digital forensics tools, http://www.sleuthkit.org/autopsy/, last accessed on: 2015-09-14

4.8.4.2 The android kernel has different versions that are split into branches. You can check the branches by issuing git branch –a inside the kernel source folder.

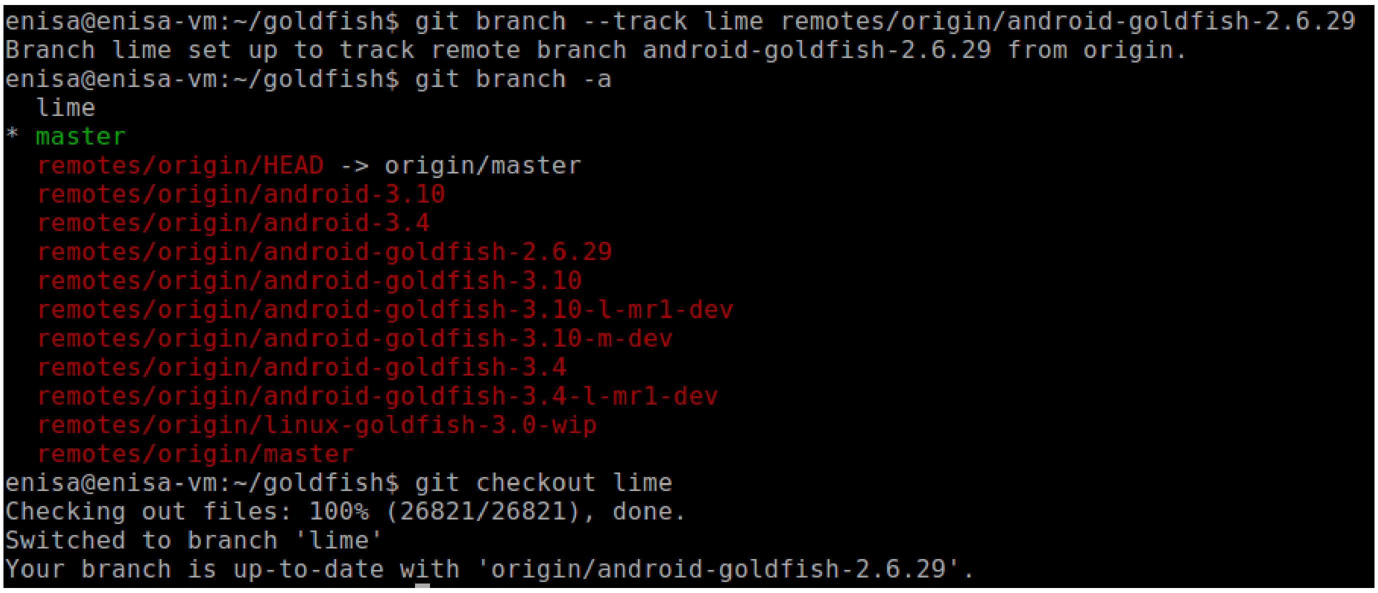


```
enisa@enisa-vm:~$ cd goldfish/
enisa@enisa-vm:~/goldfish$ git branch -a
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/android-3.10
  remotes/origin/android-3.4
  remotes/origin/android-goldfish-2.6.29
  remotes/origin/android-goldfish-3.10
  remotes/origin/android-goldfish-3.10-l-mr1-dev
  remotes/origin/android-goldfish-3.10-m-dev
  remotes/origin/android-goldfish-3.4
  remotes/origin/android-goldfish-3.4-l-mr1-dev
  remotes/origin/linux-goldfish-3.0-wip
  remotes/origin/master
enisa@enisa-vm:~/goldfish$
```

Figure 43 Kernel branches

4.8.4.3 For this exercise the android kernel version 2.6.29 is going to be used. To do this students are going to create a new branch named lime tracking the 2.6.29 kernel source. Issue the following command:

```
git branch --track lime remotes/origin/android-goldfish-2.6.29
git checkout lime
```



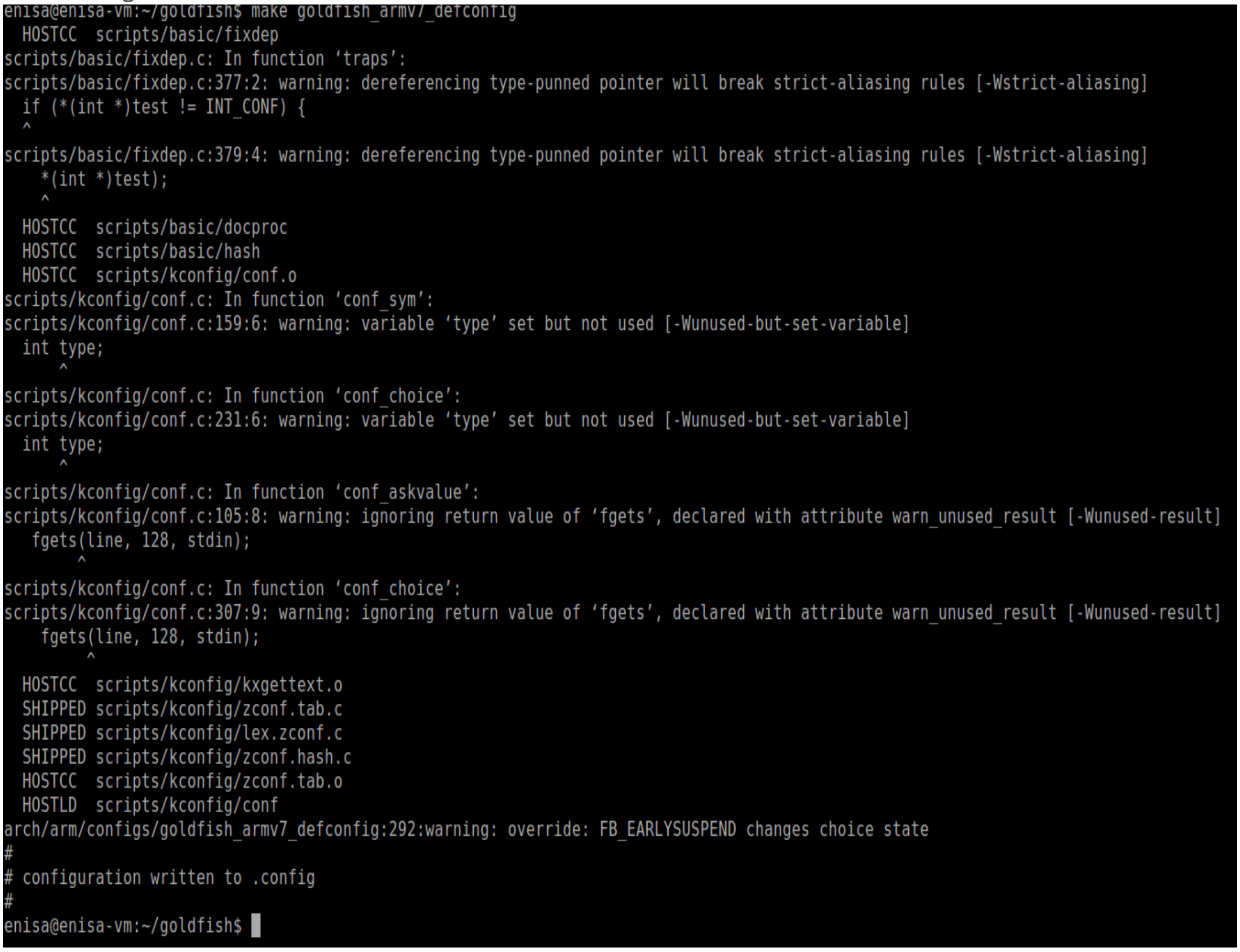
```
enisa@enisa-vm:~/goldfish$ git branch --track lime remotes/origin/android-goldfish-2.6.29
Branch lime set up to track remote branch android-goldfish-2.6.29 from origin.
enisa@enisa-vm:~/goldfish$ git branch -a
  lime
* master
  remotes/origin/HEAD -> origin/master
  remotes/origin/android-3.10
  remotes/origin/android-3.4
  remotes/origin/android-goldfish-2.6.29
  remotes/origin/android-goldfish-3.10
  remotes/origin/android-goldfish-3.10-l-mr1-dev
  remotes/origin/android-goldfish-3.10-m-dev
  remotes/origin/android-goldfish-3.4
  remotes/origin/android-goldfish-3.4-l-mr1-dev
  remotes/origin/linux-goldfish-3.0-wip
  remotes/origin/master
enisa@enisa-vm:~/goldfish$ git checkout lime
Checking out files: 100% (26821/26821), done.
Switched to branch 'lime'
Your branch is up-to-date with 'origin/android-goldfish-2.6.29'.
```

Figure 44 Create branch lime and checkout

4.8.4.4 In order to compile the kernel you will need the configuration file. Usually the configuration file for the kernel is included by the OEM in the source. Additionally if the kernel that is built in the device supports it, it can be extracted from it by pulling the /proc/config.gz file. In this case students will use the included configuration file in *arch/arm/configs/goldfish_armv7_defconfig* . First you will need to set the environment variables to compile the kernel as shown below:

```
enisa@enisa-vm:~/goldfish$ export ARCH=arm
enisa@enisa-vm:~/goldfish$ export SUBARCH=arm
enisa@enisa-vm:~/goldfish$ export CROSS_COMPILE=/usr/share/android-
ndk/toolchains/arm-linux-androideabi-4.6/prebuilt/linux-x86_64/bin/arm-
linux-androideabi-
```

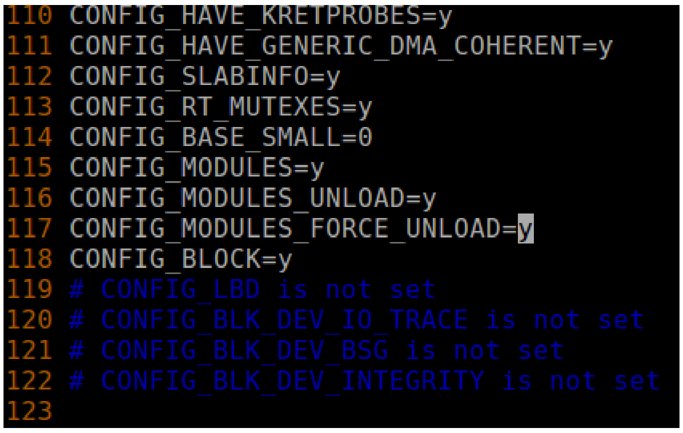
4.8.4.5 Afterwards students will create the initial configuration file from the goldfish default configuration.



```
enisa@enisa-vm:~/goldfish$ make goldfish_armv7_defconfig
  HOSTCC  scripts/basic/fixdep
scripts/basic/fixdep.c: In function ‘traps’:
scripts/basic/fixdep.c:377:2: warning: dereferencing type-punned pointer will break strict-aliasing rules [-Wstrict-aliasing]
  if (*(int *)test != INT_CONF) {
  ^
scripts/basic/fixdep.c:379:4: warning: dereferencing type-punned pointer will break strict-aliasing rules [-Wstrict-aliasing]
    *(int *)test);
    ^
  HOSTCC  scripts/basic/docproc
  HOSTCC  scripts/basic/hash
  HOSTCC  scripts/kconfig/conf.o
scripts/kconfig/conf.c: In function ‘conf_sym’:
scripts/kconfig/conf.c:159:6: warning: variable ‘type’ set but not used [-Wunused-but-set-variable]
  int type;
      ^
scripts/kconfig/conf.c: In function ‘conf_choice’:
scripts/kconfig/conf.c:231:6: warning: variable ‘type’ set but not used [-Wunused-but-set-variable]
  int type;
      ^
scripts/kconfig/conf.c: In function ‘conf_askvalue’:
scripts/kconfig/conf.c:105:8: warning: ignoring return value of ‘fgets’, declared with attribute warn_unused_result [-Wunused-result]
   fgets(line, 128, stdin);
        ^
scripts/kconfig/conf.c: In function ‘conf_choice’:
scripts/kconfig/conf.c:307:9: warning: ignoring return value of ‘fgets’, declared with attribute warn_unused_result [-Wunused-result]
    fgets(line, 128, stdin);
         ^
  HOSTCC  scripts/kconfig/kxgettext.o
  SHIPPED scripts/kconfig/zconf.tab.c
  SHIPPED scripts/kconfig/lex.zconf.c
  SHIPPED scripts/kconfig/zconf.hash.c
  HOSTCC  scripts/kconfig/zconf.tab.o
  HOSTLD  scripts/kconfig/conf
arch/arm/configs/goldfish_armv7_defconfig:292:warning: override: FB_EARLYSUSPEND changes choice state
#
# configuration written to .config
#
enisa@enisa-vm:~/goldfish$
```

Figure 45 Initial config file creation

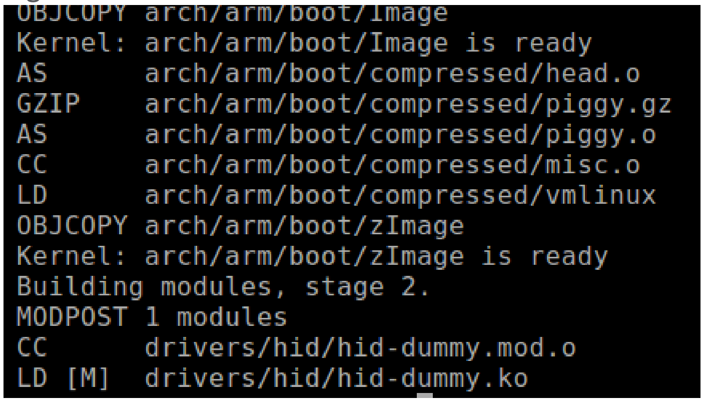
4.8.4.6 Next edit the configuration file to enable module loading. Open .config file and edit line 115 as shown below:



```
110 CONFIG_HAVE_KRETPROBES=y
111 CONFIG_HAVE_GENERIC_DMA_COHERENT=y
112 CONFIG_SLABINFO=y
113 CONFIG_RT_MUTEXES=y
114 CONFIG_BASE_SMALL=0
115 CONFIG_MODULES=y
116 CONFIG_MODULES_UNLOAD=y
117 CONFIG_MODULES_FORCE_UNLOAD=y
118 CONFIG_BLOCK=y
119 # CONFIG_LBD is not set
120 # CONFIG_BLK_DEV_IO_TRACE is not set
121 # CONFIG_BLK_DEV_BSG is not set
122 # CONFIG_BLK_DEV_INTEGRITY is not set
123
```

Figure 46 Configuration change

4.8.4.7 Compile the kernel issuing make command. When finished the compiled kernel should be in arch/arm/boot/zImage.



```
OBJCOPY arch/arm/boot/Image
Kernel: arch/arm/boot/Image is ready
AS      arch/arm/boot/compressed/head.o
GZIP    arch/arm/boot/compressed/piggy.gz
AS      arch/arm/boot/compressed/piggy.o
CC      arch/arm/boot/compressed/misc.o
LD      arch/arm/boot/compressed/vmlinux
OBJCOPY arch/arm/boot/zImage
Kernel: arch/arm/boot/zImage is ready
Building modules, stage 2.
MODPOST 1 modules
CC      drivers/hid/hid-dummy.mod.o
LD [M]  drivers/hid/hid-dummy.ko
```

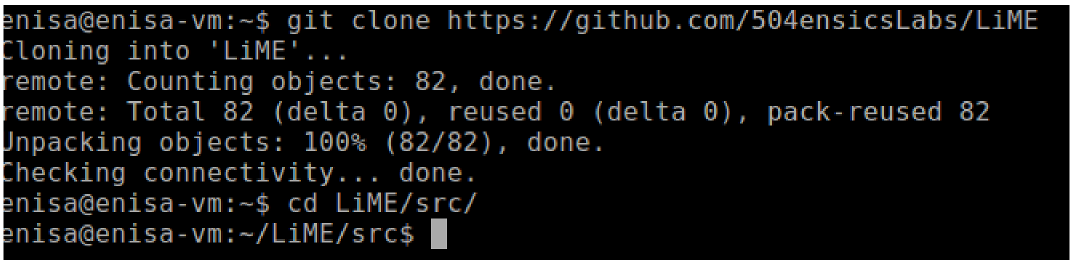
Figure 47 Kernel Compilation

4.8.4.8 Now you can start the emulator with the kernel you have just compiled issuing the following command.

```
enisa@enisa-vm:~/goldfish$ emulator -avd Nexus -kernel
arch/arm/boot/zImage
```

4.8.4.9 Next you need to download the lime module and compile it.

```
enisa@enisa-vm:~$ git clone https://github.com/504ensicsLabs/LiME
```



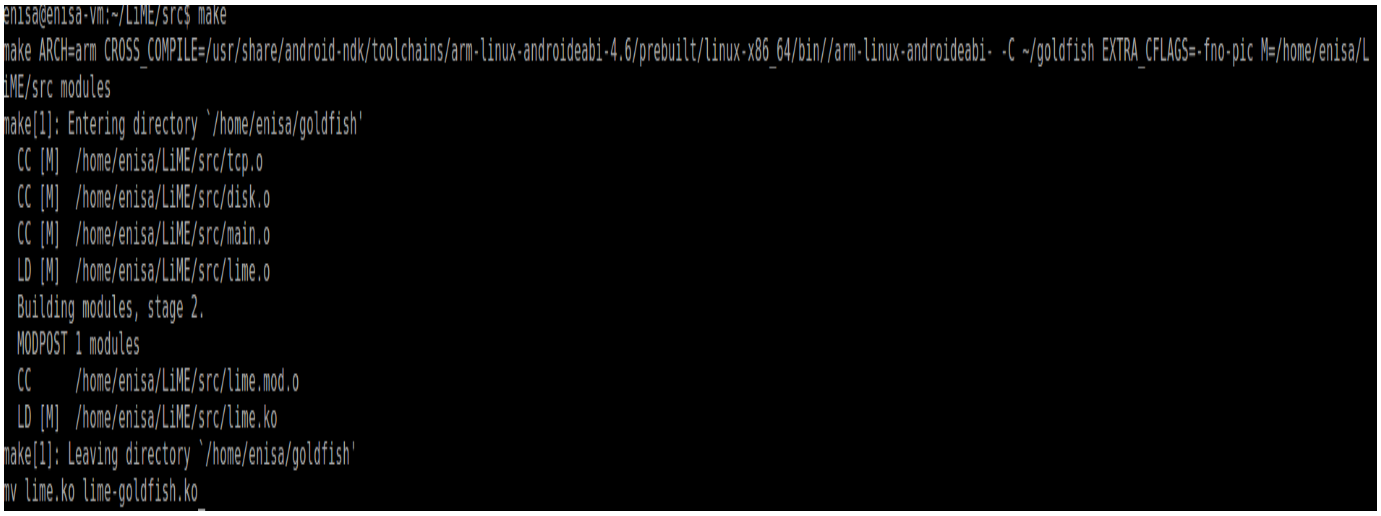
```
enisa@enisa-vm:~$ git clone https://github.com/504ensicsLabs/LiME
Cloning into 'LiME'...
remote: Counting objects: 82, done.
remote: Total 82 (delta 0), reused 0 (delta 0), pack-reused 82
Unpacking objects: 100% (82/82), done.
Checking connectivity... done.
enisa@enisa-vm:~$ cd LiME/src/
enisa@enisa-vm:~/LiME/src$
```

Figure 48 Download lime source

4.8.4.10 Edit the Makefile accordingly.

```
enisa@enisa-vm:~/LiME/src$ diff Makefile ~/Makefile
25a26,27
> KDIR_GOLDFISH := ~/goldfish
> CCPATH :=/usr/share/android-ndk/toolchains/arm-linux-androideabi-
4.6/prebuilt/linux-x86_64/bin/
33,35c35,38
<     $(MAKE) -C /lib/modules/$(KVER)/build M=$(PWD) modules
<     strip --strip-unneeded lime.ko
<     mv lime.ko lime-$(KVER).ko
---
>     $(MAKE) ARCH=arm CROSS_COMPILE=$(CCPATH)/arm-linux-androideabi- -C
$(KDIR_GOLDFISH) EXTRA_CFLAGS=-fno-pic M=$(PWD) modules
>     mv lime.ko lime-goldfish.ko
```

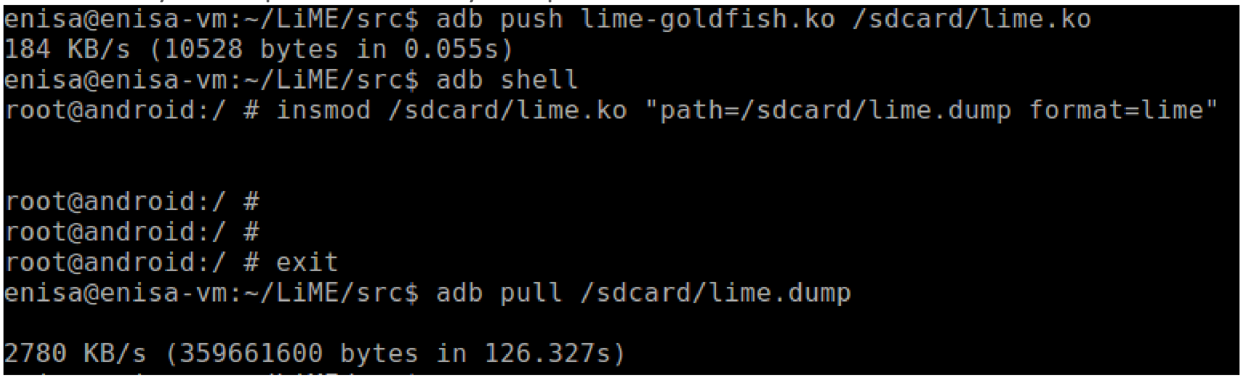
Then compile the module issuing make



```
enisa@enisa-vm:~/LiME/src$ make
make ARCH=arm CROSS_COMPILE=/usr/share/android-ndk/toolchains/arm-linux-androideabi-4.6/prebuilt/linux-x86_64/bin//arm-linux-androideabi- -C ~/goldfish EXTRA_CFLAGS=-fno-pic M=/home/enisa/L
iME/src modules
make[1]: Entering directory `/home/enisa/goldfish'
  CC [M]  /home/enisa/LiME/src/tcp.o
  CC [M]  /home/enisa/LiME/src/disk.o
  CC [M]  /home/enisa/LiME/src/main.o
  LD [M]  /home/enisa/LiME/src/lime.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/enisa/LiME/src/lime.mod.o
  LD [M]  /home/enisa/LiME/src/lime.ko
make[1]: Leaving directory `/home/enisa/goldfish'
mv lime.ko lime-goldfish.ko
```

Figure 49 Module compilation

4.8.4.11 Next push the compiled module to the running emulator and set the memory dump path. Afterwards you can pull the memory dump as show below:



```
enisa@enisa-vm:~/LiME/src$ adb push lime-goldfish.ko /sdcard/lime.ko
184 KB/s (10528 bytes in 0.055s)
enisa@enisa-vm:~/LiME/src$ adb shell
root@android:/ # insmod /sdcard/lime.ko "path=/sdcard/lime.dump format=lime"


root@android:/ #
root@android:/ #
root@android:/ # exit
enisa@enisa-vm:~/LiME/src$ adb pull /sdcard/lime.dump

2780 KB/s (359661600 bytes in 126.327s)
```

Figure 50 LiMe memory dump.

4.8.5 Examining memory dump with Volatility

4.8.5.1 Build a Volatility Profile

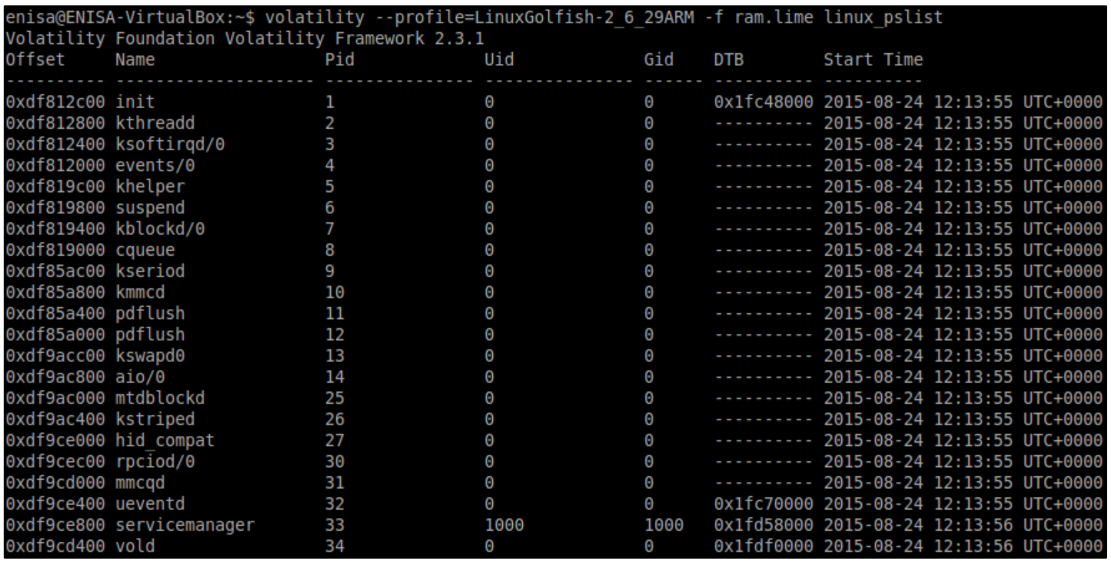
Volatility uses profiles to properly analyse RAM dump. For Android an already prepared profile called LinuxGolfish-2_6_29ARM should be used. If the students want to create their own profiles, they should refer to another exercise by ENISA: https://www.enisa.europa.eu/activities/cert/training/training-resources/documents/advanced-artifact-handling-handbook (Section 2.2.2.3, Task 1.2.3 Building a Volatility profile).

4.8.5.2 Examine the Memory Dump with Volatility

Since Android is based on Linux, students can use any of the Linux-related Volatility commands[88] to analyse the memory dump. Mostly used commands for Android are explained below. The descriptions are copied from Volatility project website:

- linux_pslist

This plugin prints the list of active processes starting from the init_task symbol and walking the task_struct->tasks linked list. It does not display the swapper process. If the DTB column is blank, the item is likely a kernel thread.



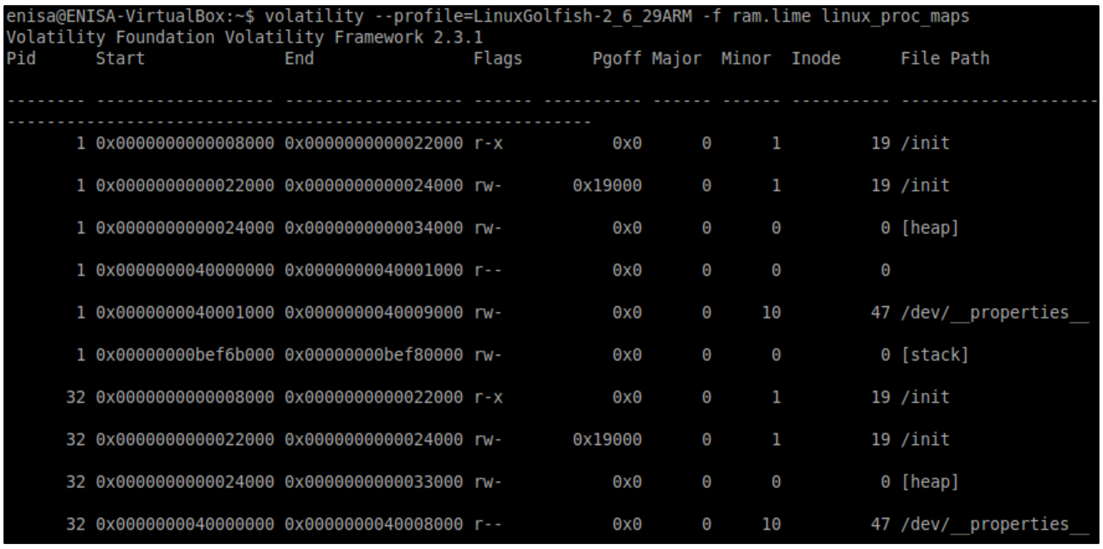
```
enisa@ENISA-VirtualBox:~$ volatility --profile=LinuxGolfish-2_6_29ARM -f ram.lime linux_pslist
Volatility Foundation Volatility Framework 2.3.1
Offset     Name                 Pid             Uid             Gid    DTB        Start Time
---------- -------------------- --------------- --------------- ------ ---------- ----------
0xdf812c00 init                 1               0               0      0x1fc48000 2015-08-24 12:13:55 UTC+0000
0xdf812800 kthreadd             2               0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf812400 ksoftirqd/0          3               0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf812000 events/0             4               0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf819c00 khelper              5               0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf819800 suspend              6               0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf819400 kblockd/0            7               0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf819000 cqueue               8               0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf85ac00 kseriod              9               0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf85a800 kmmcd                10              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf85a400 pdflush              11              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf85a000 pdflush              12              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf9acc00 kswapd0              13              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf9ac800 aio/0                14              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf9ac000 mtdblockd            25              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf9ac400 kstriped             26              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf9ce000 hid_compat           27              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf9cec00 rpciod/0             30              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf9cd000 mmcqd                31              0               0      ---------- 2015-08-24 12:13:55 UTC+0000
0xdf9ce400 uevent               32              0               0      0x1fc70000 2015-08-24 12:13:55 UTC+0000
0xdf9ce800 servicemanager       33              1000            1000   0x1fd58000 2015-08-24 12:13:56 UTC+0000
0xdf9cd400 vold                 34              0               0      0x1fdf0000 2015-08-24 12:13:56 UTC+0000
```

Figure 51 linux_pslist

- linux_proc_maps

This plugin prints details of process memory, including heaps, stacks, and shared libraries.

[88] Volatility: A command reference for Linux, https://code.google.com/p/volatility/wiki/LinuxCommandReference23, last accessed on: 2015-09-14

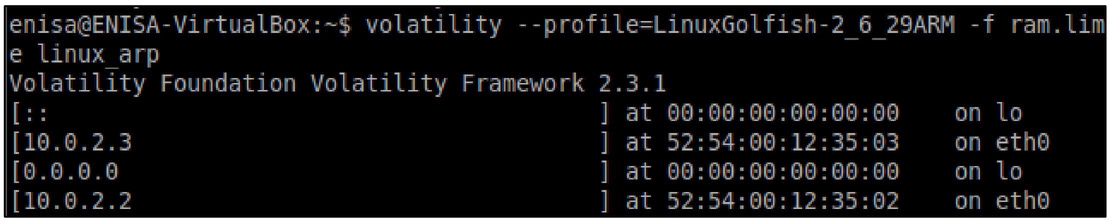


```
enisa@ENISA-VirtualBox:~$ volatility --profile=LinuxGolfish-2_6_29ARM -f ram.lime linux_proc_maps
Volatility Foundation Volatility Framework 2.3.1
Pid      Start              End                Flags       Pgoff Major  Minor  Inode      File Path

-------- ------------------ ------------------ ------ ---------- ------ ------ ---------- -------------------
--------------------------------------------------------------
       1 0x0000000000008000 0x0000000000022000 r-x          0x0      0      1         19 /init

       1 0x0000000000022000 0x0000000000024000 rw-      0x19000      0      1         19 /init

       1 0x0000000000024000 0x0000000000034000 rw-          0x0      0      0          0 [heap]

       1 0x0000000040000000 0x0000000040001000 r--          0x0      0      0          0

       1 0x0000000040001000 0x0000000040009000 rw-          0x0      0     10         47 /dev/__properties__

       1 0x00000000bef6b000 0x00000000bef80000 rw-          0x0      0      0          0 [stack]

      32 0x0000000000008000 0x0000000000022000 r-x          0x0      0      1         19 /init

      32 0x0000000000022000 0x0000000000024000 rw-      0x19000      0      1         19 /init

      32 0x0000000000024000 0x0000000000033000 rw-          0x0      0      0          0 [heap]

      32 0x0000000040000000 0x0000000040008000 r--          0x0      0     10         47 /dev/__properties__
```

Figure 52 linux_proc_maps

- linux_arp

This plugin prints the ARP table.

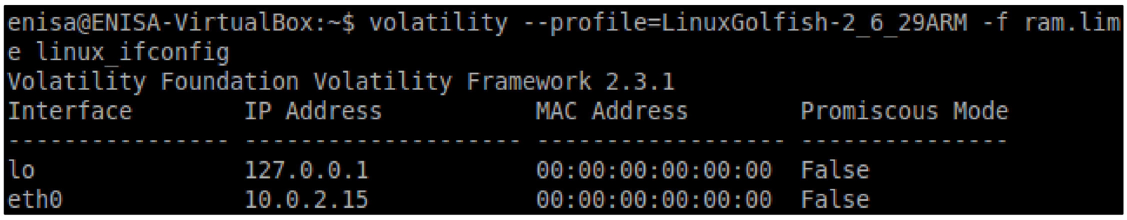


```
enisa@ENISA-VirtualBox:~$ volatility --profile=LinuxGolfish-2_6_29ARM -f ram.lim
e linux_arp
Volatility Foundation Volatility Framework 2.3.1
[::                                          ] at 00:00:00:00:00:00    on lo
[10.0.2.3                                    ] at 52:54:00:12:35:03    on eth0
[0.0.0.0                                     ] at 00:00:00:00:00:00    on lo
[10.0.2.2                                    ] at 52:54:00:12:35:02    on eth0
```

Figure 53 linux_arp

- linux_ifconfig

This plugin prints the active interface information, including IPs, interface name, MAC address, and whether the NIC is in promiscuous mode or not (sniffing).

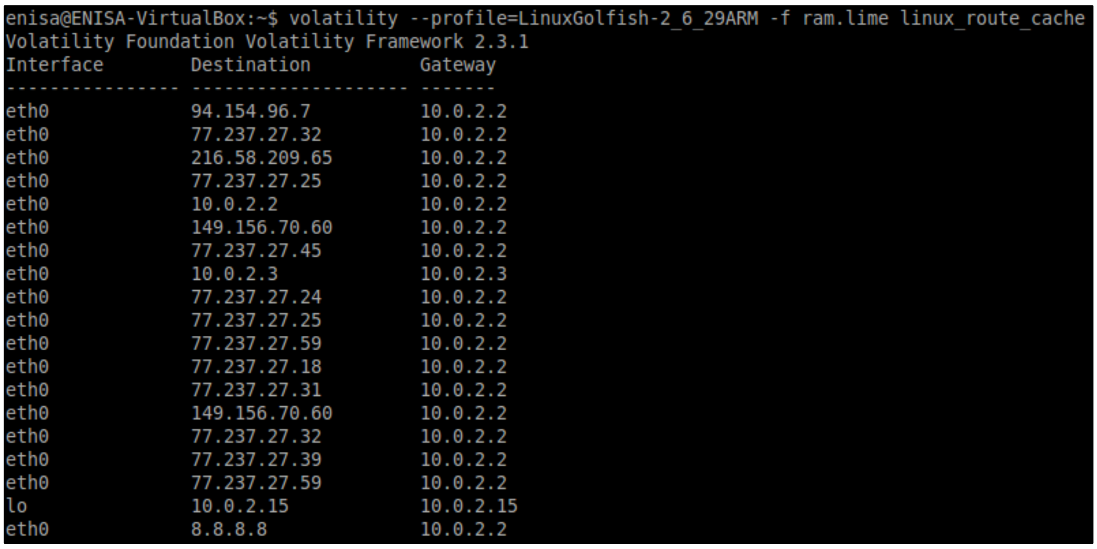


```
enisa@ENISA-VirtualBox:~$ volatility --profile=LinuxGolfish-2_6_29ARM -f ram.lim
e linux_ifconfig
Volatility Foundation Volatility Framework 2.3.1
Interface        IP Address           MAC Address        Promiscous Mode
---------------- -------------------- ------------------ ---------------
lo               127.0.0.1            00:00:00:00:00:00  False
eth0             10.0.2.15            00:00:00:00:00:00  False
```

Figure 54 linux_ifconfig

- linux_route_cache

This plugin enumerates the data in the routing table cache. It can show you which systems a machine communicated with in the past.

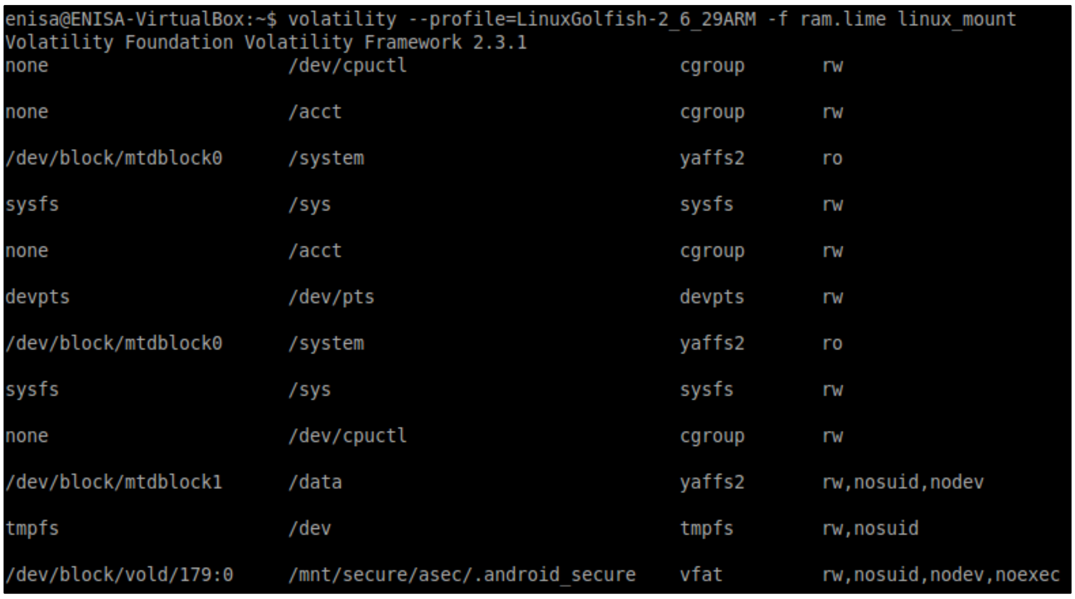


```
enisa@ENISA-VirtualBox:~$ volatility --profile=LinuxGolfish-2_6_29ARM -f ram.lime linux_route_cache
Volatility Foundation Volatility Framework 2.3.1
Interface        Destination          Gateway
---------------- -------------------- -------
eth0             94.154.96.7          10.0.2.2
eth0             77.237.27.32         10.0.2.2
eth0             216.58.209.65        10.0.2.2
eth0             77.237.27.25         10.0.2.2
eth0             10.0.2.2             10.0.2.2
eth0             149.156.70.60        10.0.2.2
eth0             77.237.27.45         10.0.2.2
eth0             10.0.2.3             10.0.2.3
eth0             77.237.27.24         10.0.2.2
eth0             77.237.27.25         10.0.2.2
eth0             77.237.27.59         10.0.2.2
eth0             77.237.27.18         10.0.2.2
eth0             77.237.27.31         10.0.2.2
eth0             149.156.70.60        10.0.2.2
eth0             77.237.27.32         10.0.2.2
eth0             77.237.27.39         10.0.2.2
eth0             77.237.27.59         10.0.2.2
lo               10.0.2.15            10.0.2.15
eth0             8.8.8.8              10.0.2.2
```

Figure 55 linux_route_cache

- linux_mount

This plugins mimics of the output of /proc/mounts on a running Linux system. For each mountpoint it prints the flags, mounted source (drive, network share, etc) and the director it is mounted on.



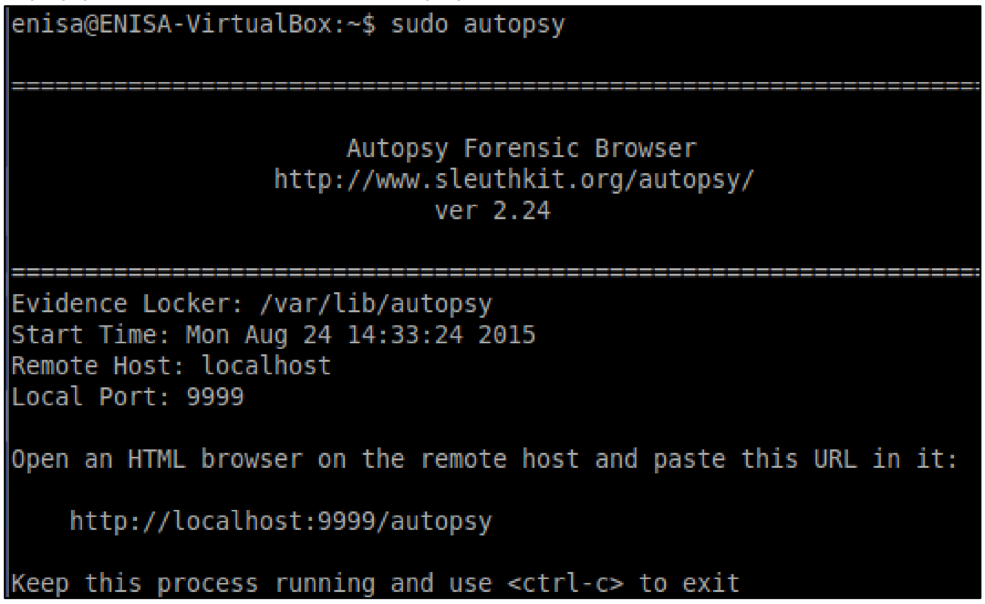
```
enisa@ENISA-VirtualBox:~$ volatility --profile=LinuxGolfish-2_6_29ARM -f ram.lime linux_mount
Volatility Foundation Volatility Framework 2.3.1
none                      /dev/cpuctl                          cgroup       rw

none                      /acct                                cgroup       rw

/dev/block/mtdblock0      /system                              yaffs2       ro

sysfs                     /sys                                 sysfs        rw

none                      /acct                                cgroup       rw

devpts                    /dev/pts                             devpts       rw

/dev/block/mtdblock0      /system                              yaffs2       ro

sysfs                     /sys                                 sysfs        rw

none                      /dev/cpuctl                          cgroup       rw

/dev/block/mtdblock1      /data                                yaffs2       rw,nosuid,nodev

tmpfs                     /dev                                 tmpfs        rw,nosuid

/dev/block/vold/179:0     /mnt/secure/asec/.android_secure     vfat         rw,nosuid,nodev,noexec
```

Figure 56 linux_mount

4.8.6 Task walk-through - Using Autopsy

Autopsy analyses the files extracted from an Android device. It supports physical dumps from most of Android devices (please note that in this exercise physical acquisition methods are not explained) as well as raw memory dump files.

4.8.6.1 To run Autopsy you need to start the Autopsy service as a root user.



```
enisa@ENISA-VirtualBox:~$ sudo autopsy

============================================================================

                    Autopsy Forensic Browser
               http://www.sleuthkit.org/autopsy/
                           ver 2.24

============================================================================
Evidence Locker: /var/lib/autopsy
Start Time: Mon Aug 24 14:33:24 2015
Remote Host: localhost
Local Port: 9999

Open an HTML browser on the remote host and paste this URL in it:

    http://localhost:9999/autopsy

Keep this process running and use <ctrl-c> to exit
```

Figure 57

4.8.6.2 When the service is started, open web browser and type this address: http://localhost:9999/autopsy.

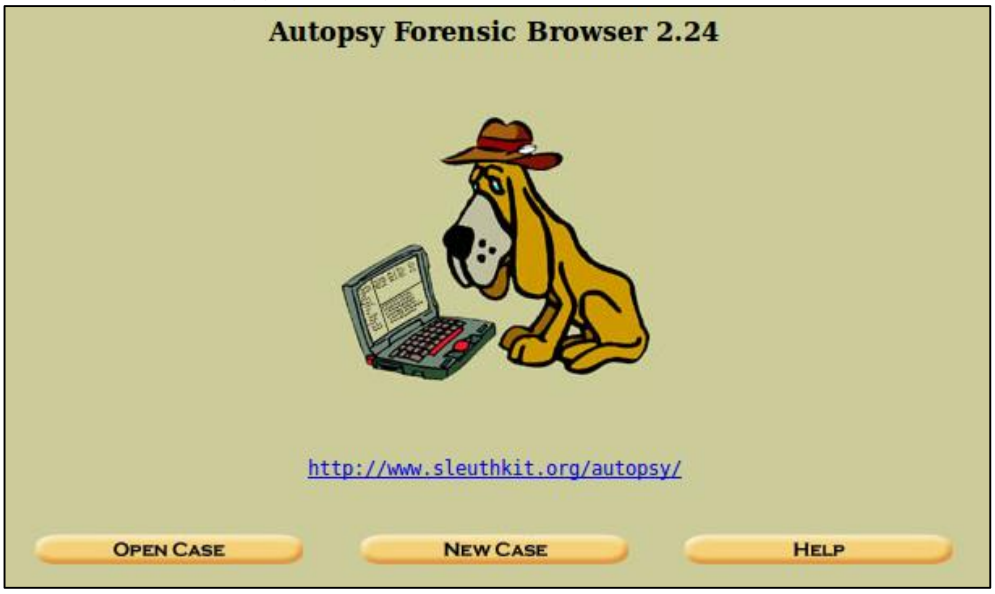
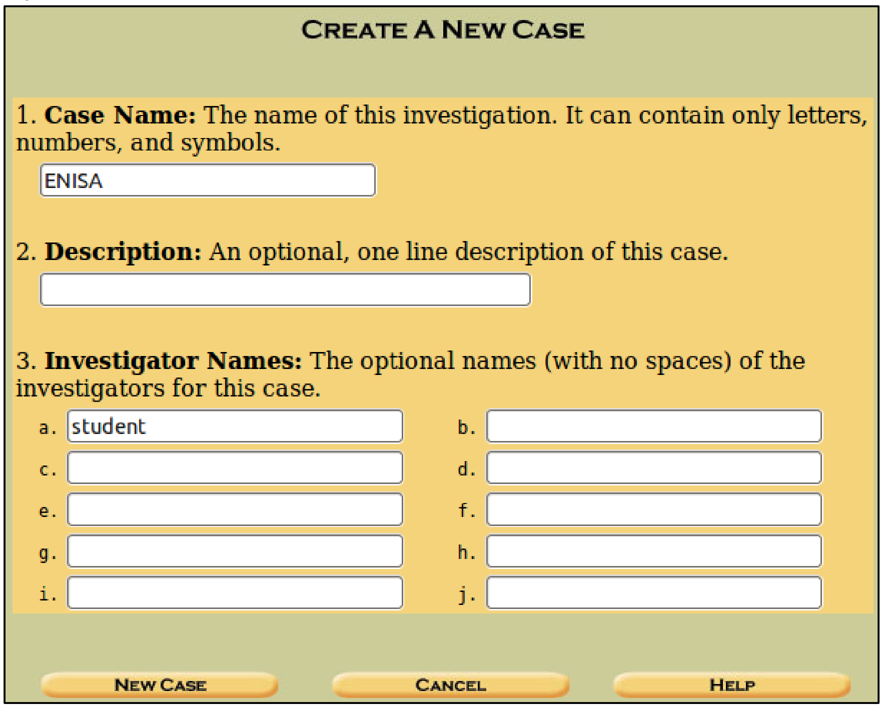

Autopsy Forensic Browser 2.24
http://www.sleuthkit.org/autopsy/
OPEN CASE
NEW CASE
HELP

Figure 58

4.8.6.3 To create new case click on NEW CASE. To open existing one, click OPEN CASE. When you create a new case you have to fill in information such as "Case name" and "Investigator names". Description is optional.


CREATE A NEW CASE

1. **Case Name:** The name of this investigation. It can contain only letters, numbers, and symbols.
ENISA

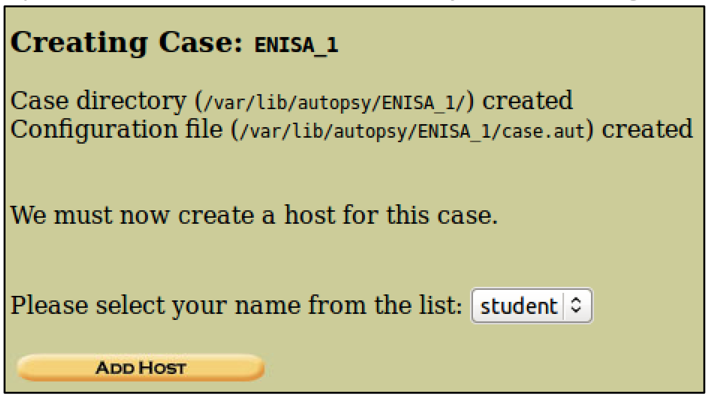
2. **Description:** An optional, one line description of this case.

3. **Investigator Names:** The optional names (with no spaces) of the investigators for this case.

a.	student	b.	
c.		d.	
e.		f.	
g.		h.	
i.		j.	

Figure 59

4.8.6.4 Next you will be prompted to add a host i.e. device subject to investigation.

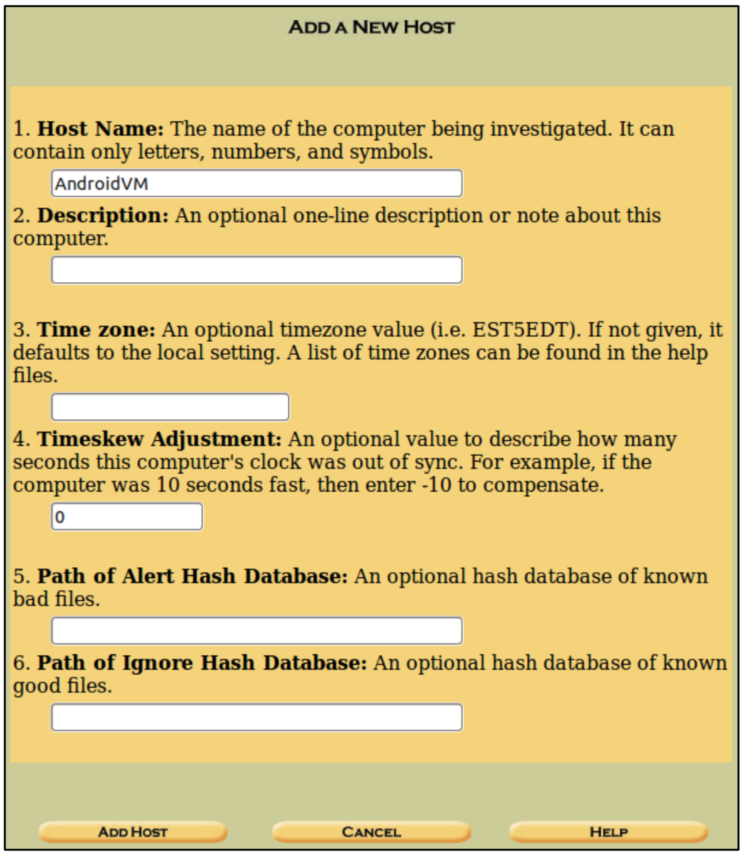

Creating Case: ENISA_1

Case directory (/var/lib/autopsy/ENISA_1/) created
Configuration file (/var/lib/autopsy/ENISA_1/case.aut) created

We must now create a host for this case.

Please select your name from the list: student

Figure 60


ADD A NEW HOST

1. **Host Name:** The name of the computer being investigated. It can contain only letters, numbers, and symbols.

AndroidVM

2. **Description:** An optional one-line description or note about this computer.

3. **Time zone:** An optional timezone value (i.e. EST5EDT). If not given, it defaults to the local setting. A list of time zones can be found in the help files.

4. **Timeskew Adjustment:** An optional value to describe how many seconds this computer's clock was out of sync. For example, if the computer was 10 seconds fast, then enter -10 to compensate.

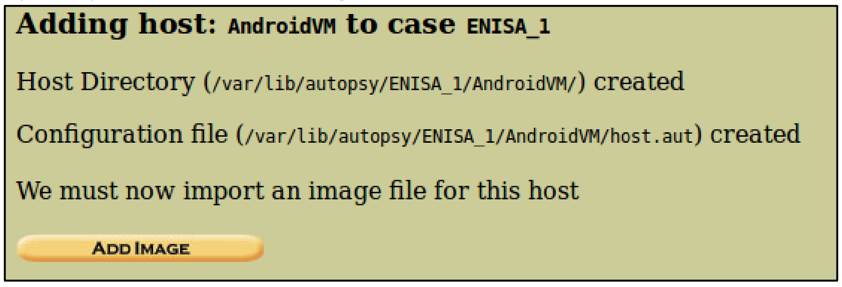
0

5. **Path of Alert Hash Database:** An optional hash database of known bad files.

6. **Path of Ignore Hash Database:** An optional hash database of known good files.

Figure 61

4.8.6.5 Next you will be prompted to add an image of the host / device.

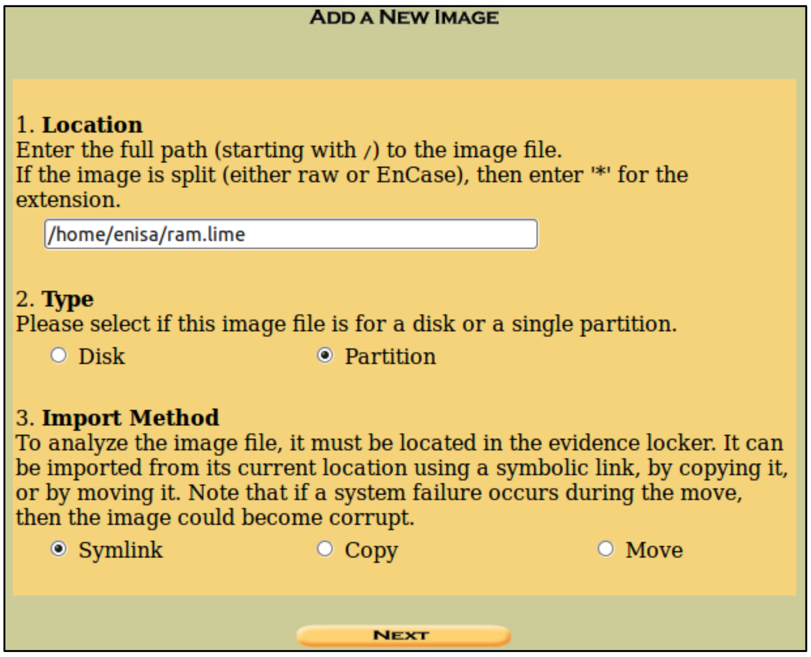

Adding host: `AndroidVM` to case `ENISA_1`

Host Directory (`/var/lib/autopsy/ENISA_1/AndroidVM/`) created

Configuration file (`/var/lib/autopsy/ENISA_1/AndroidVM/host.aut`) created

We must now import an image file for this host

Figure 62


ADD A NEW IMAGE

1. **Location**
Enter the full path (starting with /) to the image file.
If the image is split (either raw or EnCase), then enter '*' for the extension.
/home/enisa/ram.lime

2. **Type**
Please select if this image file is for a disk or a single partition.
○ Disk [✓ Partition]

3. **Import Method**
To analyze the image file, it must be located in the evidence locker. It can be imported from its current location using a symbolic link, by copying it, or by moving it. Note that if a system failure occurs during the move, then the image could become corrupt.
[✓ Symlink] ○ Copy ○ Move

Figure 63

4.8.6.6 Next step is to choose proper file system of the added image. Choose "raw" for the RAM memory dump.

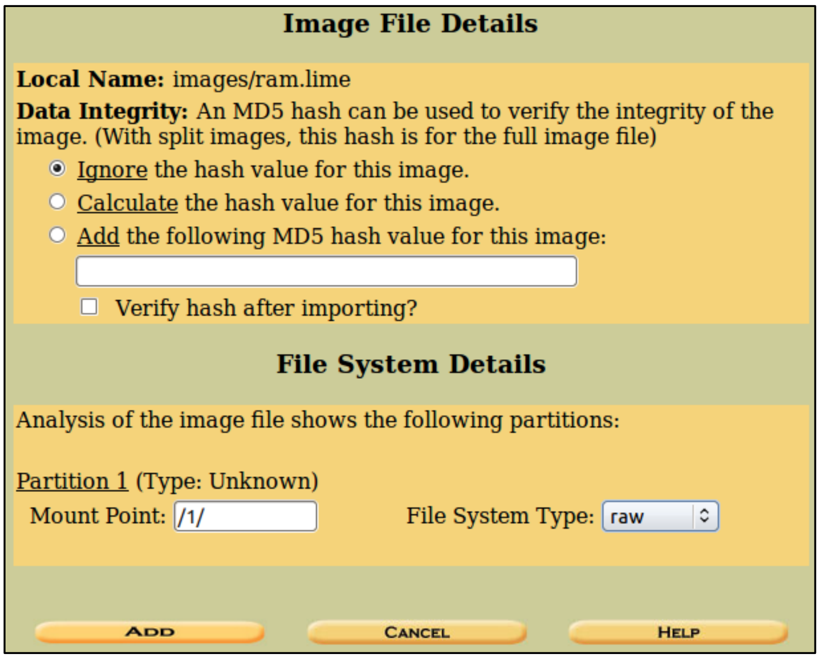


Image File Details

Local Name: images/ram.lime
Data Integrity: An MD5 hash can be used to verify the integrity of the image. (With split images, this hash is for the full image file)
[✓ Ignore the hash value for this image.]
○ Calculate the hash value for this image.
○ Add the following MD5 hash value for this image:
☐ Verify hash after importing?

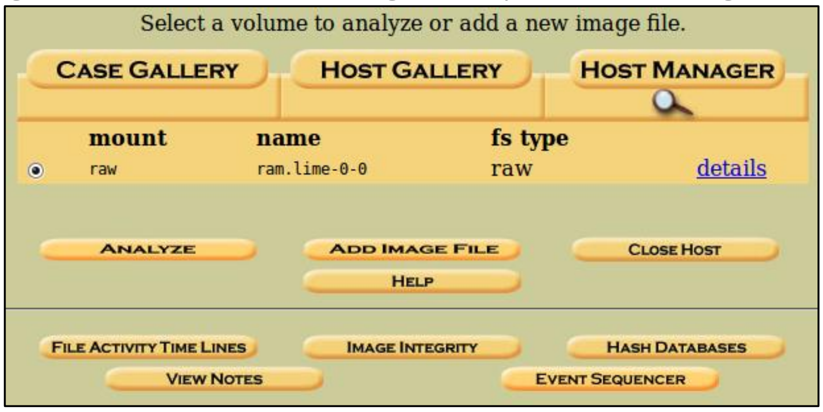
File System Details

Analysis of the image file shows the following partitions:

Partition 1 (Type: Unknown)
Mount Point: /1/ File System Type: raw

Figure 64

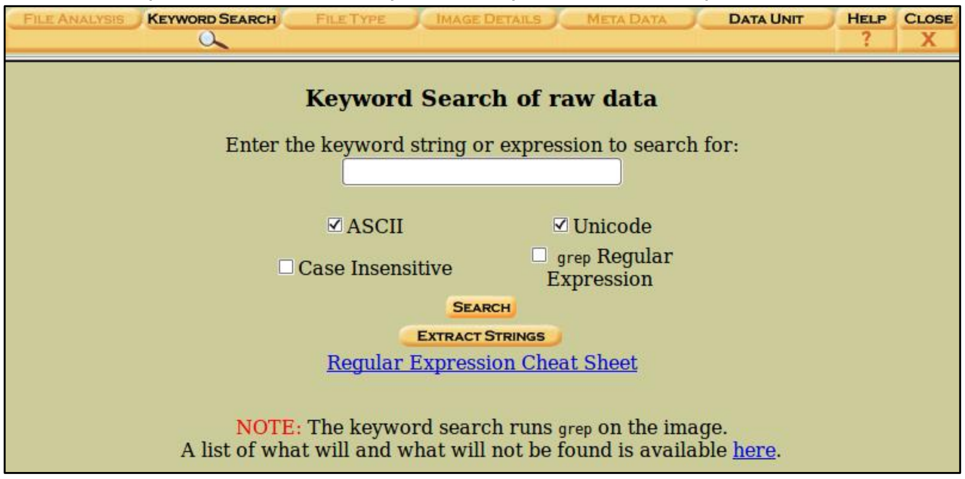
4.8.6.7 Once completed you will be presented with the following screen which allows to run analysis, add another image file, close the file or among other options check image's integrity.


Select a volume to analyze or add a new image file.

	mount	name	fs type	
●	raw	ram.lime-0-0	raw	details

Figure 65

4.8.6.8 Since the image subject to analysis if a raw image, some functionalities may not be available. As an example, try and search for keyword "@enisa.europa.eu" which in this case was used for the e-mail address set up on the Android system. Try to locate the password of an e-mail account.



Keyword Search of raw data

Enter the keyword string or expression to search for:

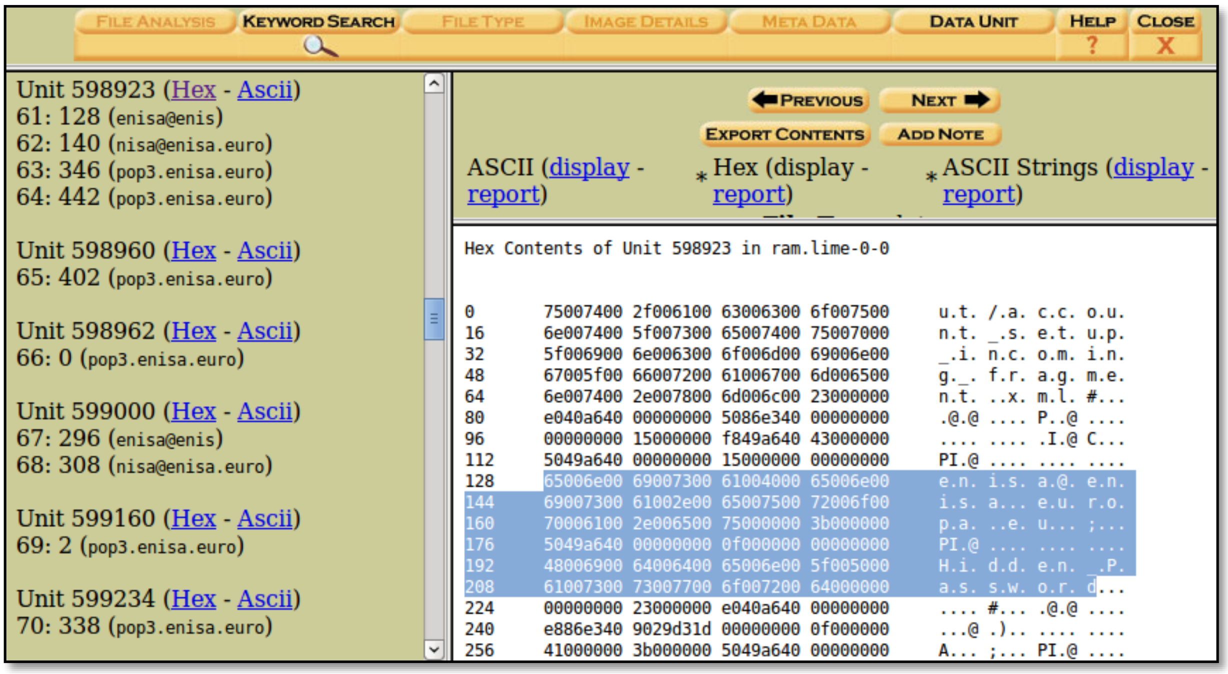
☑ ASCII ☑ Unicode

☐ Case Insensitive ☐ grep Regular Expression

Regular Expression Cheat Sheet

NOTE: The keyword search runs grep on the image.
A list of what will and what will not be found is available here.

Figure 66



Unit 598923 (Hex - Ascii)
61: 128 (enisa@enis)
62: 140 (nisa@enisa.euro)
63: 346 (pop3.enisa.euro)
64: 442 (pop3.enisa.euro)

Unit 598960 (Hex - Ascii)
65: 402 (pop3.enisa.euro)

Unit 598962 (Hex - Ascii)
66: 0 (pop3.enisa.euro)

Unit 599000 (Hex - Ascii)
67: 296 (enisa@enis)
68: 308 (nisa@enisa.euro)

Unit 599160 (Hex - Ascii)
69: 2 (pop3.enisa.euro)

Unit 599234 (Hex - Ascii)
70: 338 (pop3.enisa.euro)

ASCII (display - report) * Hex (display - report) * ASCII Strings (display - report)

```
Hex Contents of Unit 598923 in ram.lime-0-0


0       75007400 2f006100 63006300 6f007500     u.t. /.a. c.c. o.u.
16      6e007400 5f007300 65007400 75007000     n.t. _.s. e.t. u.p.
32      5f006900 6e006300 6f006d00 69006e00     _.i. n.c. o.m. i.n.
48      67005f00 66007200 61006700 6d006500     g._. f.r. a.g. m.e.
64      6e007400 2e007800 6d006c00 23000000     n.t. ..x. m.l. #...
80      e040a640 00000000 5086e340 00000000     .@.@ .... P..@ ....
96      00000000 15000000 f849a640 43000000     .... .... .I.@ C...
112     5049a640 00000000 15000000 00000000     PI.@ .... .... ....
128     65006e00 69007300 61004000 65006e00     e.n. i.s. a.@. e.n.
144     69007300 61002e00 65007500 72006f00     i.s. a... e.u. r.o.
160     70006100 2e006500 75000000 3b000000     p.a. ..e. u... ;...
176     5049a640 00000000 0f000000 00000000     PI.@ .... .... ....
192     48006900 64006400 65006e00 5f005000     H.i. d.d. e.n. _.P.
208     61007300 73007700 6f007200 64000000     a.s. s.w. o.r. d...
224     00000000 23000000 e040a640 00000000     .... #... .@.@ ....
240     e886e340 9029d31d 00000000 0f000000     ...@ .).. .... ....
256     41000000 3b000000 5049a640 00000000     A... ;... PI.@ ....
```

Figure 67

4.9 Task 4.5: iOS – iPhone Backup Analyser 2

4.9.1 Introduction

In this task, the students are given a set of files from iTunes backup system for analysis with iPhone Backup Analyser 2 (iPBA2[89]). This software allows the user to browse through the content of an iPhone/iPad backup made by iTunes (or other software able to perform iOS devices' backup).

4.9.2 Details

As explained on iPBA2 project website, it parses the backup directory and shows decoded file system tree. Each file can be clicked to see its properties, such as:

- Real name and name in the backup directory
- File UNIX permissions
- Data hash (as calculated by iOS)
- User and group ID
- Modify time, access time, creation time
- File type (from magic numbers)

Any built-in viewer will allow to browse through known file formats e.g.:

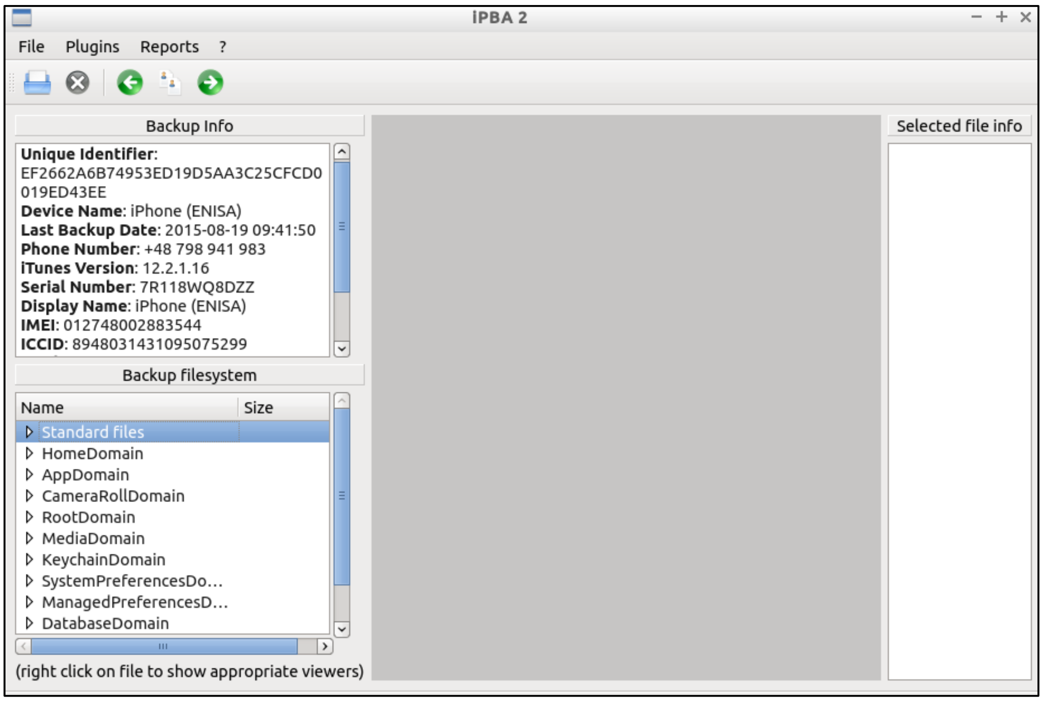
[89] iPBA2, https://github.com/PicciMario/iPhone-Backup-Analyzer-2, last accessed on: 2015-10-19

- ASCII viewer
- PLIST structure browser
- SQLITE browser
- HEX viewer

4.9.3 Task walk-through

This section will explain possible ways to analysis of data stored in iOS backup.

4.9.3.1 Open iPBA2 and open folder ef2662a6b74953ed19d5aa3c25cfcd0019ed43ee.

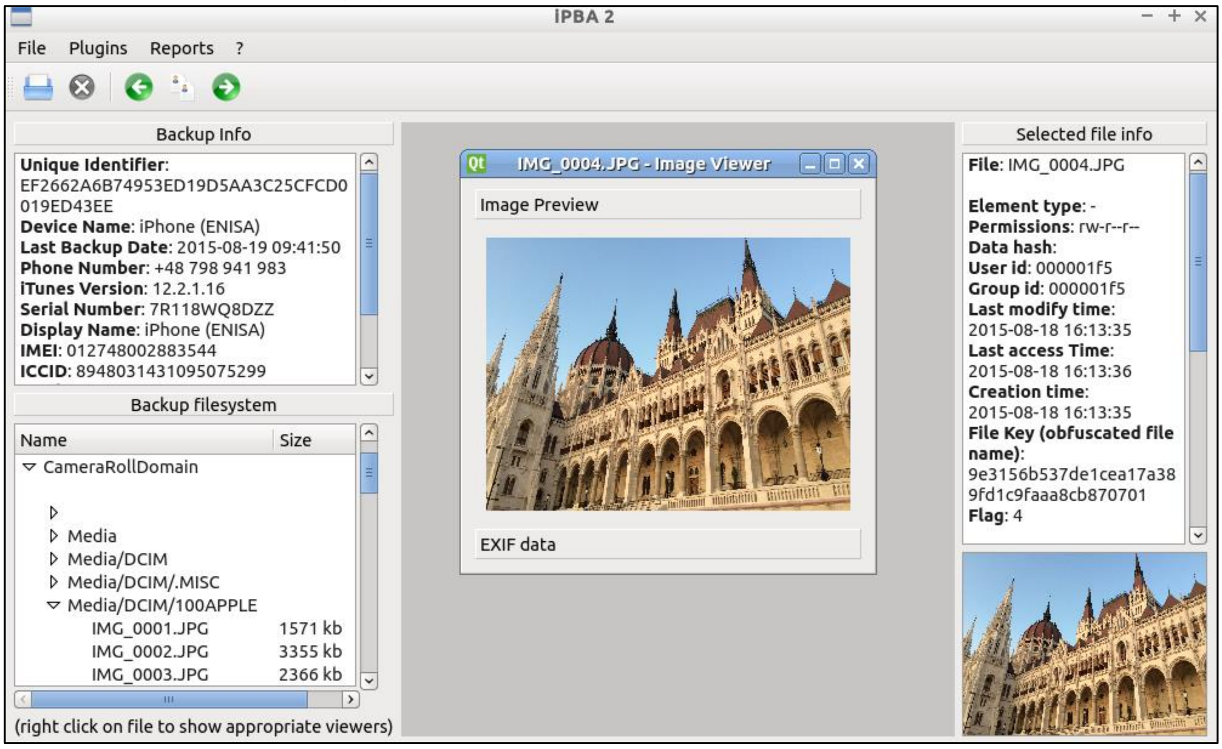


Unique Identifier: EF2662A6B74953ED19D5AA3C25CFCD0019ED43EE
Device Name: iPhone (ENISA)
Last Backup Date: 2015-08-19 09:41:50
Phone Number: +48 798 941 983
iTunes Version: 12.2.1.16
Serial Number: 7R118WQ8DZZ
Display Name: iPhone (ENISA)
IMEI: 012748002883544
ICCID: 8948031431095075299

Figure 68

4.9.3.2 You can use predefined PLUGINS to view some data and create reports from predefined REPORTS tools.

4.9.3.3 Take a look at the following option: CameraRollDomain -> Media/DCIM/100APPLE.



Unique Identifier: EF2662A6B74953ED19D5AA3C25CFCD0019ED43EE
Device Name: iPhone (ENISA)
Last Backup Date: 2015-08-19 09:41:50
Phone Number: +48 798 941 983
iTunes Version: 12.2.1.16
Serial Number: 7R118WQ8DZZ
Display Name: iPhone (ENISA)
IMEI: 012748002883544
ICCID: 8948031431095075299

Name	Size
CameraRollDomain	
Media	
Media/DCIM	
Media/DCIM/.MISC	
Media/DCIM/100APPLE	
IMG_0001.JPG	1571 kb
IMG_0002.JPG	3355 kb
IMG_0003.JPG	2366 kb

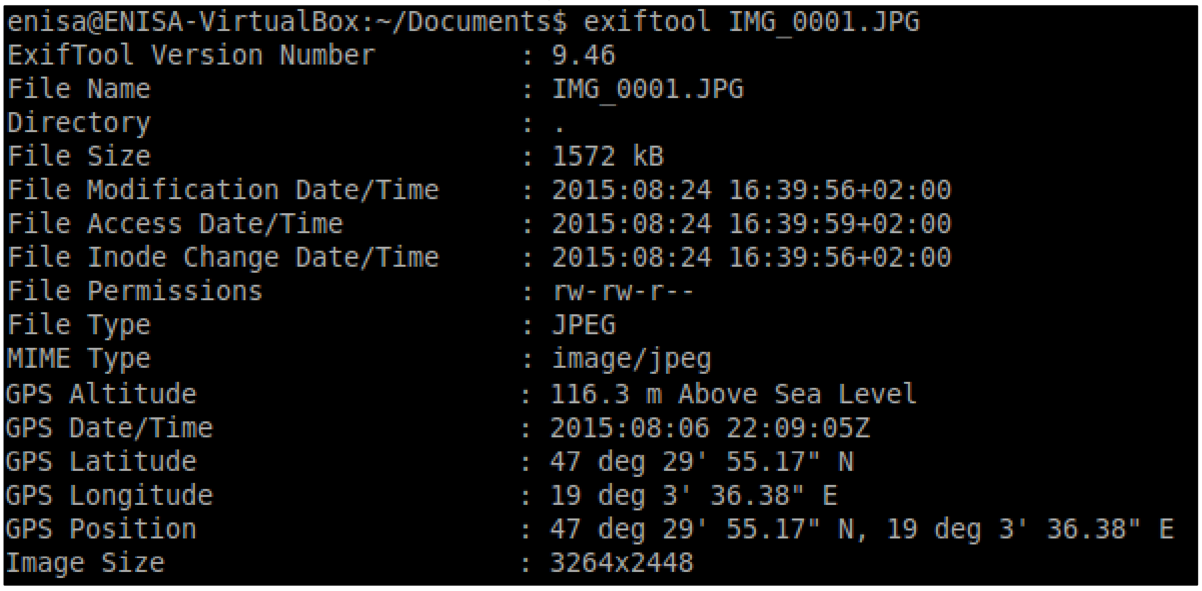
File: IMG_0004.JPG

Element type: -
Permissions: rw-r--r--
Data hash:
User id: 000001f5
Group id: 000001f5
Last modify time: 2015-08-18 16:13:35
Last access Time: 2015-08-18 16:13:36
Creation time: 2015-08-18 16:13:35
File Key (obfuscated file name): 9e3156b537de1cea17a389fd1c9faaa8cb870701
Flag: 4

Figure 69

4.9.3.4 You can view or even export simple JPG file from backup. Please export few JPG files from backup.

4.9.3.5 Let us take a look at JPG files which you exported by another tool called exiftool. This is a software which allows to read EXIF data. Try to locate GPS co-ordinates.



```
enisa@ENISA-VirtualBox:~/Documents$ exiftool IMG_0001.JPG
ExifTool Version Number         : 9.46
File Name                       : IMG_0001.JPG
Directory                       : .
File Size                       : 1572 kB
File Modification Date/Time     : 2015:08:24 16:39:56+02:00
File Access Date/Time           : 2015:08:24 16:39:59+02:00
File Inode Change Date/Time     : 2015:08:24 16:39:56+02:00
File Permissions                : rw-rw-r--
File Type                       : JPEG
MIME Type                       : image/jpeg
GPS Altitude                    : 116.3 m Above Sea Level
GPS Date/Time                   : 2015:08:06 22:09:05Z
GPS Latitude                    : 47 deg 29' 55.17" N
GPS Longitude                   : 19 deg 3' 36.38" E
GPS Position                    : 47 deg 29' 55.17" N, 19 deg 3' 36.38" E
Image Size                      : 3264x2448
```

Figure 70

4.9.3.6 Now you are able to see GPS position where this image file was presumably taken. Open any website that allows to search GPS position (e.g. https://maps.google.com) and the co-ordinates into search field. You need to change word "deg" to a degree symbol (i.e. °) with a combination of ALT + 248 on keyboard.

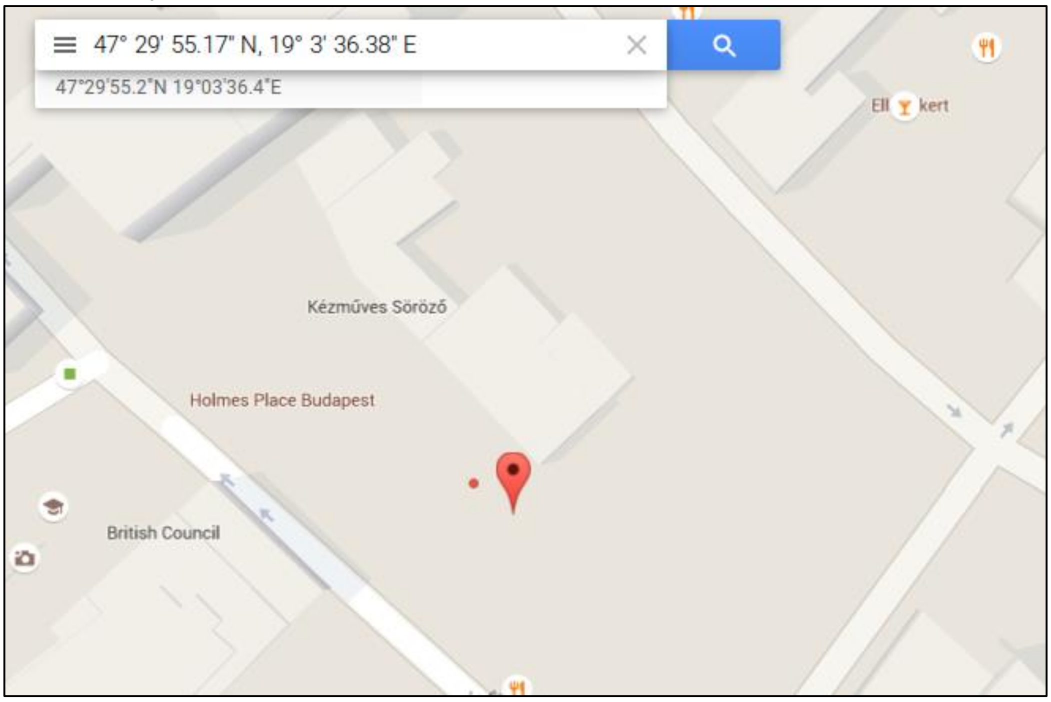
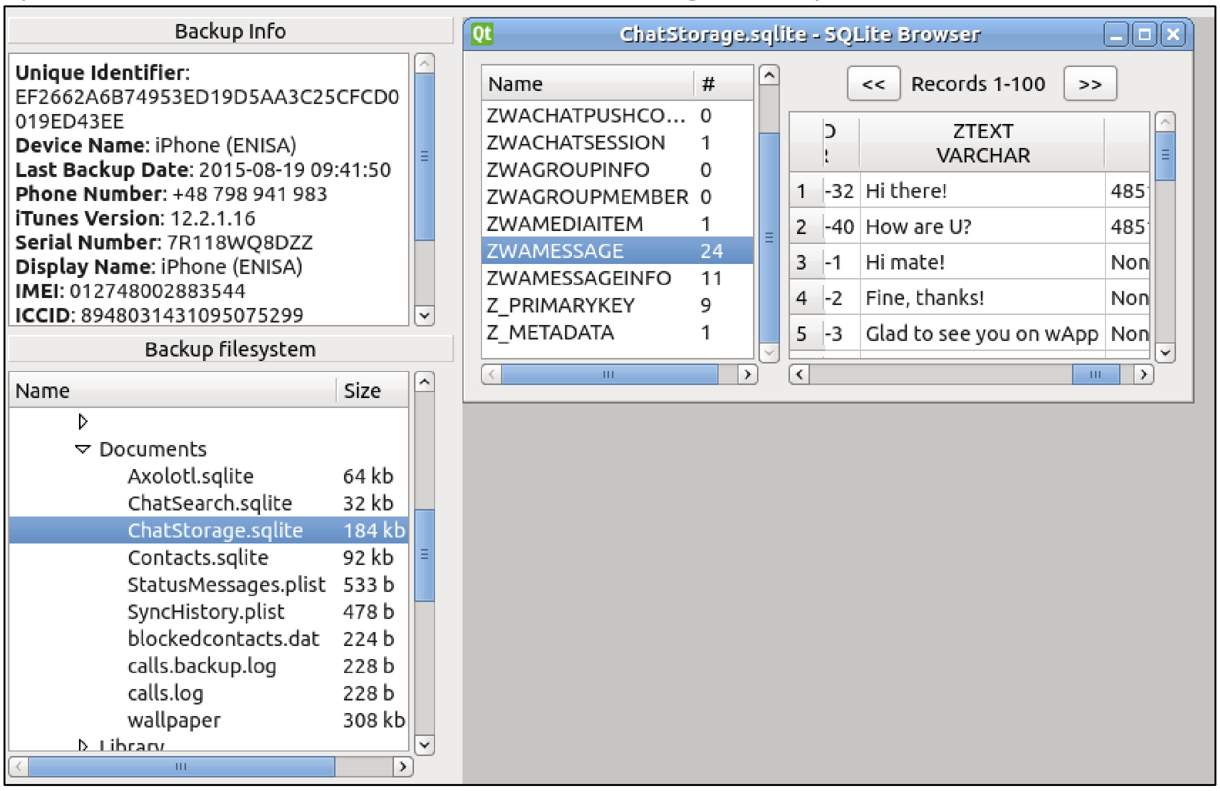


Figure 71

4.9.3.7 Now take a look into AppDomain. This is a list of installed applications. Locate folder Documents for application WhatsApp (net.wshatsapp.WhatsApp). Here you can find an SQLite databases used by WhatsApp instant messenger. Open database called ChatStorage.sqlite in SQLite viewer. Try to find ZWAMESSAGE table which contains messages history.


Backup Info

Unique Identifier: EF2662A6B74953ED19D5AA3C25CFCD0019ED43EE
Device Name: iPhone (ENISA)
Last Backup Date: 2015-08-19 09:41:50
Phone Number: +48 798 941 983
iTunes Version: 12.2.1.16
Serial Number: 7R118WQ8DZZ
Display Name: iPhone (ENISA)
IMEI: 012748002883544
ICCID: 8948031431095075299

Backup Filesystem

Name	Size
Documents	
Axolotl.sqlite	64 kb
ChatSearch.sqlite	32 kb
ChatStorage.sqlite	184 kb
Contacts.sqlite	92 kb
StatusMessages.plist	533 b
SyncHistory.plist	478 b
blockedcontacts.dat	224 b
calls.backup.log	228 b
calls.log	228 b
wallpaper	308 kb
Library	

ChatStorage.sqlite - SQLite Browser

Name	#
ZWACHATPUSHCO...	0
ZWACHATSESSION	1
ZWAGROUPINFO	0
ZWAGROUPMEMBER	0
ZWAMEDIAITEM	1
ZWAMESSAGE	24
ZWAMESSAGEINFO	11
Z_PRIMARYKEY	9
Z_METADATA	1

<< Records 1-100 >>

	D	ZTEXT VARCHAR	
1	-32	Hi there!	485
2	-40	How are U?	485
3	-1	Hi mate!	Non
4	-2	Fine, thanks!	Non
5	-3	Glad to see you on wApp	Non

Figure 72

4.10 Task 4.6: Brute-forcing Android encryption mechanisms

4.10.1 Introduction

In this task, the students will try the process of cracking the PIN used to encrypt Android device (Ice Cream Sandwich and Jelly Bean) using brute force methods.

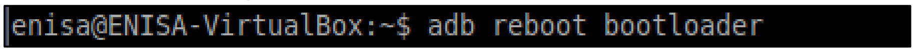
4.10.2 Details

Students will have to prepare a recovery partition for Android device. After that they will be able to download necessary files which will be used during cracking process.

4.10.3 Task walk-through

This section will show possible approaches to brute-force attacks against PIN-based encryption of Android devices. <u>The task requires students to use a physical device, since AVD emulators will not support fastboot mode</u>.

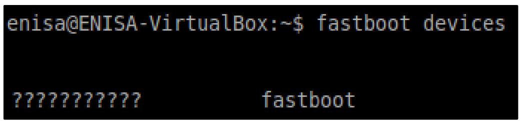
4.10.3.1 First you need to put the phone in recovery mode so that it can boot a custom recovery image. If the device already has a custom recovery image with root address installed, you can skip this step. There is an easy way to accomplish this with adb. If you have a device with adb enabled, simply connect it to PC (make sure you pass it through if running in a virtual machine), run a terminal and issue the following command:



```
enisa@ENISA-VirtualBox:~$ adb reboot bootloader
```

Figure 73

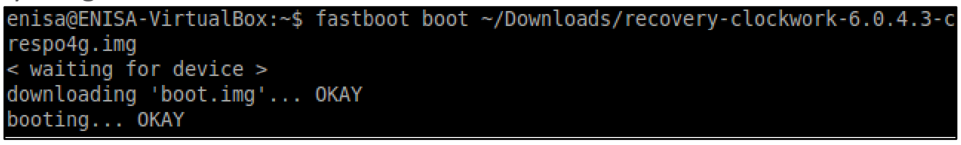
4.10.3.2 Next, boot the device from a rooted recovery image. For this purpose, a Clockwork Mod[90] image is used, but you can use any device-compatible recovery image with root and adb enabled. Please, note where you saved the recovery image. From a terminal, run fastboot and make sure you can communicate with the device.



```
enisa@ENISA-VirtualBox:~$ fastboot devices


???????????          fastboot
```

Figure 74

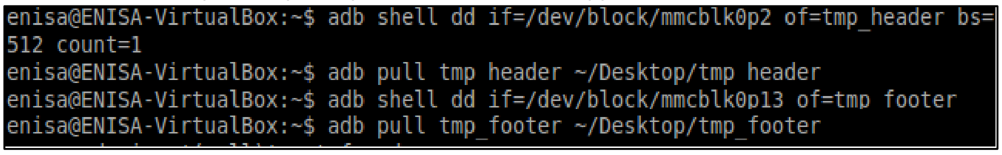
4.10.3.3 Now that you checked that you can communicate with the device over fastboot, boot it using the recovery image.



```
enisa@ENISA-VirtualBox:~$ fastboot boot ~/Downloads/recovery-clockwork-6.0.4.3-c
respo4g.img
< waiting for device >
downloading 'boot.img'... OKAY
booting... OKAY
```

Figure 75

4.10.3.4 Your device should be in a recovery mode now. Next, pull the needed necessary header and footer data so that you can brute-force the encryption PIN. Their location varies device by device so choose the steps for your particular device type.

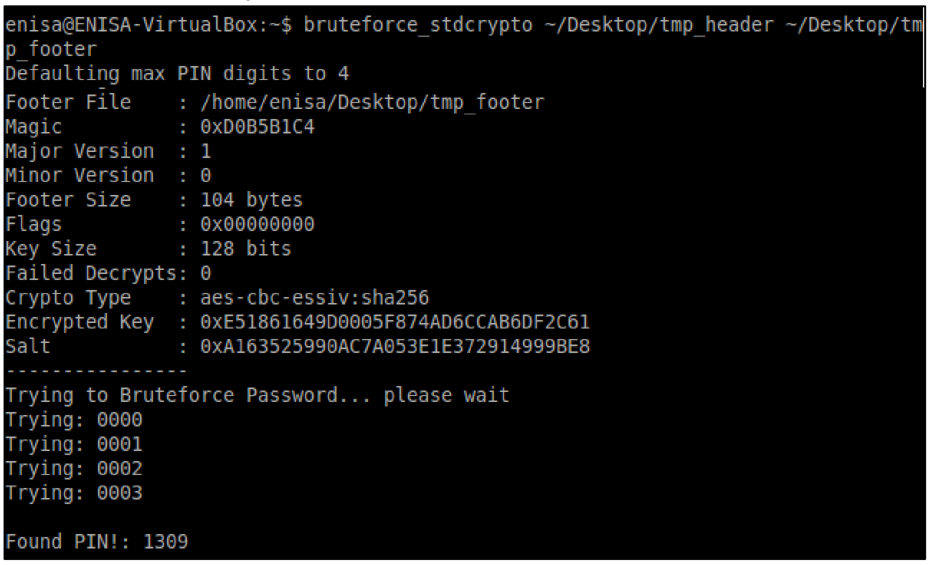


```
enisa@ENISA-VirtualBox:~$ adb shell dd if=/dev/block/mmcblk0p2 of=tmp_header bs=
512 count=1
enisa@ENISA-VirtualBox:~$ adb pull tmp_header ~/Desktop/tmp_header
enisa@ENISA-VirtualBox:~$ adb shell dd if=/dev/block/mmcblk0p13 of=tmp_footer
enisa@ENISA-VirtualBox:~$ adb pull tmp_footer ~/Desktop/tmp_footer
```

Figure 76

[90] ROM Manager: ROMs and Recovery Images, http://www.clockworkmod.com/rommanage, last accessed on: 2015-09-14

4.10.3.5 Now that you have everything you need you will run the Android Brute Force Encryption cracking program against the header and footer files. By default, a 4-digit numeric passcodes will be used but you can change the number of digits at your will remembering that the longer the PIN, the more time is necessary to brute-force it.



```
enisa@ENISA-VirtualBox:~$ bruteforce_stdcrypto ~/Desktop/tmp_header ~/Desktop/tm
p_footer
Defaulting max PIN digits to 4
Footer File    : /home/enisa/Desktop/tmp_footer
Magic          : 0xD0B5B1C4
Major Version  : 1
Minor Version  : 0
Footer Size    : 104 bytes
Flags          : 0x00000000
Key Size       : 128 bits
Failed Decrypts: 0
Crypto Type    : aes-cbc-essiv:sha256
Encrypted Key  : 0xE51861649D0005F874AD6CCAB6DF2C61
Salt           : 0xA163525990AC7A053E1E372914999BE8
----------------
Trying to Bruteforce Password... please wait
Trying: 0000
Trying: 0001
Trying: 0002
Trying: 0003

Found PIN!: 1309
```

Figure 77

After a while you will be able to see a PIN code.

5. Mobile network forensics

This exercise extends previously published ENISA training exercises (see prerequisites) with special requirements and approaches regarding mobile platforms and environments. The exercise starts with an introduction regarding the requirements to access mobile network traffic and some approaches in addressing these with an open-source software.

Following this section, the students will be given prepared samples of malware traffic captured with tcpdump[91] and mitmproxy[92] for analysis. After the exercise and accompanying studies the students should be aware of tools and techniques to build an environment to capture and analyse network traffic generated by mobile malware.

5.1 Introduction to accessing mobile traffic

Apart from network connections common with the traditional computer equipment (cable, WLAN), there is one type of a link, which has to be addressed in a specific way to be able to capture and analyse traffic.

Mobile network connectivity through technologies like GSM, UMTS, LTE is not commonly used in computer environments and thus has to be dealt with in a special way in a lab situation. One way is to use specialised commercial systems[93] available to law enforcement agencies. Alternatives are open-source implementations of management and data forwarding applications which, when combined with software defined radio (SDR) hardware, can be used to create a closed mobile network suitable for analysis.

This approach could be necessary when malware analyses its environment and activates its functionality only when connected to a mobile network.

The following list contains a set of possible free / low cost solutions which can be assembled to build research lab environments:

- Software: OpenLTE,[94] Osmocom,[95] OpenBTS, YateBTS[96]
- Hardware: DVB-T USB,[97] USRP[98] [99] [100]
- Network Traffic access: tcpdump, mitmproxy

The assembly of a lab environment is beyond the scope of this exercise.

[91] Tcpdump: network traffic sniffer, http://www.tcpdump.org/, last accessed on: 2015-09-14
[92] MitM Proxy: An interactive console program that allows traffic flows to be intercepted, inspected, modified and replayed, https://mitmproxy.org/, last accessed on: 2015-09-14
[93] Cellular Intercept and Cellular Monitoring technologies give Law Enforcement and Government Agencies a technological edge, http://www.cellularintercept.com/, last accessed on: 2015-09-14
[94] OpenLTE is an open source implementation of the 3GPP LTE specifications, http://openlte.sourceforge.net/, last accessed on: 2015-09-14
[95] The Osmocom project is a family of projects regarding Open source mobile communications, http://osmocom.org/, last accessed on: 2015-09-14
[96] YateBTS, http://www.yatebts.com/, last accessed on: 2015-09-14
[97] OsmoSDR is a 100% Free Software based small form-factor inexpensive SDR (Software Defined Radio) project, http://sdr.osmocom.org/trac/, last accessed on: 2015-09-14
[98] Ettus research, http://www.ettus.com/home, last accessed on: 2015-09-14
[99] Fairwaves, http://shop.fairwaves.co/UmTRX-transceiver-for-OpenBTS-and-Osmocom-OpenBSC, last accessed on: 2015-09-14
[100] Nuand, https://nuand.com/, last accessed on: 2015-09-14

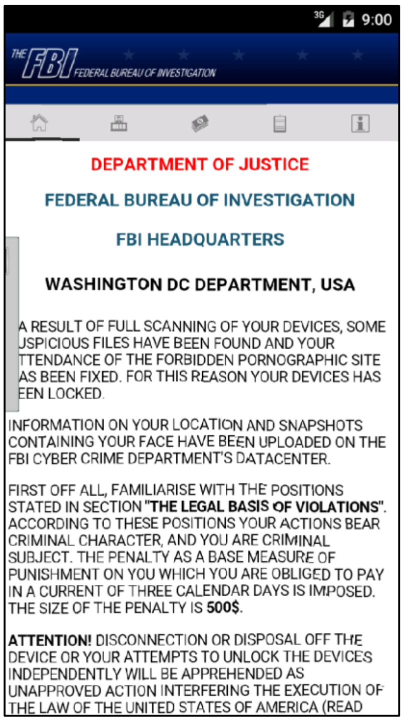
For both tasks the same environment for capturing network traffic has been used. It consisted of a Linux system with two network interfaces, one pointing to the internet and the other one towards the infected systems. On this system two applications were run to capture the traffic:

- mitmproxy in transparent mode
- tcpdump

5.1.1 Malware Information

5.1.1.1 Android

The chosen malware sample is a ransomware disguised as an Adobe Flash Player Update, purporting to be a video player app.[101] After the activation the malware shows the following screen to the user:



DEPARTMENT OF JUSTICE

FEDERAL BUREAU OF INVESTIGATION

FBI HEADQUARTERS

WASHINGTON DC DEPARTMENT, USA

A RESULT OF FULL SCANNING OF YOUR DEVICES, SOME USPICIOUS FILES HAVE BEEN FOUND AND YOUR TTENDANCE OF THE FORBIDDEN PORNOGRAPHIC SITE AS BEEN FIXED. FOR THIS REASON YOUR DEVICES HAS EEN LOCKED.

INFORMATION ON YOUR LOCATION AND SNAPSHOTS CONTAINING YOUR FACE HAVE BEEN UPLOADED ON THE FBI CYBER CRIME DEPARTMENT'S DATACENTER.

FIRST OFF ALL, FAMILIARISE WITH THE POSITIONS STATED IN SECTION **"THE LEGAL BASIS OF VIOLATIONS"**. ACCORDING TO THESE POSITIONS YOUR ACTIONS BEAR CRIMINAL CHARACTER, AND YOU ARE CRIMINAL SUBJECT. THE PENALTY AS A BASE MEASURE OF PUNISHMENT ON YOU WHICH YOU ARE OBLIGED TO PAY IN A CURRENT OF THREE CALENDAR DAYS IS IMPOSED. THE SIZE OF THE PENALTY IS **500$**.

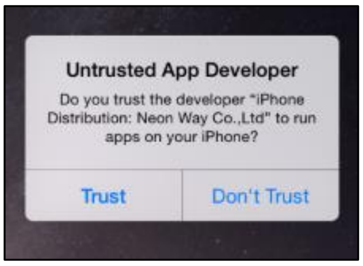
ATTENTION! DISCONNECTION OR DISPOSAL OFF THE DEVICE OR YOUR ATTEMPTS TO UNLOCK THE DEVICES INDEPENDENTLY WILL BE APPREHENDED AS UNAPPROVED ACTION INTERFERING THE EXECUTION OF THE LAW OF THE UNITED STATES OF AMERICA (READ

Figure 78: Message from malware appears on the screen

5.1.1.2 iOS

The most interesting part of the chosen iOS malware is the way of infection. The installer is delivered to the user when browsing an Adult Video site.[102] The developers apparently have access to Apple's Enterprise Developer program, which can be used to deliver applications to iOS devices without using the Apple App Store (see screenshot).

[101] Bitdefender: Android ransomware uses fake FBI porn warning, http://www.cbronline.com/news/mobility/security/bitdefender-android-ransomware-uses-fake-fbi-porn-warning-4585753, last accessed on: 2015-09-14
[102] Japanese one-click fraudsters target iOS users with malicious app delivered over the air, http://www.symantec.com/connect/blogs/japanese-one-click-fraudsters-target-ios-users-malicious-app-delivered-over-air, last accessed on: 2015-09-14


Untrusted App Developer

Do you trust the developer "iPhone Distribution: Neon Way Co.,Ltd" to run apps on your iPhone?

Figure 79: A user is asked if he / she trusts the application developer

After activation, the application would show the user a message that he has signed up with the adult video site and has agreed to pay a fee. At the time being, the server part of the application is not available.

5.2 Task 5.1: Analysing pcap data and proxy logs of Android.Trojan.SLocker.DZ

5.2.1 Introduction

In this task the students are given a set of files (PCAP, mitmproxy) created during observing activity by an Android device infected with the Android.Trojan.Slocker.DZ ransomware. The students will use Wireshark and the text editor of their choice to search for patterns indicating malicious behaviour and analyse and describe this.

5.2.2 Tools used

- Wireshark
- MITMProxy

5.2.3 Details

There are two files in the traces subdirectory of the exercise environment named:

F836F5C6267F13BF9F6109A6B8D79175.pcap and F836F5C6267F13BF9F6109A6B8D79175.log.

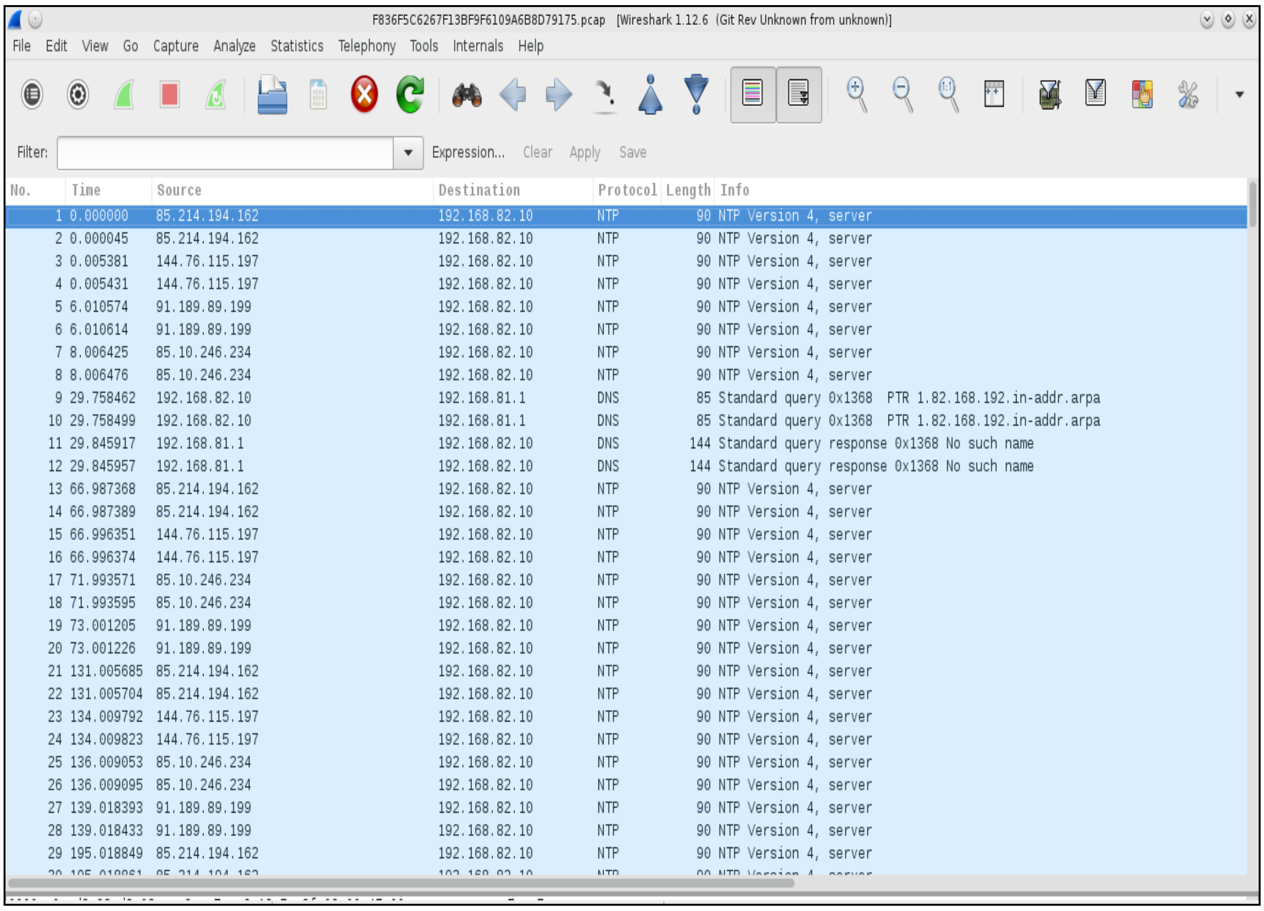
NO.	QUESTION	ANSWER
1	IP ADDRESS OF MALWARE SERVER?	148.251.154.104
2	PROTOCOL AND DATA FORMAT OF MALWARE COMMUNICATION?	HTTP, APPLICATION/JSON
3	MESSAGE SHOWN TO THE VICTIM?	"OMG!!! GUESS WHO'S ON A VIDEO HERE, YOU WILL NOT BELIEVE IT!!! DOWNLOAD THIS APPLICATION AND LOOK ON FIRST PAGE HTTP://CNNFOXNEWS.COM/"
4	CLASSIFICATION OF THE MALWARE?	RANSOMWARE
5	WHICH DATA IS TRANSFERRED?	PUBLIC IP, LONGITUDE, LATITUDE, CITY, COUNTRY

Table 5: Questions and Answers

5.2.4 Task walk-through

This section will contain possible ways to analyse the given information and identify the answers to the requests in the table:

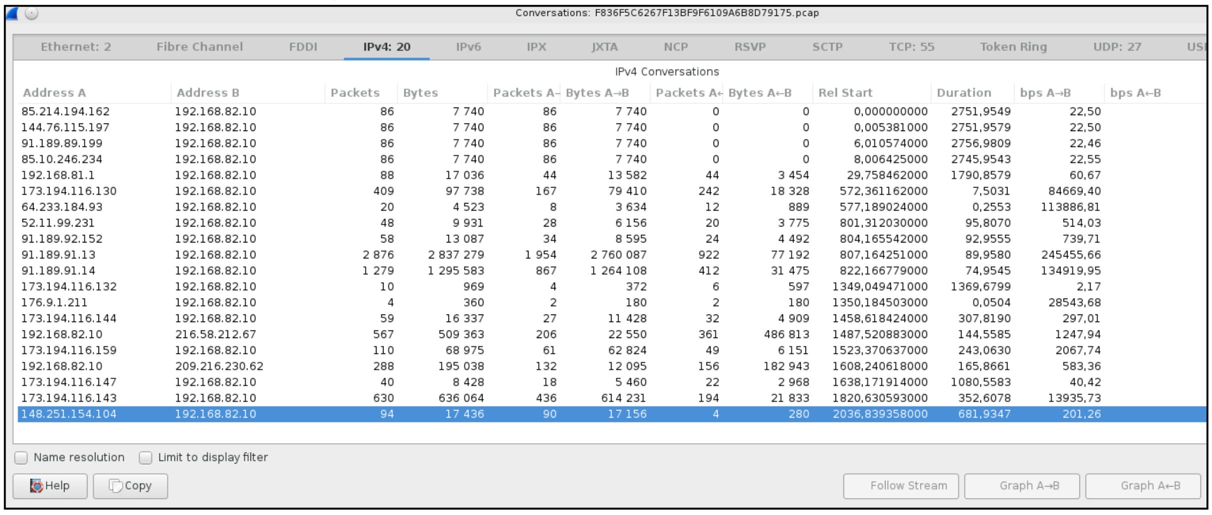
5.2.4.1 Open the PCAP file in Wireshark. After loading the PCAP file you will see the list of packets captured. Only a subset of these packets are part of the malware communication, to identify these we can use some of the tools Wireshark provides.


F836F5C6267F13BF9F6109A6B8D79175.pcap [Wireshark 1.12.6 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

```
No.  Time        Source           Destination    Protocol Length Info
   1 0.000000    85.214.194.162   192.168.82.10  NTP       90 NTP Version 4, server
   2 0.000045    85.214.194.162   192.168.82.10  NTP       90 NTP Version 4, server
   3 0.005381    144.76.115.197   192.168.82.10  NTP       90 NTP Version 4, server
   4 0.005431    144.76.115.197   192.168.82.10  NTP       90 NTP Version 4, server
   5 6.010574    91.189.89.199    192.168.82.10  NTP       90 NTP Version 4, server
   6 6.010614    91.189.89.199    192.168.82.10  NTP       90 NTP Version 4, server
   7 8.006425    85.10.246.234    192.168.82.10  NTP       90 NTP Version 4, server
   8 8.006476    85.10.246.234    192.168.82.10  NTP       90 NTP Version 4, server
   9 29.758462   192.168.82.10    192.168.81.1   DNS       85 Standard query 0x1368  PTR 1.82.168.192.in-addr.arpa
  10 29.758499   192.168.82.10    192.168.81.1   DNS       85 Standard query 0x1368  PTR 1.82.168.192.in-addr.arpa
  11 29.845917   192.168.81.1     192.168.82.10  DNS      144 Standard query response 0x1368 No such name
  12 29.845957   192.168.81.1     192.168.82.10  DNS      144 Standard query response 0x1368 No such name
  13 66.987368   85.214.194.162   192.168.82.10  NTP       90 NTP Version 4, server
  14 66.987389   85.214.194.162   192.168.82.10  NTP       90 NTP Version 4, server
  15 66.996351   144.76.115.197   192.168.82.10  NTP       90 NTP Version 4, server
  16 66.996374   144.76.115.197   192.168.82.10  NTP       90 NTP Version 4, server
  17 71.993571   85.10.246.234    192.168.82.10  NTP       90 NTP Version 4, server
  18 71.993595   85.10.246.234    192.168.82.10  NTP       90 NTP Version 4, server
  19 73.001205   91.189.89.199    192.168.82.10  NTP       90 NTP Version 4, server
  20 73.001226   91.189.89.199    192.168.82.10  NTP       90 NTP Version 4, server
  21 131.005685  85.214.194.162   192.168.82.10  NTP       90 NTP Version 4, server
  22 131.005704  85.214.194.162   192.168.82.10  NTP       90 NTP Version 4, server
  23 134.009792  144.76.115.197   192.168.82.10  NTP       90 NTP Version 4, server
  24 134.009823  144.76.115.197   192.168.82.10  NTP       90 NTP Version 4, server
  25 136.009053  85.10.246.234    192.168.82.10  NTP       90 NTP Version 4, server
  26 136.009095  85.10.246.234    192.168.82.10  NTP       90 NTP Version 4, server
  27 139.018393  91.189.89.199    192.168.82.10  NTP       90 NTP Version 4, server
  28 139.018433  91.189.89.199    192.168.82.10  NTP       90 NTP Version 4, server
  29 195.018849  85.214.194.162   192.168.82.10  NTP       90 NTP Version 4, server
```

Figure 80

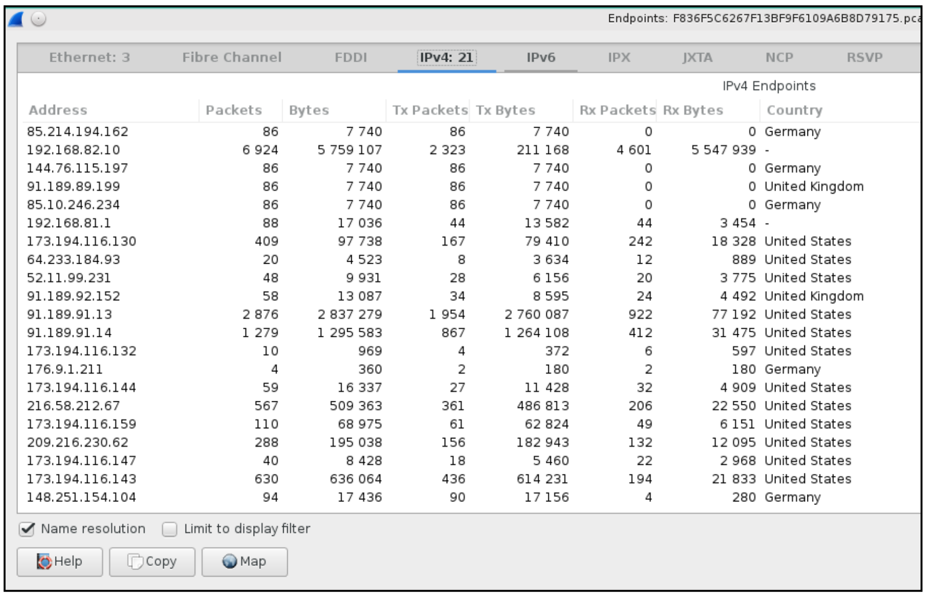
5.2.4.2 Open the list of conversations (Statistics → Conversations → Tab IPv4). In this list you will find all conversations contained in the network traffic dump accompanied by additional information regarding starting point in time, amount of data transferred and duration of the connection. This does not show the malicious traffic by itself but delivers an overview and some details regarding the information to be analysed.



Address A	Address B	Packets	Bytes	Packets A→B	Bytes A→B	Packets A←B	Bytes A←B	Rel Start	Duration	bps A→B	bps A←B
85.214.194.162	192.168.82.10	86	7 740	86	7 740	0	0	0,000000000	2751,9549	22,50	
144.76.115.197	192.168.82.10	86	7 740	86	7 740	0	0	0,005381000	2751,9579	22,50	
91.189.89.199	192.168.82.10	86	7 740	86	7 740	0	0	6,010574000	2756,9809	22,46	
85.10.246.234	192.168.82.10	86	7 740	86	7 740	0	0	8,006425000	2745,9543	22,55	
192.168.81.1	192.168.82.10	88	17 036	44	13 582	44	3 454	29,758462000	1790,8579	60,67	
173.194.116.130	192.168.82.10	409	97 738	167	79 410	242	18 328	572,361162000	7,5031	84669,40	
64.233.184.93	192.168.82.10	20	4 523	8	3 634	12	889	577,189024000	0,2553	113886,81	
52.11.99.231	192.168.82.10	48	9 931	28	6 156	20	3 775	801,312030000	95,8070	514,03	
91.189.92.152	192.168.82.10	58	13 087	34	8 595	24	4 492	804,165542000	92,9555	739,71	
91.189.91.13	192.168.82.10	2 876	2 837 279	1 954	2 760 087	922	77 192	807,164251000	89,9580	245455,66	
91.189.91.14	192.168.82.10	1 279	1 295 583	867	1 264 108	412	31 475	822,166779000	74,9545	134919,95	
173.194.116.132	192.168.82.10	10	969	4	372	6	597	1349,049471000	1369,6799	2,17	
176.9.1.211	192.168.82.10	4	360	2	180	2	180	1350,184503000	0,0504	28543,68	
173.194.116.144	192.168.82.10	59	16 337	27	11 428	32	4 909	1458,618424000	307,8190	297,01	
192.168.82.10	216.58.212.67	567	509 363	206	22 550	361	486 813	1487,520883000	144,5585	1247,94	
173.194.116.159	192.168.82.10	110	68 975	61	62 824	49	6 151	1523,370637000	243,0630	2067,74	
192.168.82.10	209.216.230.62	288	195 038	132	12 095	156	182 943	1608,240618000	165,8661	583,36	
173.194.116.147	192.168.82.10	40	8 428	18	5 460	22	2 968	1638,171914000	1080,5583	40,42	
173.194.116.143	192.168.82.10	630	636 064	436	614 231	194	21 833	1820,630593000	352,6078	13935,73	
148.251.154.104	192.168.82.10	94	17 436	90	17 156	4	280	2036,839358000	681,9347	201,26	

Figure 81

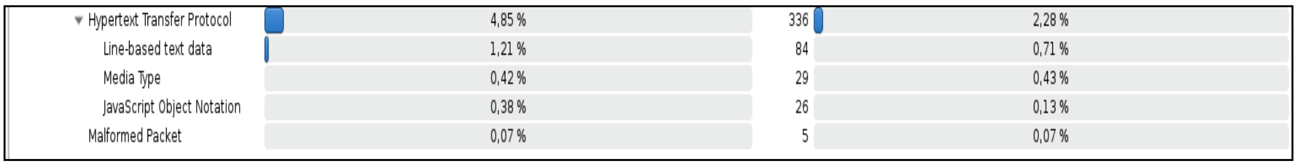
5.2.4.3 Open the list of endpoints (Statistics → Endpoints → Tab IPv4). In this list the geo-location of the conversation endpoints is added to the list.



Address	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes	Country
85.214.194.162	86	7 740	86	7 740	0	0	Germany
192.168.82.10	6 924	5 759 107	2 323	211 168	4 601	5 547 939	-
144.76.115.197	86	7 740	86	7 740	0	0	Germany
91.189.89.199	86	7 740	86	7 740	0	0	United Kingdom
85.10.246.234	86	7 740	86	7 740	0	0	Germany
192.168.81.1	88	17 036	44	13 582	44	3 454	-
173.194.116.130	409	97 738	167	79 410	242	18 328	United States
64.233.184.93	20	4 523	8	3 634	12	889	United States
52.11.99.231	48	9 931	28	6 156	20	3 775	United States
91.189.92.152	58	13 087	34	8 595	24	4 492	United Kingdom
91.189.91.13	2 876	2 837 279	1 954	2 760 087	922	77 192	United States
91.189.91.14	1 279	1 295 583	867	1 264 108	412	31 475	United States
173.194.116.132	10	969	4	372	6	597	United States
176.9.1.211	4	360	2	180	2	180	Germany
173.194.116.144	59	16 337	27	11 428	32	4 909	United States
216.58.212.67	567	509 363	361	486 813	206	22 550	United States
173.194.116.159	110	68 975	61	62 824	49	6 151	United States
209.216.230.62	288	195 038	156	182 943	132	12 095	United States
173.194.116.147	40	8 428	18	5 460	22	2 968	United States
173.194.116.143	630	636 064	436	614 231	194	21 833	United States
148.251.154.104	94	17 436	90	17 156	4	280	Germany

Figure 82

5.2.4.4 Open the Protocol Hierarchy (Statistics → Protocol Hierarchy). The previous approaches have given no clear indicator for malicious behaviour or even a hint for which connections should be inspected in more detail. Thus you have to try to dig deeper and find information regarding the protocols used in the dump. In this case you will find interesting information regarding JSON data which has been transmitted in clear-text.

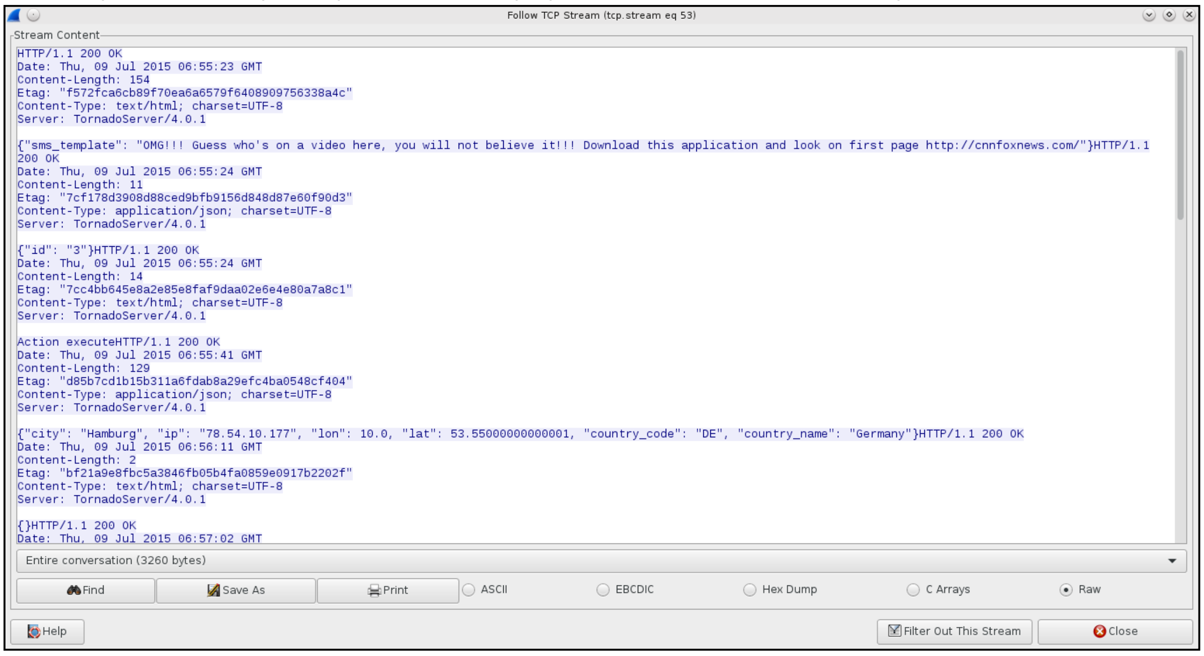


▾ Hypertext Transfer Protocol	4,85 %	336	2,28 %
Line-based text data	1,21 %	84	0,71 %
Media Type	0,42 %	29	0,43 %
JavaScript Object Notation	0,38 %	26	0,13 %
Malformed Packet	0,07 %	5	0,07 %

Figure 83

5.2.4.5 Apply a filter to the captured traffic (Right-click on the selected entry → Apply as Filter → Selected).

5.2.4.6 Use TCP stream analysis (Right-click → Follow TCP Stream). Using this feature provides a human-readable presentation of the HTTP traffic. Indicators of malicious traffic can be clearly identified, for example the deception phrase in the payload of the first server response.



Follow TCP Stream (tcp.stream eq 53)

Stream Content

```
HTTP/1.1 200 OK
Date: Thu, 09 Jul 2015 06:55:23 GMT
Content-Length: 154
Etag: "f572fca6cb89f70ea6a6579f6408909756338a4c"
Content-Type: text/html; charset=UTF-8
Server: TornadoServer/4.0.1

{"sms_template": "OMG!!! Guess who's on a video here, you will not believe it!!! Download this application and look on first page http://cnnfoxnews.com/"}HTTP/1.1
200 OK
Date: Thu, 09 Jul 2015 06:55:24 GMT
Content-Length: 11
Etag: "7cf178d3908d88ced9bfb9156d848d87e60f90d3"
Content-Type: application/json; charset=UTF-8
Server: TornadoServer/4.0.1

{"id": "3"}HTTP/1.1 200 OK
Date: Thu, 09 Jul 2015 06:55:24 GMT
Content-Length: 14
Etag: "7cc4bb645e8a2e85e8faf9daa02e6e4e80a7a8c1"
Content-Type: text/html; charset=UTF-8
Server: TornadoServer/4.0.1

Action executeHTTP/1.1 200 OK
Date: Thu, 09 Jul 2015 06:55:41 GMT
Content-Length: 129
Etag: "d85b7cd1b15b311a6fdab8a29efc4ba0548cf404"
Content-Type: application/json; charset=UTF-8
Server: TornadoServer/4.0.1

{"city": "Hamburg", "ip": "78.54.10.177", "lon": 10.0, "lat": 53.55000000000001, "country_code": "DE", "country_name": "Germany"}HTTP/1.1 200 OK
Date: Thu, 09 Jul 2015 06:56:11 GMT
Content-Length: 2
Etag: "bf21a9e8fbc5a3846fb05b4fa0859e0917b2202f"
Content-Type: text/html; charset=UTF-8
Server: TornadoServer/4.0.1

{}HTTP/1.1 200 OK
Date: Thu, 09 Jul 2015 06:57:02 GMT
```

Entire conversation (3260 bytes)

Find | Save As | Print | ASCII | EBCDIC | Hex Dump | C Arrays | [✓ Raw]

Help | Filter Out This Stream | Close

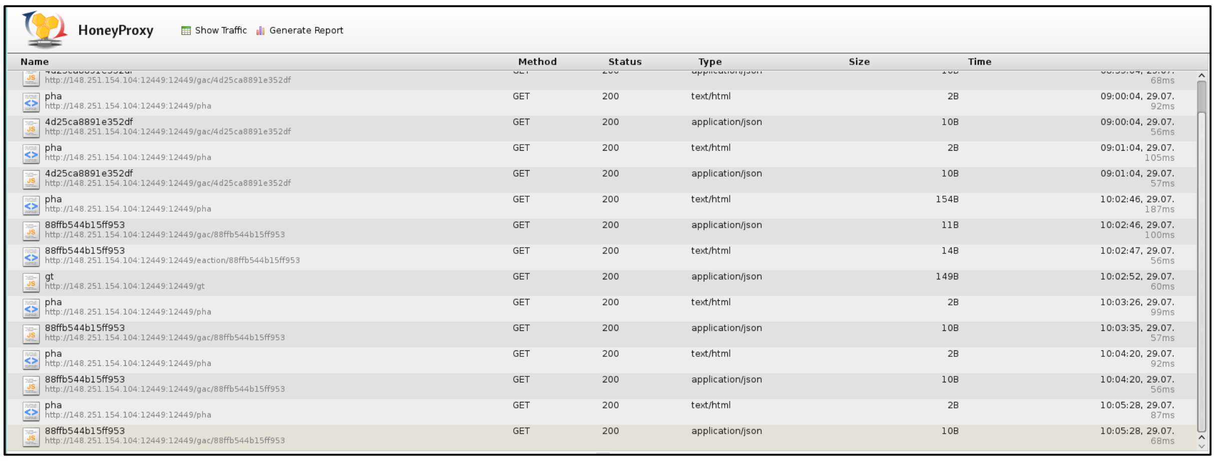
Figure 84

5.2.5 Task walk-through with mitmproxy logs

For the creation of the following screenshots Honeyproxy[103] was used. The project is based on MITMProxy and creates a web interface to inspect and analyse the traffic captured. Unfortunately it is not under active development as of the time of preparation of this exercise.

[103] HoneyProxy: A man-in-the-middle SSL proxy & traffic analyser, http://honeyproxy.org/, last accessed on: 2015-09-14

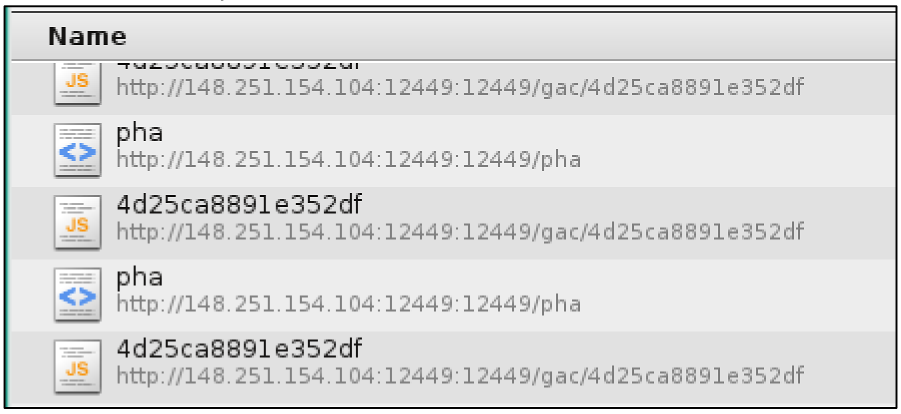
5.2.5.1 Overview of the web interface and the captured data.



Name	Method	Status	Type	Size	Time
http://148.251.154.104:12449:12449/gac/4d25ca8891e352df					68ms
pha http://148.251.154.104:12449:12449/pha	GET	200	text/html	2B	09:00:04, 29.07. 92ms
4d25ca8891e352df http://148.251.154.104:12449:12449/gac/4d25ca8891e352df	GET	200	application/json	10B	09:00:04, 29.07. 56ms
pha http://148.251.154.104:12449:12449/pha	GET	200	text/html	2B	09:01:04, 29.07. 105ms
4d25ca8891e352df http://148.251.154.104:12449:12449/gac/4d25ca8891e352df	GET	200	application/json	10B	09:01:04, 29.07. 57ms
pha http://148.251.154.104:12449:12449/pha	GET	200	text/html	154B	10:02:46, 29.07. 187ms
88ffb544b15ff953 http://148.251.154.104:12449:12449/gac/88ffb544b15ff953	GET	200	application/json	11B	10:02:46, 29.07. 100ms
88ffb544b15ff953 http://148.251.154.104:12449:12449/eaction/88ffb544b15ff953	GET	200	text/html	14B	10:02:47, 29.07. 56ms
gt http://148.251.154.104:12449:12449/gt	GET	200	application/json	149B	10:02:52, 29.07. 60ms
pha http://148.251.154.104:12449:12449/pha	GET	200	text/html	2B	10:03:26, 29.07. 99ms
88ffb544b15ff953 http://148.251.154.104:12449:12449/gac/88ffb544b15ff953	GET	200	application/json	10B	10:03:35, 29.07. 57ms
pha http://148.251.154.104:12449:12449/pha	GET	200	text/html	2B	10:04:20, 29.07. 92ms
88ffb544b15ff953 http://148.251.154.104:12449:12449/gac/88ffb544b15ff953	GET	200	application/json	10B	10:04:20, 29.07. 56ms
pha http://148.251.154.104:12449:12449/pha	GET	200	text/html	2B	10:05:28, 29.07. 87ms
88ffb544b15ff953 http://148.251.154.104:12449:12449/gac/88ffb544b15ff953	GET	200	application/json	10B	10:05:28, 29.07. 68ms

Figure 85

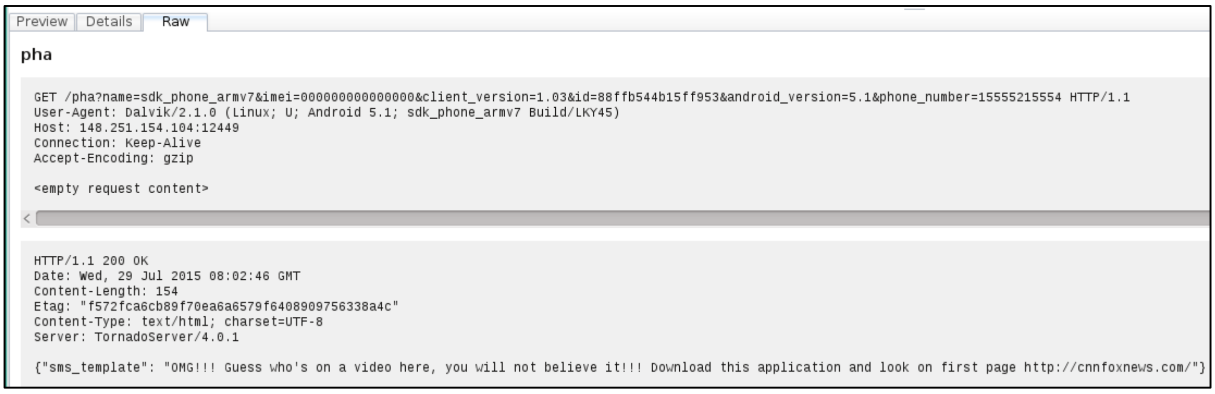
5.2.5.2 Close view of the malware requests.



Name
http://148.251.154.104:12449:12449/gac/4d25ca8891e352df
pha http://148.251.154.104:12449:12449/pha
4d25ca8891e352df http://148.251.154.104:12449:12449/gac/4d25ca8891e352df
pha http://148.251.154.104:12449:12449/pha
4d25ca8891e352df http://148.251.154.104:12449:12449/gac/4d25ca8891e352df

Figure 86

5.2.5.3 Request showing the message displayed to the victim.



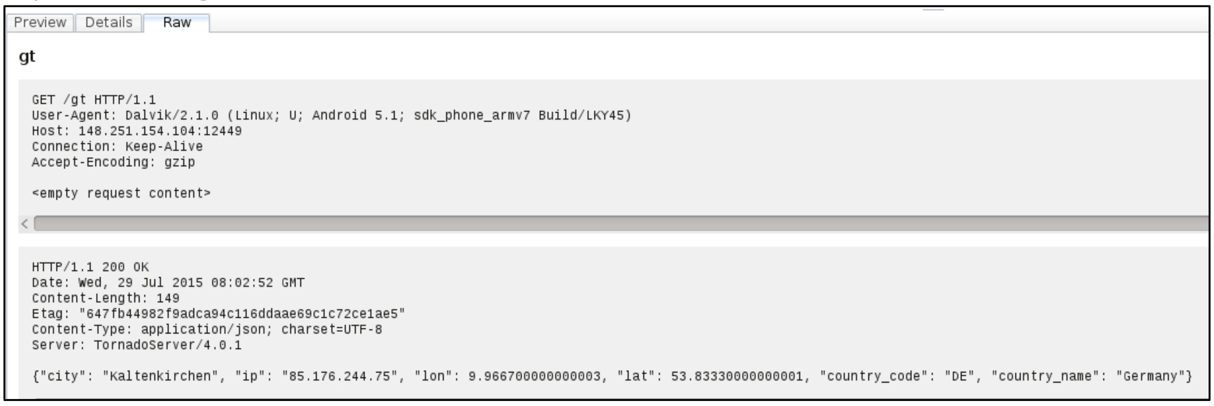
pha

```
GET /pha?name=sdk_phone_armv7&imei=000000000000000&client_version=1.03&id=88ffb544b15ff953&android_version=5.1&phone_number=15555215554 HTTP/1.1
User-Agent: Dalvik/2.1.0 (Linux; U; Android 5.1; sdk_phone_armv7 Build/LKY45)
Host: 148.251.154.104:12449
Connection: Keep-Alive
Accept-Encoding: gzip

<empty request content>
```

```
HTTP/1.1 200 OK
Date: Wed, 29 Jul 2015 08:02:46 GMT
Content-Length: 154
Etag: "f572fca6cb89f70ea6a6579f6408909756338a4c"
Content-Type: text/html; charset=UTF-8
Server: TornadoServer/4.0.1

{"sms_template": "OMG!!! Guess who's on a video here, you will not believe it!!! Download this application and look on first page http://cnnfoxnews.com/"}
```

Figure 87

5.2.5.4 Request showing the transmitted information.



```
GET /gt HTTP/1.1
User-Agent: Dalvik/2.1.0 (Linux; U; Android 5.1; sdk_phone_armv7 Build/LKY45)
Host: 148.251.154.104:12449
Connection: Keep-Alive
Accept-Encoding: gzip

<empty request content>
```

```
HTTP/1.1 200 OK
Date: Wed, 29 Jul 2015 08:02:52 GMT
Content-Length: 149
Etag: "647fb44982f9adca94c116ddaae69c1c72ce1ae5"
Content-Type: application/json; charset=UTF-8
Server: TornadoServer/4.0.1

{"city": "Kaltenkirchen", "ip": "85.176.244.75", "lon": 9.966700000000003, "lat": 53.83330000000001, "country_code": "DE", "country_name": "Germany"}
```

Figure 88

5.3 Task 5.2: Analysing pcap data and proxy logs of iOS.Oneclickfraud

5.3.1 Introduction

In this task the students are given a set of files (PCAP, mitmproxy) created during observing activity by an iOS device infected with the iOS.Oneclickfraud malware. The students will use Wireshark and the text editor of their choice to search for patterns indicating malicious behaviour and analyse and describe these.

5.3.2 Tools

- Wireshark

5.3.3 Details

There are two files in the traces subdirectory of the exercise environment named:

71972F763EB5EAEB87681D2615E9E68E.pcap and 71972F763EB5EAEB87681D2615E9E68E.log.

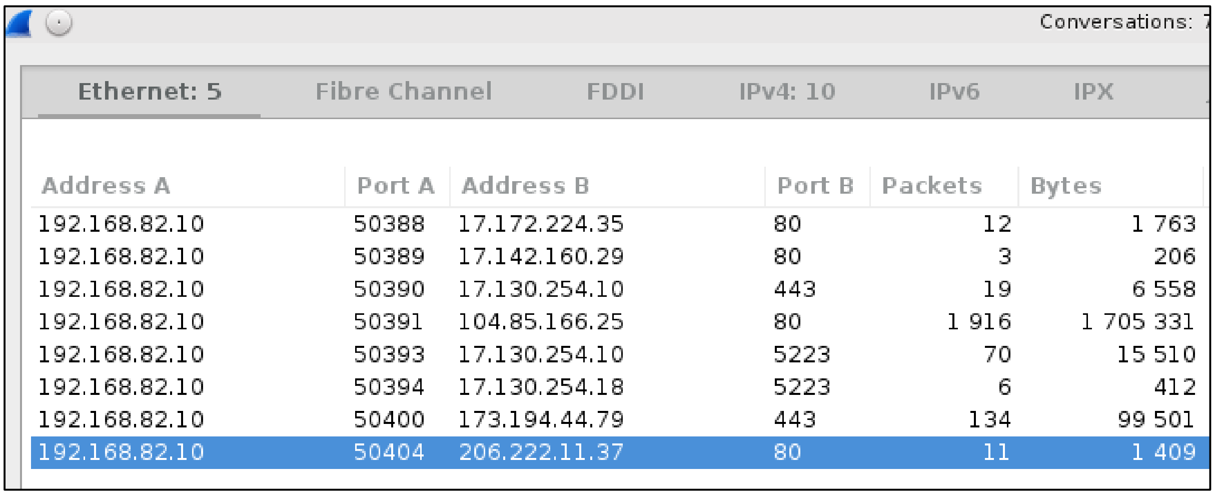
NO.	QUESTION	ANSWER
1	IP ADDRESS / HOST NAME OF MALWARE SERVER?	206.222.11.37, EROEROOU.COM
2	PROTOCOL OF MALWARE COMMUNICATION?	HTTP
3	URI CLIENT ACCESSES?	/APP/INIT
4	RESPOND OF THE SERVER?	HTTP ERROR CODE 404
5	USER-AGENT OF THE CLIENT?	EROEROMOVIE/1 CFNETWORK/711.1.12 DARWIN/14.0.0

Table 6: Questions and Answers

5.3.4 Test walk-through

The general approach to identify the malign traffic in the PCAP file is identical to Task 5.1. Following we will show the screenshots unique to Task 5.2. There is no walk-through with proxy logs as the server is not responding to the malware requests.

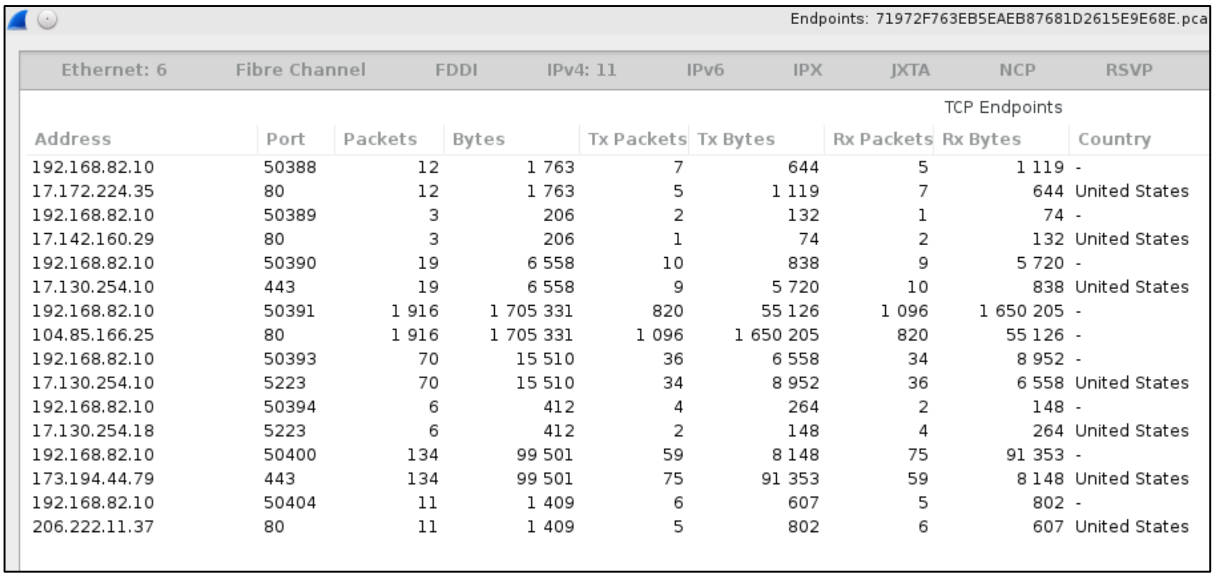
5.3.4.1 List of conversations.



Address A	Port A	Address B	Port B	Packets	Bytes
192.168.82.10	50388	17.172.224.35	80	12	1 763
192.168.82.10	50389	17.142.160.29	80	3	206
192.168.82.10	50390	17.130.254.10	443	19	6 558
192.168.82.10	50391	104.85.166.25	80	1 916	1 705 331
192.168.82.10	50393	17.130.254.10	5223	70	15 510
192.168.82.10	50394	17.130.254.18	5223	6	412
192.168.82.10	50400	173.194.44.79	443	134	99 501
192.168.82.10	50404	206.222.11.37	80	11	1 409

Figure 89

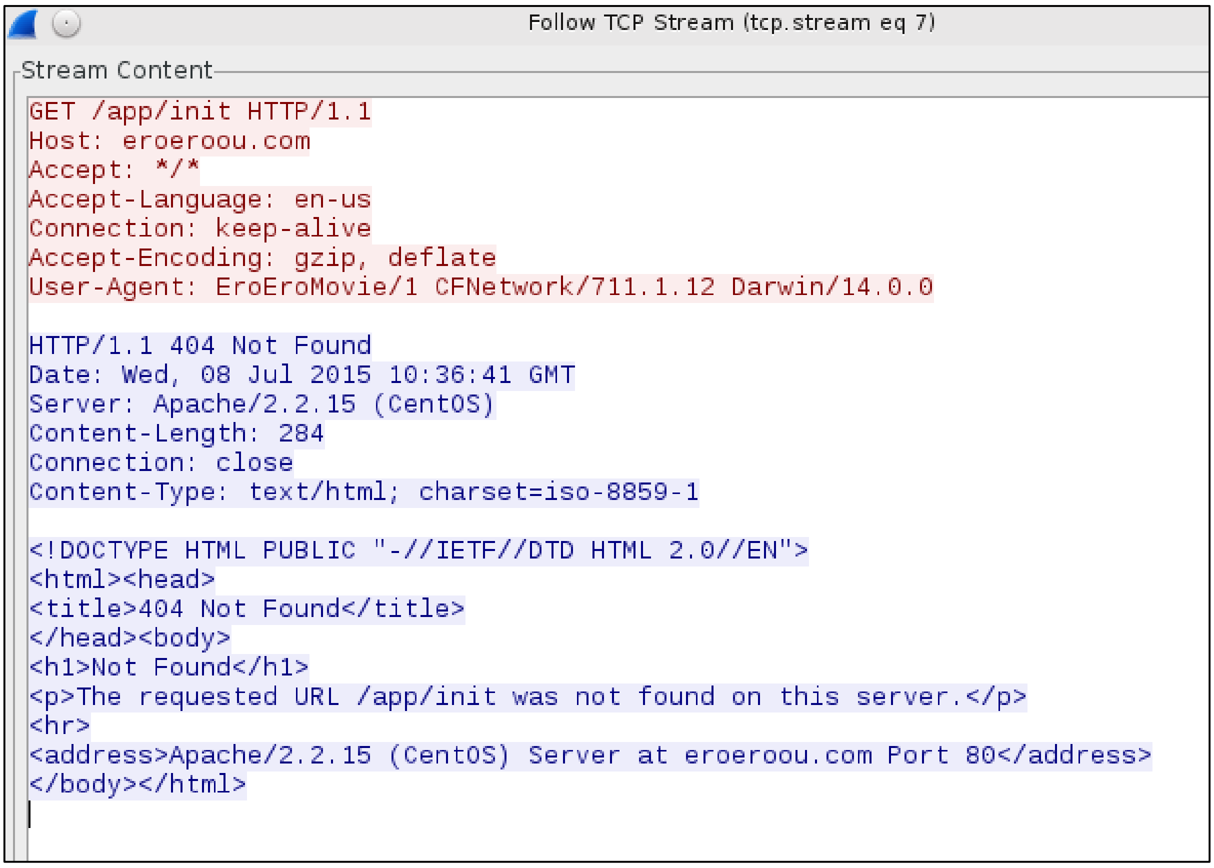
5.3.4.2 List of endpoints.



Address	Port	Packets	Bytes	Tx Packets	Tx Bytes	Rx Packets	Rx Bytes	Country
192.168.82.10	50388	12	1 763	7	644	5	1 119	-
17.172.224.35	80	12	1 763	5	1 119	7	644	United States
192.168.82.10	50389	3	206	2	132	1	74	-
17.142.160.29	80	3	206	1	74	2	132	United States
192.168.82.10	50390	19	6 558	10	838	9	5 720	-
17.130.254.10	443	19	6 558	9	5 720	10	838	United States
192.168.82.10	50391	1 916	1 705 331	820	55 126	1 096	1 650 205	-
104.85.166.25	80	1 916	1 705 331	1 096	1 650 205	820	55 126	-
192.168.82.10	50393	70	15 510	36	6 558	34	8 952	-
17.130.254.10	5223	70	15 510	34	8 952	36	6 558	United States
192.168.82.10	50394	6	412	4	264	2	148	-
17.130.254.18	5223	6	412	2	148	4	264	United States
192.168.82.10	50400	134	99 501	59	8 148	75	91 353	-
173.194.44.79	443	134	99 501	75	91 353	59	8 148	United States
192.168.82.10	50404	11	1 409	6	607	5	802	-
206.222.11.37	80	11	1 409	5	802	6	607	United States

Figure 90

5.3.4.3 TCP Stream.



```
GET /app/init HTTP/1.1
Host: eroeroou.com
Accept: */*
Accept-Language: en-us
Connection: keep-alive
Accept-Encoding: gzip, deflate
User-Agent: EroEroMovie/1 CFNetwork/711.1.12 Darwin/14.0.0

HTTP/1.1 404 Not Found
Date: Wed, 08 Jul 2015 10:36:41 GMT
Server: Apache/2.2.15 (CentOS)
Content-Length: 284
Connection: close
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /app/init was not found on this server.</p>
<hr>
<address>Apache/2.2.15 (CentOS) Server at eroeroou.com Port 80</address>
</body></html>
```

Figure 91

6. Mobile malware reverse engineering

In this exercise the task will be to analyse malicious applications developed for mobile platforms (Android, iOS) and use a variety of tools to identify information leading to the development of countermeasures. Extracting the applications from a mobile device will not be part of this exercise.

6.1 Introduction to special requirements in mobile malware

We will demonstrate the analysis of two mobile malware applications (Android.Trojan.SLocker.DZ for Android and iOS.Oneclickfraud) using a couple of publically available tools and make the students acquainted with them.

6.1.1 Tools

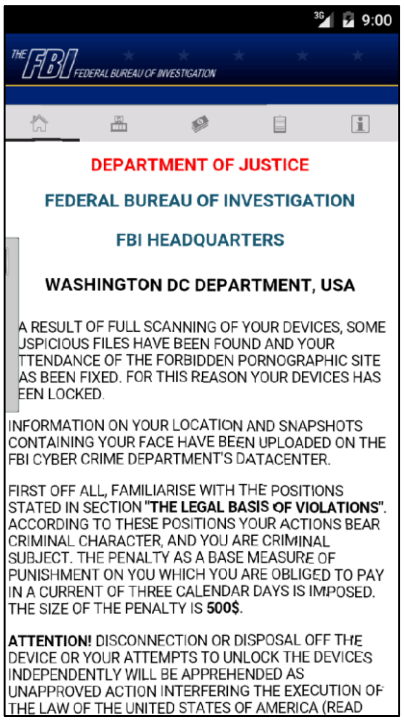
- Andrubis[104] – according to the project's website: "Andrubis executes Android apps in a sandbox and provides a detailed report on their behaviour, including file access, network access, crypto operations, dynamic code loading and information leaks".
- AndroGuard[105] – a tool for "reverse engineering, Malware and goodware analysis of Android applications".
- apktool[106] – a tool for "reverse engineering of 3rd party, closed, binary Android apps".
- Text editor of choice.
- class-dump-z[107] – a tool for extraction of Objective-C class interfaces.
- Hopper for iOS applications[108] – is a "reverse engineering tool for OS X and Linux, that lets disassemble, decompile and debug 32/64bits Intel Mac, Linux, Windows and iOS executables".

6.1.2 Malware Information

6.1.2.1 Android

The chosen malware sample is a ransomware disguised as an Adobe Flash Player Update, purporting to be a video player application.[109] After the activation the malware shows the following screen to the user:

[104] Andrubis: A Tool for Analysing Unknown Android Applications, http://blog.iseclab.org/2012/06/04/andrubis-a-tool-for-analysing-unknown-android-applications-2/ last accessed on: 2015-09-14
[105] Reverse engineering, Malware and goodware analysis of Android applications , https://github.com/androguard/androguard, last accessed on: 2015-09-14
[106] Apktool: A tool for reverse engineering Android apk files, https://github.com/iBotPeaches/Apktool, last accessed on: 2015-09-14
[107] Class-dump-z: Extracting class interface for Objective-C, https://code.google.com/p/networkpx/wiki/class_dump_z last accessed on: 2015-09-14
[108] The OS X and Linux Disassembler, http://www.hopperapp.com/, last accessed on: 2015-09-14
[109] Bitdefender: Android ransomware uses fake FBI porn warning, http://www.cbronline.com/news/mobility/security/bitdefender-android-ransomware-uses-fake-fbi-porn-warning-4585753, last accessed on: 2015-09-14



DEPARTMENT OF JUSTICE

FEDERAL BUREAU OF INVESTIGATION

FBI HEADQUARTERS

WASHINGTON DC DEPARTMENT, USA

A RESULT OF FULL SCANNING OF YOUR DEVICES, SOME USPICIOUS FILES HAVE BEEN FOUND AND YOUR TTENDANCE OF THE FORBIDDEN PORNOGRAPHIC SITE AS BEEN FIXED. FOR THIS REASON YOUR DEVICES HAS EEN LOCKED.

INFORMATION ON YOUR LOCATION AND SNAPSHOTS CONTAINING YOUR FACE HAVE BEEN UPLOADED ON THE FBI CYBER CRIME DEPARTMENT'S DATACENTER.

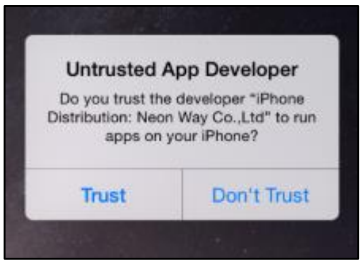
FIRST OFF ALL, FAMILIARISE WITH THE POSITIONS STATED IN SECTION **"THE LEGAL BASIS OF VIOLATIONS"**. ACCORDING TO THESE POSITIONS YOUR ACTIONS BEAR CRIMINAL CHARACTER, AND YOU ARE CRIMINAL SUBJECT. THE PENALTY AS A BASE MEASURE OF PUNISHMENT ON YOU WHICH YOU ARE OBLIGED TO PAY IN A CURRENT OF THREE CALENDAR DAYS IS IMPOSED. THE SIZE OF THE PENALTY IS **500$**.

ATTENTION! DISCONNECTION OR DISPOSAL OFF THE DEVICE OR YOUR ATTEMPTS TO UNLOCK THE DEVICES INDEPENDENTLY WILL BE APPREHENDED AS UNAPPROVED ACTION INTERFERING THE EXECUTION OF THE LAW OF THE UNITED STATES OF AMERICA (READ

Figure 92: Message from malware appears on the screen

6.1.2.2 iOS

The most interesting part of the chosen iOS malware is the method of the infection. The installer is delivered to the user when browsing an Adult Video site.[110] The developers apparently seem to have access to Apple's Enterprise Developer program, which can be used to deliver applications to iOS devices without using the Apple App Store (see Screenshot).



Untrusted App Developer

Do you trust the developer "iPhone Distribution: Neon Way Co.,Ltd" to run apps on your iPhone?

Figure 93: A user is asked if he / she trusts the application developer

After activation the application shows the user a message that he has signed up with the adult video site and has agreed to pay a fee. At the time being the server part of the application is not available any more.

[110] Japanese one-click fraudsters target iOS users with malicious app delivered over the air, http://www.symantec.com/connect/blogs/japanese-one-click-fraudsters-target-ios-users-malicious-app-delivered-over-air, last accessed on: 2015-09-14

6.2 Task 6.1: Analysing Android.Trojan.SLocker.DZ

6.2.1 Introduction

In this task the students will analyse an Android.Trojan.SLocker.DZ APK file. They will have to answer a couple of questions which will lead them to the identification of various characteristics of this trojan malware.

6.2.2 Tools

- AndroGuard
- apktool

6.2.3 Details

In the exercise directory students will find an APK file F836F5C6267F13BF9F6109A6B8D79175.apk. For the analysis of this file they can use the pre-installed AndroGuard, apktool and the text editor of their choice (there are several available on the system). In the next section students will find questions they have to answer during the analysis to identify the behaviour of the application.

No.	Question	Answer
1	Which permissions does the malware request?	SEND_SMS, SYSTEM_ALERT_WINDOW, RECEIVE_BOOT_COMPLETED, INTERNET, ACCESS_NETWORK_STATE, READ_PHONE_STATE, RECEIVE_SMS, READ_SMS, READ_CONTACTS, CAMERA, WRITE_EXTERNAL_STORAGE, WAKE_LOCK, READ_HISTORY_BOOKMARKS
2	Package name?	com.adobe.videoprayer
3	Intent-filters?	MAIN, LAUNCHER, BOOT_COMPLETED, SMS_RECEIVED
4	Which organisation is the pretended source?	FBI
5	Class with network connection?	AbsRequest
6	Supposed payment method?	PayPal card
7	Malware classification?	Ransomware

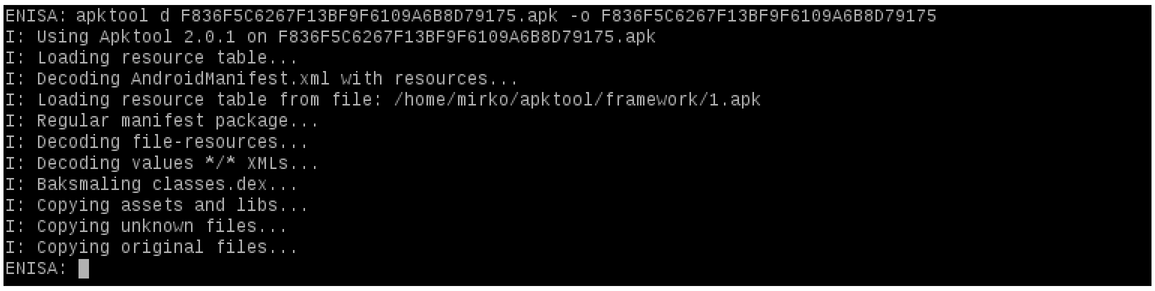
Table 7: Questions and Answers

6.2.4 Task walk-through

In this section, we will walk through a possible approach to analyse the malware and extract requested information.

6.2.4.1 Decode the APK file with the following command:

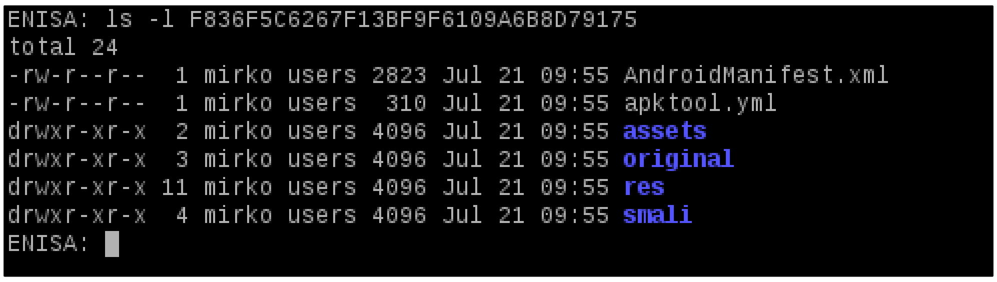
apktool d F836F5C6267F13BF9F6109A6B8D79175.apk -o F836F5C6267F13BF9F6109A6B8D79175



```
ENISA: apktool d F836F5C6267F13BF9F6109A6B8D79175.apk -o F836F5C6267F13BF9F6109A6B8D79175
I: Using Apktool 2.0.1 on F836F5C6267F13BF9F6109A6B8D79175.apk
I: Loading resource table...
I: Decoding AndroidManifest.xml with resources...
I: Loading resource table from file: /home/mirko/apktool/framework/1.apk
I: Regular manifest package...
I: Decoding file-resources...
I: Decoding values */* XMLs...
I: Baksmaling classes.dex...
I: Copying assets and libs...
I: Copying unknown files...
I: Copying original files...
ENISA: 
```

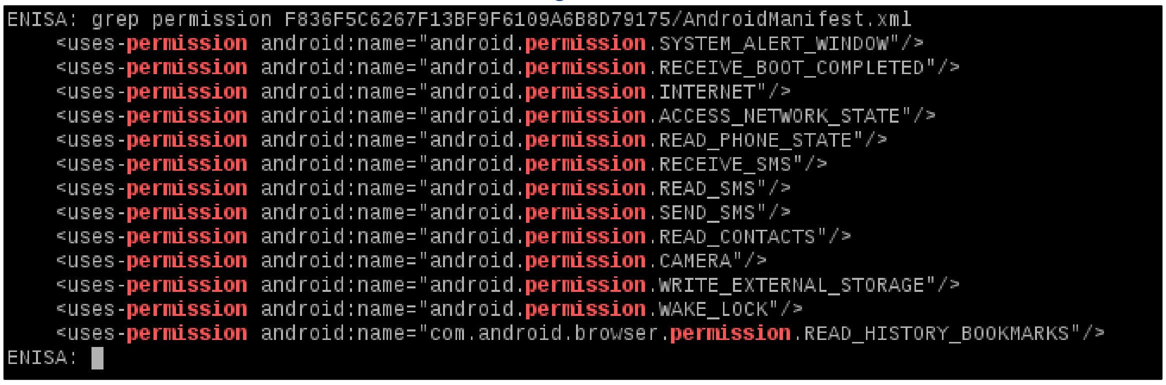
Figure 94

6.2.4.2 Search for permissions in the AndroidManifest:
grep permission F836F5C6267F13BF9F6109A6B8D79175/AndroidManifest.xml



```
ENISA: ls -l F836F5C6267F13BF9F6109A6B8D79175
total 24
-rw-r--r--  1 mirko users 2823 Jul 21 09:55 AndroidManifest.xml
-rw-r--r--  1 mirko users  310 Jul 21 09:55 apktool.yml
drwxr-xr-x  2 mirko users 4096 Jul 21 09:55 assets
drwxr-xr-x  3 mirko users 4096 Jul 21 09:55 original
drwxr-xr-x 11 mirko users 4096 Jul 21 09:55 res
drwxr-xr-x  4 mirko users 4096 Jul 21 09:55 smali
ENISA: 
```

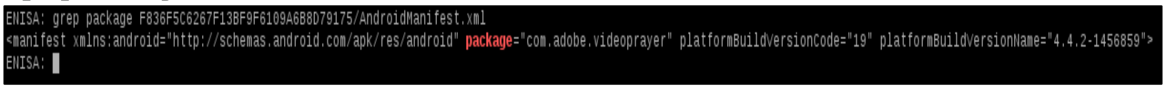
Figure 95



```
ENISA: grep permission F836F5C6267F13BF9F6109A6B8D79175/AndroidManifest.xml
    <uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW"/>
    <uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED"/>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
    <uses-permission android:name="android.permission.READ_PHONE_STATE"/>
    <uses-permission android:name="android.permission.RECEIVE_SMS"/>
    <uses-permission android:name="android.permission.READ_SMS"/>
    <uses-permission android:name="android.permission.SEND_SMS"/>
    <uses-permission android:name="android.permission.READ_CONTACTS"/>
    <uses-permission android:name="android.permission.CAMERA"/>
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
    <uses-permission android:name="android.permission.WAKE_LOCK"/>
    <uses-permission android:name="com.android.browser.permission.READ_HISTORY_BOOKMARKS"/>
ENISA: 
```

Figure 96

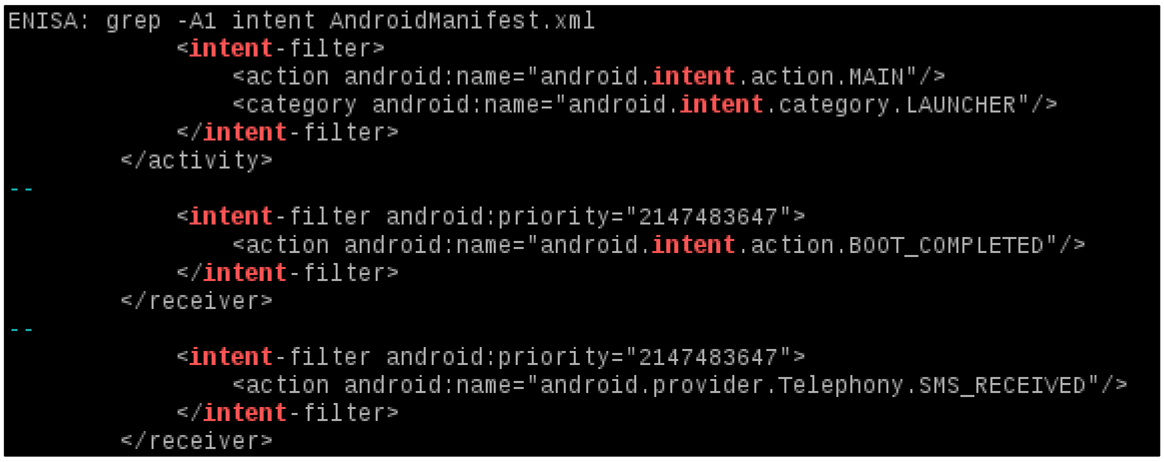
6.2.4.3 Search for the package name in the AndroidManifest:
grep package F836F5C6267F13BF9F6109A6B8D79175/AndroidManifest.xml



```
ENISA: grep package F836F5C6267F13BF9F6109A6B8D79175/AndroidManifest.xml
<manifest xmlns:android="http://schemas.android.com/apk/res/android" package="com.adobe.videoprayer" platformBuildVersionCode="19" platformBuildVersionName="4.4.2-1456859">
ENISA: 
```

Figure 97

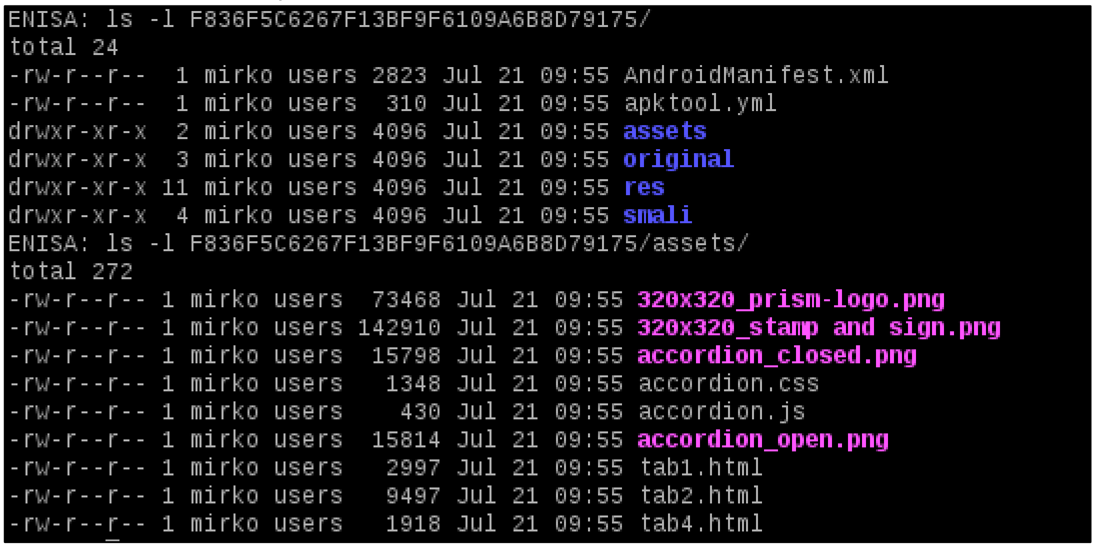
6.2.4.4 Search for the intents in the AndroidManifest.



```
ENISA: grep -A1 intent AndroidManifest.xml
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
--
            <intent-filter android:priority="2147483647">
                <action android:name="android.intent.action.BOOT_COMPLETED"/>
            </intent-filter>
        </receiver>
--
            <intent-filter android:priority="2147483647">
                <action android:name="android.provider.Telephony.SMS_RECEIVED"/>
            </intent-filter>
        </receiver>
```

Figure 98

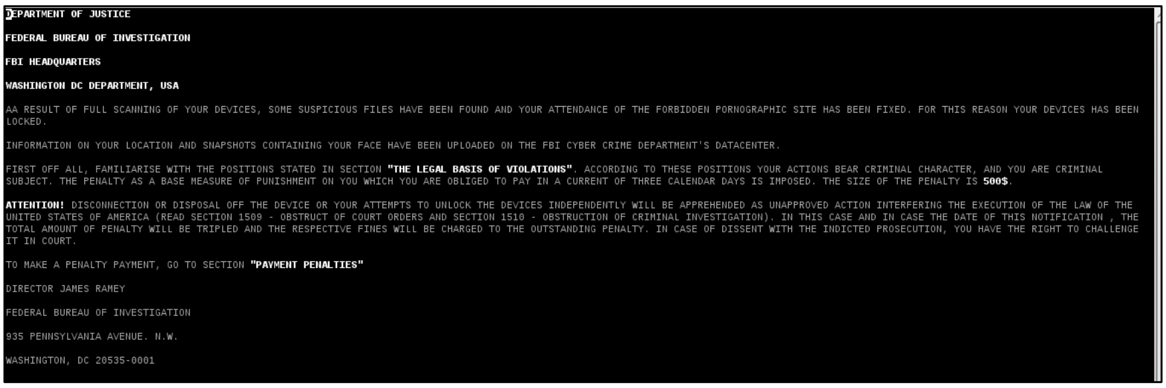
6.2.4.5 Control the assets directory.



```
ENISA: ls -l F836F5C6267F13BF9F6109A6B8D79175/
total 24
-rw-r--r--  1 mirko users 2823 Jul 21 09:55 AndroidManifest.xml
-rw-r--r--  1 mirko users  310 Jul 21 09:55 apktool.yml
drwxr-xr-x  2 mirko users 4096 Jul 21 09:55 assets
drwxr-xr-x  3 mirko users 4096 Jul 21 09:55 original
drwxr-xr-x 11 mirko users 4096 Jul 21 09:55 res
drwxr-xr-x  4 mirko users 4096 Jul 21 09:55 smali
ENISA: ls -l F836F5C6267F13BF9F6109A6B8D79175/assets/
total 272
-rw-r--r-- 1 mirko users  73468 Jul 21 09:55 320x320_prism-logo.png
-rw-r--r-- 1 mirko users 142910 Jul 21 09:55 320x320_stamp and sign.png
-rw-r--r-- 1 mirko users  15798 Jul 21 09:55 accordion_closed.png
-rw-r--r-- 1 mirko users   1348 Jul 21 09:55 accordion.css
-rw-r--r-- 1 mirko users    430 Jul 21 09:55 accordion.js
-rw-r--r-- 1 mirko users  15814 Jul 21 09:55 accordion_open.png
-rw-r--r-- 1 mirko users   2997 Jul 21 09:55 tab1.html
-rw-r--r-- 1 mirko users   9497 Jul 21 09:55 tab2.html
-rw-r--r-- 1 mirko users   1918 Jul 21 09:55 tab4.html
```

Figure 99

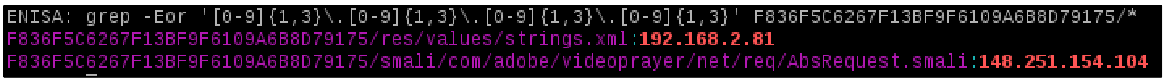
6.2.4.6 View the contents of the HTML file:
w3m F836F5C6267F13BF9F6109A6B8D79175/assets/tab1.html



```
DEPARTMENT OF JUSTICE

FEDERAL BUREAU OF INVESTIGATION

FBI HEADQUARTERS

WASHINGTON DC DEPARTMENT, USA

AA RESULT OF FULL SCANNING OF YOUR DEVICES, SOME SUSPICIOUS FILES HAVE BEEN FOUND AND YOUR ATTENDANCE OF THE FORBIDDEN PORNOGRAPHIC SITE HAS BEEN FIXED. FOR THIS REASON YOUR DEVICES HAS BEEN
LOCKED.

INFORMATION ON YOUR LOCATION AND SNAPSHOTS CONTAINING YOUR FACE HAVE BEEN UPLOADED ON THE FBI CYBER CRIME DEPARTMENT'S DATACENTER.

FIRST OFF ALL, FAMILIARISE WITH THE POSITIONS STATED IN SECTION "THE LEGAL BASIS OF VIOLATIONS". ACCORDING TO THESE POSITIONS YOUR ACTIONS BEAR CRIMINAL CHARACTER, AND YOU ARE CRIMINAL
SUBJECT. THE PENALTY AS A BASE MEASURE OF PUNISHMENT ON YOU WHICH YOU ARE OBLIGED TO PAY IN A CURRENT OF THREE CALENDAR DAYS IS IMPOSED. THE SIZE OF THE PENALTY IS 500$.

ATTENTION! DISCONNECTION OR DISPOSAL OFF THE DEVICE OR YOUR ATTEMPTS TO UNLOCK THE DEVICES INDEPENDENTLY WILL BE APPREHENDED AS UNAPPROVED ACTION INTERFERING THE EXECUTION OF THE LAW OF THE
UNITED STATES OF AMERICA (READ SECTION 1509 - OBSTRUCT OF COURT ORDERS AND SECTION 1510 - OBSTRUCTION OF CRIMINAL INVESTIGATION). IN THIS CASE AND IN CASE THE DATE OF THIS NOTIFICATION , THE
TOTAL AMOUNT OF PENALTY WILL BE TRIPLED AND THE RESPECTIVE FINES WILL BE CHARGED TO THE OUTSTANDING PENALTY. IN CASE OF DISSENT WITH THE INDICTED PROSECUTION, YOU HAVE THE RIGHT TO CHALLENGE
IT IN COURT.

TO MAKE A PENALTY PAYMENT, GO TO SECTION "PAYMENT PENALTIES"

DIRECTOR JAMES RAMEY

FEDERAL BUREAU OF INVESTIGATION

935 PENNSYLVANIA AVENUE. N.W.

WASHINGTON, DC 20535-0001
```

Figure 100

6.2.4.7 Search for IP addresses in the dataset:
*grep -Eor '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}' F836F5C6267F13BF9F6109A6B8D79175/**

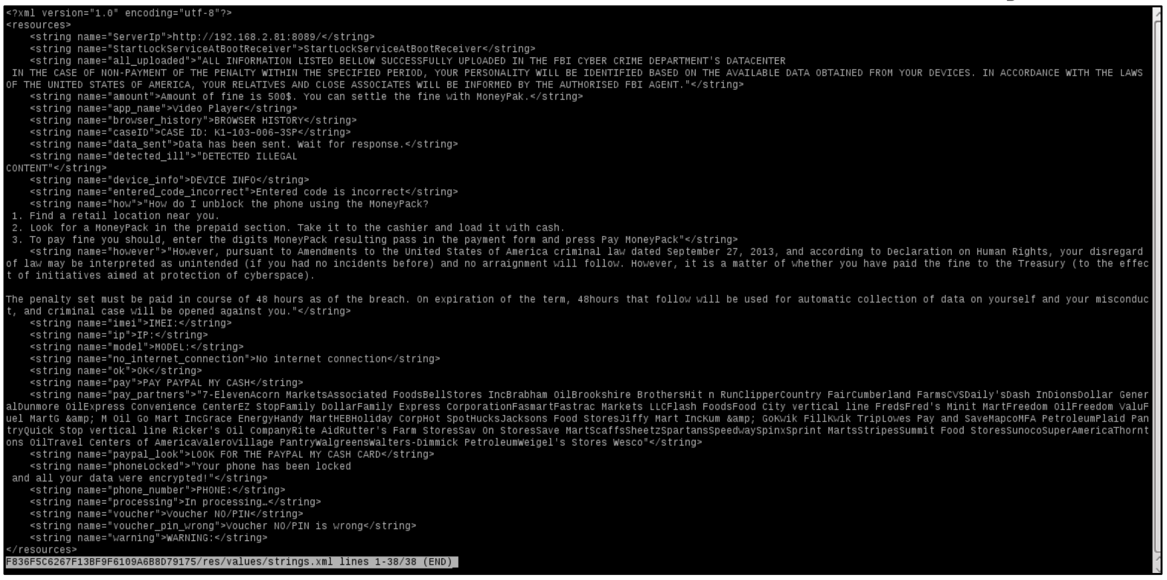


```
ENISA: grep -Eor '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}' F836F5C6267F13BF9F6109A6B8D79175/*
F836F5C6267F13BF9F6109A6B8D79175/res/values/strings.xml:192.168.2.81
F836F5C6267F13BF9F6109A6B8D79175/smali/com/adobe/videoprayer/net/req/AbsRequest.smali:148.251.154.104
```

Figure 101

6.2.4.8 Inspect the strings.xml file:

less F836F5C6267F13BF9F6109A6B8D79175/res/values/strings.xml



```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="ServerIp">http://192.168.2.81:8089/</string>
    <string name="StartLockServiceAtBootReceiver">StartLockServiceAtBootReceiver</string>
    <string name="all_uploaded">"ALL INFORMATION LISTED BELLOW SUCCESSFULLY UPLOADED IN THE FBI CYBER CRIME DEPARTMENT'S DATACENTER
 IN THE CASE OF NON-PAYMENT OF THE PENALTY WITHIN THE SPECIFIED PERIOD, YOUR PERSONALITY WILL BE IDENTIFIED BASED ON THE AVAILABLE DATA OBTAINED FROM YOUR DEVICES. IN ACCORDANCE WITH THE LAWS
OF THE UNITED STATES OF AMERICA, YOUR RELATIVES AND CLOSE ASSOCIATES WILL BE INFORMED BY THE AUTHORISED FBI AGENT."</string>
    <string name="amount">Amount of fine is 500$. You can settle the fine with MoneyPak.</string>
    <string name="app_name">Video Player</string>
    <string name="browser_history">BROWSER HISTORY</string>
    <string name="caseID">CASE ID: K1-103-006-3SP</string>
    <string name="data_sent">Data has been sent. Wait for response.</string>
    <string name="detected_ill">"DETECTED ILLEGAL
CONTENT"</string>
    <string name="device_info">DEVICE INFO</string>
    <string name="entered_code_incorrect">Entered code is incorrect</string>
    <string name="how">"How do I unblock the phone using the MoneyPack?
 1. Find a retail location near you.
 2. Look for a MoneyPack in the prepaid section. Take it to the cashier and load it with cash.
 3. To pay fine you should, enter the digits MoneyPack resulting pass in the payment form and press Pay MoneyPack"</string>
    <string name="however">"However, pursuant to Amendments to the United States of America criminal law dated September 27, 2013, and according to Declaration on Human Rights, your disregard
of law may be interpreted as unintended (if you had no incidents before) and no arraignment will follow. However, it is a matter of whether you have paid the fine to the Treasury (to the effec
t of initiatives aimed at protection of cyberspace).

The penalty set must be paid in course of 48 hours as of the breach. On expiration of the term, 48hours that follow will be used for automatic collection of data on yourself and your misconduc
t, and criminal case will be opened against you."</string>
    <string name="imei">IMEI:</string>
    <string name="ip">IP:</string>
    <string name="model">MODEL:</string>
    <string name="no_internet_connection">No internet connection</string>
    <string name="ok">OK</string>
    <string name="pay">PAY PAYPAL MY CASH</string>
    <string name="pay_partners">"7-ElevenAcorn MarketsAssociated FoodsBellStores IncBrabham OilBrookshire BrothersHit n RunClipperCountry FairCumberland FarmsCVSDaily'sDash InDionsDollar Gener
alDunmore OilExpress Convenience CenterEZ StopFamily DollarFamily Express CorporationFasmartFastrac Markets LLCFlash FoodsFood City vertical line FredsFred's Minit MartFreedom OilFreedom ValuF
uel MartG &amp; M Oil Go Mart IncGrace EnergyHandy MartHEBHoliday CorpHot SpotHucksJacksons Food StoresJiffy Mart IncKum &amp; GoKwik FillKwik TripLowes Pay and SaveMapcoMFA PetroleumPlaid Pan
tryQuick Stop vertical line Ricker's Oil CompanyRite AidRutter's Farm StoresSav On StoresSave MartScaffsSheetzSpartansSpeedwaySpinxSprint MartsStripesSummit Food StoresSunocoSuperAmericaThornt
ons OilTravel Centers of AmericaValeroVillage PantryWalgreensWalters-Dimmick PetroleumWeigel's Stores Wesco"</string>
    <string name="paypal_look">LOOK FOR THE PAYPAL MY CASH CARD</string>
    <string name="phoneLocked">"Your phone has been locked
 and all your data were encrypted!"</string>
    <string name="phone_number">PHONE:</string>
    <string name="processing">In processing…</string>
    <string name="voucher">Voucher NO/PIN</string>
    <string name="voucher_pin_wrong">Voucher NO/PIN is wrong</string>
    <string name="warning">WARNING:</string>
</resources>
F836F5C6267F13BF9F6109A6B8D79175/res/values/strings.xml lines 1-38/38 (END)
```

Figure 102

6.3 Task 6.2: Analysing iOS.Oneclickfraud

6.3.1 Introduction

In this task the students will use class-dump-z to analyse iOS.Oneclickfraud. As in Task 6.1 they will have to answer some questions regarding the characteristics. The trainer will give a short introduction into the usage of the disassembler.

6.3.2 Tools

- class-dump-z

6.3.3 Details

In the exercise directory, students will find an iOS application file 71972F763EB5EAEB87681D2615E9E68E. For the analysis of this file they will have to use the pre-installed class-dump-z disassembler. In the next section they will find questions they have to answer during the analysis to identify the behaviour of the application.

No.	Question	Answer
1	Developer organisation name?	Neon Way Co.,Ltd
	Application identifier and team identifier?	EW4327SQ5D.s.EroEroMovie, EW4327SQ5D
2	Creation and expiration date of the certificate?	2015-05-14T11:08:26, 2016-05-13T11:08:26
3	Certification authorities used in the malware?	Apple iPhone Certification Authority0 Apple Certification Authority1 Apple Root CA0
4	Name of the app as it will be shown to the user?	EroEroMovie
5	Malware classification?	Clickfraud

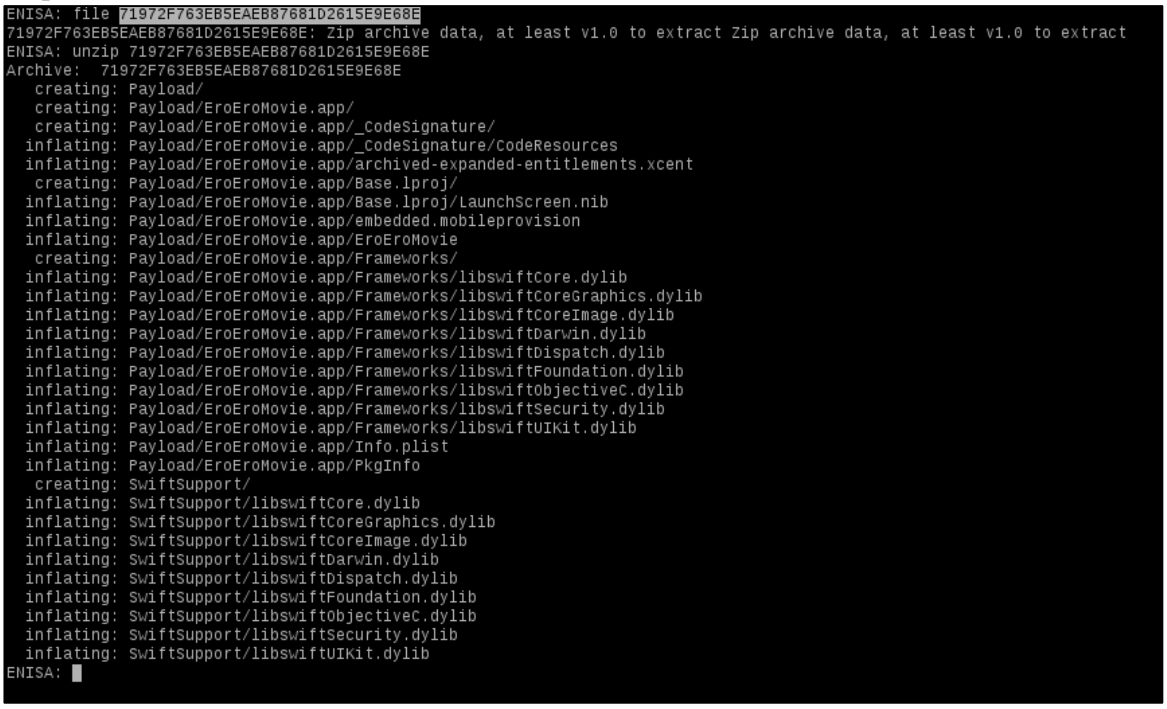
Table 8: Questions and Answers

6.3.4 Task walk-through

In this section a possible approach how to analyse the malware and extract the requested information will be shown.

6.3.4.1 Identify the file and unzip it:

file 71972F763EB5EAEB87681D2615E9E68E
unzip 71972F763EB5EAEB87681D2615E9E68E

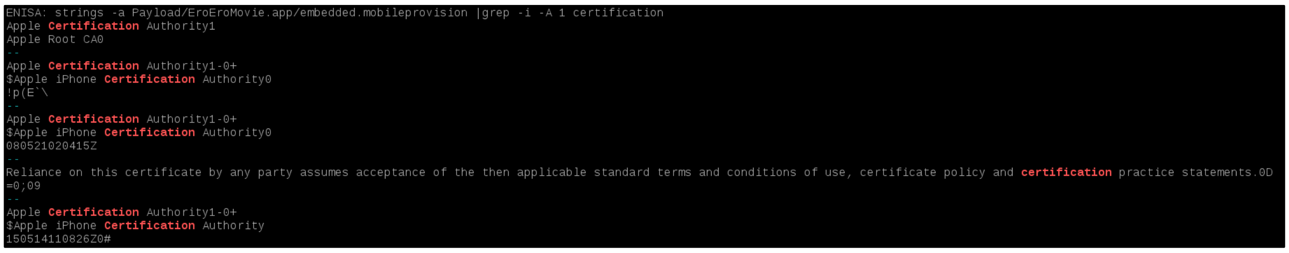


```
ENISA: file 71972F763EB5EAEB87681D2615E9E68E
71972F763EB5EAEB87681D2615E9E68E: Zip archive data, at least v1.0 to extract Zip archive data, at least v1.0 to extract
ENISA: unzip 71972F763EB5EAEB87681D2615E9E68E
Archive:  71972F763EB5EAEB87681D2615E9E68E
   creating: Payload/
   creating: Payload/EroEroMovie.app/
   creating: Payload/EroEroMovie.app/_CodeSignature/
  inflating: Payload/EroEroMovie.app/_CodeSignature/CodeResources
  inflating: Payload/EroEroMovie.app/archived-expanded-entitlements.xcent
   creating: Payload/EroEroMovie.app/Base.lproj/
  inflating: Payload/EroEroMovie.app/Base.lproj/LaunchScreen.nib
  inflating: Payload/EroEroMovie.app/embedded.mobileprovision
  inflating: Payload/EroEroMovie.app/EroEroMovie
   creating: Payload/EroEroMovie.app/Frameworks/
  inflating: Payload/EroEroMovie.app/Frameworks/libswiftCore.dylib
  inflating: Payload/EroEroMovie.app/Frameworks/libswiftCoreGraphics.dylib
  inflating: Payload/EroEroMovie.app/Frameworks/libswiftCoreImage.dylib
  inflating: Payload/EroEroMovie.app/Frameworks/libswiftDarwin.dylib
  inflating: Payload/EroEroMovie.app/Frameworks/libswiftDispatch.dylib
  inflating: Payload/EroEroMovie.app/Frameworks/libswiftFoundation.dylib
  inflating: Payload/EroEroMovie.app/Frameworks/libswiftObjectiveC.dylib
  inflating: Payload/EroEroMovie.app/Frameworks/libswiftSecurity.dylib
  inflating: Payload/EroEroMovie.app/Frameworks/libswiftUIKit.dylib
  inflating: Payload/EroEroMovie.app/Info.plist
  inflating: Payload/EroEroMovie.app/PkgInfo
   creating: SwiftSupport/
  inflating: SwiftSupport/libswiftCore.dylib
  inflating: SwiftSupport/libswiftCoreGraphics.dylib
  inflating: SwiftSupport/libswiftCoreImage.dylib
  inflating: SwiftSupport/libswiftDarwin.dylib
  inflating: SwiftSupport/libswiftDispatch.dylib
  inflating: SwiftSupport/libswiftFoundation.dylib
  inflating: SwiftSupport/libswiftObjectiveC.dylib
  inflating: SwiftSupport/libswiftSecurity.dylib
  inflating: SwiftSupport/libswiftUIKit.dylib
ENISA: 
```

Figure 103

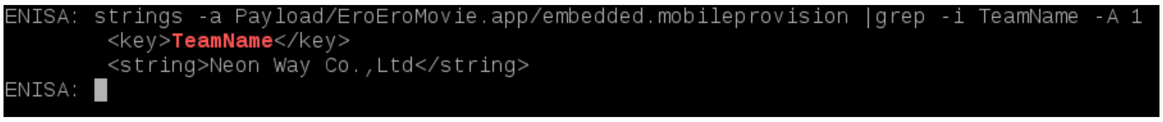
6.3.4.2 Use strings to gather information:

strings -a Payload/EroEroMovie.app/embedded.mobileprovision



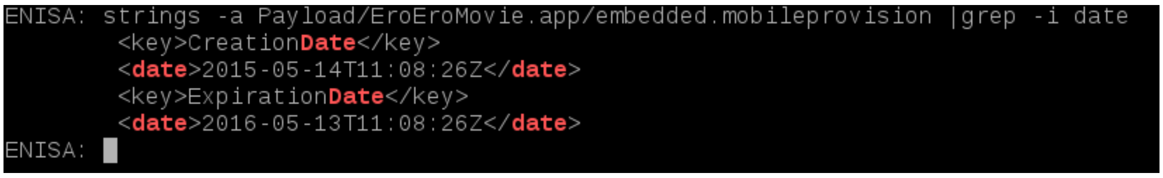
```
ENISA: strings -a Payload/EroEroMovie.app/embedded.mobileprovision |grep -i -A 1 certification
Apple Certification Authority1
Apple Root CA0
--
Apple Certification Authority1-0+
$Apple iPhone Certification Authority0
!p(E`\
--
Apple Certification Authority1-0+
$Apple iPhone Certification Authority0
080521020415Z
--
Reliance on this certificate by any party assumes acceptance of the then applicable standard terms and conditions of use, certificate policy and certification practice statements.0D
=0;09
--
Apple Certification Authority1-0+
$Apple iPhone Certification Authority
150514110826Z0#
```

Figure 104



```
ENISA: strings -a Payload/EroEroMovie.app/embedded.mobileprovision |grep -i TeamName -A 1
        <key>TeamName</key>
        <string>Neon Way Co.,Ltd</string>
ENISA: 
```

Figure 105



```
ENISA: strings -a Payload/EroEroMovie.app/embedded.mobileprovision |grep -i date
        <key>CreationDate</key>
        <date>2015-05-14T11:08:26Z</date>
        <key>ExpirationDate</key>
        <date>2016-05-13T11:08:26Z</date>
ENISA: 
```

Figure 106

7. Recap of mobile forensic tools

Apart from commercial solutions mentioned in Section 3.2, there is a number of open-source or free tools that support mobile forensic investigators in data extraction, examination and analysis. Below is a recap of tools used during the exercises.

7.1 Android SDK[111]

A set of official libraries and tools useful in analysing (as well as developing) of Android-based devices.

7.2 AF Logical OSE[112]

AFLogical Open Source Edition is a free software (available through Google Code) that pulls all available MMS, SMS, Contacts, and Call Logs from an Android device.

7.3 Volatility[113]

This is a completely open collection of tools, implemented in Python under the GNU General Public License, for the extraction of digital artefacts from the volatile memory (RAM) samples.

7.4 Autopsy[114]

Autopsy is a graphical interface to the command line digital investigation analysis tools in The Sleuth Kit. The Sleuth Kit and Autopsy are both open-source and run on UNIX platforms. Autopsy is HTML-based software where one can set up new cases, add files and analyse them from the forensic point of view.

7.5 iPBA2[115]

iPBA2 parses the backup directory and shows the decoded file system tree. Each file can be clicked to see its properties, such as real name and name in the backup directory, file UNIX permissions, data hash (as calculated by iOS), user and group ID, file type (from magic numbers).

7.6 whHexEditor[116]

This is an open-source cross-platform Hex Editor written in C++ and wxWidgets. It can work as low level disk editor too. It uses 64 bit file descriptors. It does not copy the whole file to RAM. This makes it faster and lets it open very large files.

7.7 Exiftool[117]

ExifTool is a free software for reading, writing, and manipulating image, audio, and video metadata.

[111] Android SDK, http://developer.android.com/sdk/index.html, last accessed on: 2015-09-14
[112] The NowSecure Forensics™ community edition allows to complete non-commercial extractions of data from mobile devices, https://www.nowsecure.com/forensics/community/#viaforensics, last accessed on: 2015-09-14
[113] Volatility Foundation, https://github.com/volatilityfoundation, last accessed on: 2015-09-14
[114] Autopsy® is a digital forensics platform and graphical interface to The Sleuth Kit® and other digital forensics tools, http://www.sleuthkit.org/autopsy/, last accessed on: 2015-09-14
[115] iPhone Backup Analyzer 2, https://github.com/PicciMario/iPhone-Backup-Analyzer-2, last accessed on: 2015-09-14
[116] wxHexEditor: A free hex editor / disk editor for Linux, Windows and MacOSX, http://www.wxhexeditor.org/, last accessed on: 2015-09-14
[117] ExifTool is a platform-independent Perl library plus a command-line application for reading, writing and editing meta information in a wide variety of files, http://www.sno.phy.queensu.ca/~phil/exiftool/, last accessed on: 2015-09-14

7.8 Tcpdump[118]

Tcpdump is an open-source tool to access network traffic on COTS systems running a Linux or UNIX variant. It has a feature rich filtering language and can be used to write network data to files (PCAP) for later analysis with applications like Wireshark.

7.9 MITMproxy[119]

This tool is a handy proxy to capture HTTP traffic and dump it to a file. It is also capable to intercept SSL protected traffic by creating certificates for target servers on the fly.

7.10 HoneyProxy[120]

The project is based on MITMProxy and creates a web interface to inspect and analyse the traffic captured. Unfortunately it is not under active development currently.

7.11 Wireshark[121]

Wireshark listens to the network traffic and can read PCAP files and provides a set of analytical functions (statistics, visualisation, protocol decoding) to support an analyst in the process of network traffic inspection.

7.12 Apktool[122]

This is a reverse engineering tool for Android binaries. It decodes resources and tries to provide a structure which can be used to rebuild the original application from the source.

7.13 Strings[123]

Strings is a simple tool to extract ASCII strings from binary files. It is part of the GNU Binutils package.

[118] Tcpdump: Network traffic sniffer, http://www.tcpdump.org/, last accessed on: 2015-09-14
[119] MitM Proxy: An interactive console program that allows traffic flows to be intercepted, inspected, modified and replayed, https://mitmproxy.org/, last accessed on: 2015-09-14
[120] HoneyProxy: A man-in-the-middle SSL proxy & traffic analyser, http://honeyproxy.org/, last accessed on: 2015-09-14
[121] Wireshark: Network protocol analyser, https://www.wireshark.org/, last accessed on: 2015-09-14
[122] Apktool: A tool for reverse engineering Android apk files, https://github.com/iBotPeaches/Apktool, last accessed on: 2015-09-14
[123] GNU Binutils, https://www.gnu.org/software/binutils/, last accessed on: 2015-09-14

8. Countermeasures, protective measures

8.1 Sandboxes

Sandboxing is a method aimed at providing a secure environment for data processing by limiting the way the application interacts with the host operating system, with other applications and *vice versa*. The limits are imposed upon file system access, network and inter-process communications, device access, memory and system resources management. The approach is based on the assumption that any interaction must be controlled and performed in a supervised, structured and almost-formally approved way. Applications and their developers are subject to security policies and standards implemented by a mobile device vendor. Omitting some of those security measures is possible through device jail-breaking or rooting (explained later in this chapter). Specific security settings for most popular mobile platforms are explained in their vendors' websites:

- Android
  http://developer.android.com/training/articles/security-tips.html
  https://source.android.com/devices/tech/security/

- iOS
  https://developer.apple.com/app-sandboxing/
  https://developer.apple.com/library/mac/documentation/Security/Conceptual/AppSandboxDesignGuide/AboutAppSandbox/AboutAppSandbox.html
  https://developer.apple.com/library/mac/documentation/Security/Conceptual/AppSandboxDesignGuide/AppSandboxInDepth/AppSandboxInDepth.html

- Windows Phone
  https://msdn.microsoft.com/en-us/library/windows/apps/Ff402533(v=VS.105).aspx

Sandboxing has its pros and cons. Among the advantages are the following features:

- Sandboxes provide separation between applications and the operating system making it more difficult to expand an unauthorised access in case of successful exploitation of a vulnerability.
- Sandboxing provides better control over access to system resources and hardware components.
- From mobile applications developers' point of view sandboxing provides a more or less standard environment that end-users have little or no impact upon.

The disadvantages include:

- Extended attack surface.
- False perception of security in case of a compromise of sandboxing mechanism or exploitation of operating system's vulnerability.
- Mobile application developers may tend to ignore security principles, over-relying on sandboxes for protection.

After all, sandboxes are an efficient and suitable way of ensuring separation and segregation of data processing environments on mobile platforms (and other platforms as well). They should not be treated as a final solution or a security measure of a last resort.

8.2 Antivirus software for mobile systems

For many years mobile phone were not seen as a target for viruses and malware. Due to a rapid growth of the user base, connectivity and functionalities of mobile devices attackers started looking for vulnerabilities and exploiting them. Since mobile devices were perceived as a secure channel of communications they were used for providing one-time passwords or transaction authorisation numbers e.g. by the financial institutions. But attacks such as ZitMo[124] proved this assumption to be wrong, hence shifting the paradigm for security companies and researchers, attackers and last but possibly least for the users.

Nevertheless an anti-virus software dedicated for mobile devices still does not seem to be as popular as on desktop or laptop computers, even though most major (and the less major) AV players provide their solutions for mobile devices.[125] Thus the software developers and hardware providers utilise a number of other security controls (e.g. sandboxing, Mobile Device Management systems) to better protect mobile devices' users.

8.3 Mobile Device Management (MDM) systems

Mobile device management systems are intended to provide organisations with the capability to verify, monitor and control behaviour of mobile devices' users. Among their major functionalities are:

- User & group management
- Access control
- Enforcing policies and settings
- Provisioning of a standard or pre-set configuration
- Mobile application management
- Monitoring of a mobile device
- Reporting of the usage
- Remote control, lock-out and wiping

The architecture of an MDM system includes the mobile-device component and the server infrastructure. MDM system can be provided as a service or as an appliance. It often integrates with corporate systems like Active Directory or LDAP to pull information related to users and groups.

For the mobile part, MDM solutions often rely on a secure-container solution that is supposed to:

- Control access to data stored on the device
- Only allow certain operations to be performed and applications run
- Authenticate and authorise user of the device before accessing any of the above

[124] ZITMO: The new mobile threat, http://www.cert.pl/news/3193/langswitch_lang/en, last accessed on: 2015-09-14
[125] Anti-virus software for Android platform, https://www.av-test.org/en/antivirus/mobile-devices/, last accessed on: 2015-09-14

Successful implementation of the above-mentioned features requires the secure container to be resistant to attacks aimed at the:

- Extraction of the encryption key
- Brute-forcing of the password
- Bypassing access control settings
- Omitting of limitations on running applications or using certain features

With software-based MDM solutions, it is very difficult to achieve the first two security objectives, hence such MDM solutions are theoretically prone to password brute-forcing or dictionary attacks if the security container or the certificate is protected by a simple or short password or PIN code. Defence-in-depth or two-factor authentication allows for a higher level of protection but on the other hand lacks flexibility and user-friendliness.

For the server-side part, functionalities provided may also be lacking necessary security controls allowing more advanced mobile device users (or attackers) to exploit vulnerabilities resulting from:

- Lack of access control
- Poor input and output data validations
- Business logic flaws
- Out-dated software etc.

It is worth to mention Android for Work[126] solution which is an EMM (Enterprise Mobility Management) platform acting as an extension to MDM solutions and providing many more functions and higher level of security through application of business context to mobile technologies management.

From the mobile forensics point of view, MDM and EMM systems provide a standardised environment for mobile data processing thus enabling forensic investigators to be able to compare the device in question with default configuration or application set, detect anomalies and follow the patterns. On the other hand, an MDM system dramatically expands the attack surface, becoming a source of potential vulnerabilities and a false sense of security for organisations.

[126] Android for Work makes your favourite smartphone or tablet the perfect business tool, https://www.android.com/work/, last accessed on: 2015-09-14

9. References

- Apktool: A tool for reverse engineering Android apk files
  https://github.com/iBotPeaches/Apktool
- Andrubis: A Tool for Analysing Unknown Android Applications
  http://blog.iseclab.org/2012/06/04/andrubis-a-tool-for-analysing-unknown-android-applications-2/
- Bitdefender: Android ransomware uses fake FBI porn warning
  http://www.cbronline.com/news/mobility/security/bitdefender-android-ransomware-uses-fake-fbi-porn-warning-4585753
- Class-dump-z: Extracting class interface for Objective-C
  https://code.google.com/p/networkpx/wiki/class_dump_z
- Ettus research
  http://www.ettus.com/home
- Fairwaves
  http://shop.fairwaves.co/UmTRX-transceiver-for-OpenBTS-and-Osmocom-OpenBSC
- Nuand
  https://nuand.com/
- GNU Binutils
  https://www.gnu.org/software/binutils/
- Android Open Kang Project
  http://aokp.co/
- Google quietly backs away from encrypting new Lollipop devices by default
  http://arstechnica.com/gadgets/2015/03/google-quietly-backs-away-from-encrypting-new-lollipop-devices-by-default/
- 950 million Android users at risk as researcher uncovers massive security flaw
  http://bgr.com/2015/07/27/android-security-flaw-mms-hack/
- Android applications permissions
  http://developer.android.com/preview/features/runtime-permissions.html
- Android SDK
  http://developer.android.com/sdk/index.html
- Build System Overview
  http://developer.android.com/sdk/installing/studio-build.html
- Android NDK
  http://developer.android.com/tools/sdk/ndk/index.html
- MIUI ROM
  http://en.miui.com/
- Pangu Jailbreak for iOS 8
  http://en.pangu.io/
- evasi0n7 Jailbreak for iOS 7
  http://evasi0n.com/
- The 7 bit default alphabet
  http://holman.id.au/apps/ipsms/default_alphabet.html
- HoneyProxy: a man-in-the-middle SSL proxy & traffic analyser
  http://honeyproxy.org/
- OpenCellID is the world's largest collaborative community project that collects GPS positions of cell towers, used free of charge, for a multitude of commercial and private purposes.
  http://opencellid.org/

- OpenLTE is an open source implementation of the 3GPP LTE specifications
  http://openlte.sourceforge.net/
- The Osmocom project is a family of projects regarding Open source mobile communications
  http://osmocom.org/
- PerfectRoot's Remote Android Rooting Services ensure increased security and higher functionality such as increased memory, battery life, system speeds, and custom app capability
  http://perfectroot.com/
- Keyraider: iOS malware steals over 225,000 Apple accounts to create free app Utopia
  http://researchcenter.paloaltonetworks.com/2015/08/keyraider-ios-malware-steals-over-225000-apple-accounts-to-create-free-app-utopia/
- OsmoSDR is a 100% Free Software based small form-factor inexpensive SDR (Software Defined Radio) project
  http://sdr.osmocom.org/trac/
- Libdwarf and Dwarfdump
  http://wiki.dwarfstd.org/index.php?title=Libdwarf_And_Dwarfdump
- How is Google fixing the Stagefright vulnerability that affects 95% of all Android phones?
  http://www.androidauthority.com/how-is-google-fixing-the-stagefright-vulnerability-that-affects-95-of-all-android-phones-631560/
- UFED CHINEX is an end-to-end solution for the physical extraction and decoding of evidentiary data and passwords from phones manufactured with Chinese chipsets — MTK, Spreadtrum and Infineon
  http://www.cellebrite.com/Pages/ufed-chinex
- Cellular Intercept and Cellular Monitoring technologies give Law Enforcement and Government Agencies a technological edge
  http://www.cellularintercept.com/
- ZITMO: The new mobile threat
  http://www.cert.pl/news/3193/langswitch_lang/en
- ROM Manager: ROMs and Recovery Images
  http://www.clockworkmod.com/rommanager
- CoreText in Apple iOS 8.x through 8.3 allows remote attackers to cause a denial of service
  http://www.cvedetails.com/cve/CVE-2015-1157/
- CyanogenMod is an enhanced open source firmware distribution for smartphones and tablet computers based on the Android mobile operating system
  http://www.cyanogenmod.org/
- File signatures table
  http://www.garykessler.net/library/file_sigs.html
- One-click Android root software
  http://www.kingoapp.com/android-root/feature-one.htm
- iOS 8 jailbreak
  http://www.redsn0w.us/
- Autopsy® is a digital forensics platform and graphical interface to The Sleuth Kit® and other digital forensics tools
  http://www.sleuthkit.org/autopsy/
- ExifTool is a platform-independent Perl library plus a command-line application for reading, writing and editing meta information in a wide variety of files
  http://www.sno.phy.queensu.ca/~phil/exiftool/
- iOS 8.4 jailbreak
  http://www.taig.com/en/
- Tcpdump: Network traffic sniffer

http://www.tcpdump.org/
- XACT: Mobile phone investigations go deeper
  http://www.veille.ma/IMG/pdf/xact-datasheet.pdf
- wxHexEditor: a free hex editor / disk editor for Linux, Windows and MacOSX
  http://www.wxhexeditor.org/
- YateBTS
  http://www.yatebts.com/
- Why I Hacked Apple's TouchID, And Still Think It Is Awesome
  https://blog.lookout.com/blog/2013/09/23/why-i-hacked-apples-touchid-and-still-think-it-is-awesome/
- Why I hacked TouchID (again) and still think it's awesome
  https://blog.lookout.com/blog/2014/09/23/iphone-6-touchid-hack/
- Volatility: RAM dump analyser
  https://code.google.com/p/volatility/wiki/
- Volatility: A command reference for Linux
  https://code.google.com/p/volatility/wiki/LinuxCommandReference23
- Universal Binaries and 32-bit/64-bit PowerPC Binaries
  https://developer.apple.com/library/mac/documentation/DeveloperTools/Conceptual/MachORuntime/index.html#//apple_ref/c/tag/fat_header
- Android version history
  https://en.wikipedia.org/wiki/Android_version_history
- BlackBerry OS release history
  https://en.wikipedia.org/wiki/BlackBerry_OS#Release_history
- iOS version history
  https://en.wikipedia.org/wiki/IOS_version_history
- Mobile operating system
  https://en.wikipedia.org/wiki/Mobile_operating_system
- Window Phone version history
  https://en.wikipedia.org/wiki/Windows_Phone_version_history
- LiME: Linux Memory Extractor
  https://github.com/504ensicslabs/lime
- iPhone Backup Analyzer 2
  https://github.com/PicciMario/iPhone-Backup-Analyzer-2
- Volatility Foundation
  https://github.com/volatilityfoundation
- GSMA Intelligence
  https://gsmaintelligence.com/
- MitM Proxy: An interactive console program that allows traffic flows to be intercepted, inspected, modified and replayed
  https://mitmproxy.org/
- Santoku is dedicated to mobile forensics, analysis, and security, and packaged in an easy to use, Open Source platform
  https://santoku-linux.com/
- Android for Work makes your favourite smartphone or tablet the perfect business tool
  https://www.android.com/work/
- iOS Security Guide
  https://www.apple.com/business/docs/iOS_Security_Guide.pdf
- Anti-virus software for Android platform

https://www.av-test.org/en/antivirus/mobile-devices/

- Fingerprints on mobile devices: abusing and leaking
  https://www.blackhat.com/us-15/briefings.html#fingerprints-on-mobile-devices-abusing-and-leaking
- Mobile threats incident handling — Handbook, Document for teachers
  https://www.enisa.europa.eu/activities/cert/support/exercise/files/Mobileincidenthandlinghandbook.pdf
- Mobile threats incident handling — Toolset, Document for students
  https://www.enisa.europa.eu/activities/cert/support/exercise/files/Mobilethreatsincidenthandlingtoolset.pdf
- XRY PinPoint: The Solution for Non-Standard Mobile Device Support
  https://www.msab.com/products/pinpoint/
- The NowSecure Forensics™ community edition allows to complete non-commercial extractions of data from mobile devices
  https://www.nowsecure.com/forensics/community/#viaforensics
- Wireshark: Network protocol analyser
  https://www.wireshark.org/
- Symantec's Internet Security Threat Report
  https://www4.symantec.com/mktginfo/whitepaper/ISTR/21347932_GA-internet-security-threat-report-volume-20-2015-social_v2.pdf
- Japanese one-click fraudsters target iOS users with malicious app delivered over the air
  http://www.symantec.com/connect/blogs/japanese-one-click-fraudsters-target-ios-users-malicious-app-delivered-over-air
- Reverse engineering, Malware and goodware analysis of Android applications
  https://github.com/androguard/androguard
- The OS X and Linux Disassembler
  http://www.hopperapp.com/
- Chinese Chipsets - Physical Extraction from Mobile Devices with Chinese Chipsets
  http://www.cellebrite.com/Pages/chinese-chipsets-mobile-forensics-for-chinese-chipsets
- iOS 9
  http://www.apple.com/ios/whats-new/
- Android 6.0 Marshmallow
  https://www.android.com/versions/marshmallow-6-0/
- Windows 10 Mobile
  http://www.microsoft.com/en-us/mobile/windows10/
- Palo Alto Networks
  https://www.paloaltonetworks.com/
- Samsung KNOX system
  https://www.samsungknox.com/en
- NowSecure Forensics
  https://www.nowsecure.com/forensics/
- AccessData's MPE+
  http://accessdata.com/solutions/digital-forensics/mpe
- Cellebrite UFED
  http://www.cellebrite.com/Mobile-Forensics
- EnCase Forensic
  https://www.guidancesoftware.com/products/Pages/encase-forensic/overview.aspx
- Micro Systemation XRY
  https://www.msab.com/products/xry/?gclid=CJ3I47uMzsgCFerpwgod6BED2g

- FINALDATA FINALMobile Forensics
  http://www.finaldata.com/
- Lantern 3
  http://shop.osxforensics.com/
- Logicube CellXtract
  http://www.logicube.com/shop/cellxtract/
- MOBILedit! Forensic
  http://www.mobiledit.com/forensic
- Oxygen Forensic Suite
  http://www.oxygen-forensic.com/en/
- Paraben Device Seizure
  https://www.paraben.com/device-seizure.html
- Phone Forensics Express
  http://www.mobiledit.com/forensic-express
- Radio Tactics' Athena
  http://www.radio-tactics.com/
- Secure View 3
  http://secureview.us/
- MediaTek (MTK)
  http://www.mediatek.com/
- Spreadtrum
  www.spreadtrum.com/
- MStar
  www.mstarsemi.com/
- Cellebrite's CHINEX
  http://www.cellebrite.com/Pages/ufed-chinex
- eDEC's Tarantula
  https://www.edecdf.com/product/tarantula-cell-phone-extraction-kit/
- XRY PinPoint
  https://www.msab.com/products/pinpoint/
- MFC Dongle
  http://www.mfcbox.com/
- XPIN Clip
  http://xpinclip.com/
- Genie Universal 2010 Clip
  http://multi-com.eu/,details,id_pr,6998,key,genie-universal-2010-clip.html
- AVD
  http://developer.android.com/tools/help/avd-manager.html
- AF Logical OSE
  https://santoku-linux.com/howto/howto-use-aflogical-ose-logical-forensics-android/
- wxHexEditor
  http://www.wxhexeditor.org/
- Micro Systemation XRY/XACT
  https://www.msab.com/
- foremost.
  https://en.wikipedia.org/wiki/Foremost_%28software%29
- Volatility
  https://code.google.com/p/volatility/

- iPBA2
  https://github.com/PicciMario/iPhone-Backup-Analyzer-2
- Wireshark
  https://www.wireshark.org/
- MITMProxy
  https://mitmproxy.org
- AndroGuard
  https://github.com/androguard/androguard
- apktool
  http://ibotpeaches.github.io/Apktool/
- class-dump-z
  https://code.google.com/p/networkpx/wiki/class_dump_z

ENISA

European Union Agency for Network
and Information Security
Science and Technology Park of Crete (ITE)
Vassilika Vouton, 700 13, Heraklion, Greece

Athens Office

1 Vass. Sofias & Meg. Alexandrou
Marousi 151 24, Athens, Greece

PO Box 1309, 710 01 Heraklion, Greece
Tel: +30 28 14 40 9710
info@enisa.europa.eu
www.enisa.europa.eu