Android Forensics and Security Testing Exercises and Linux Commands

Contents

Exercise 1 - Create AVD and explore directories of interest

Objectives

- Create an Android Virtual Device for use during the class
- Identify file system directories and familiarize with the directory tree

Instructions

1. Create AVD titled “FroyoForensics” with Android 2.2
   a. Use slides on AVD for guidance
2. (Optional) Create AVD based on your own Android device
3. Explore /.android subdirectories, using command line tools
   a. Use Directory Tree slide for guidance
4. Locate cache.img
   a. Use Interesting Files slide for guidance
5. (Optional) Add UDEV Rules for your Android device. This will allow you to provide forensic analysis on your device.
   a. Use USB Vendor ID and UDEV slides for guidance

NOTES:

Exercise 2 - Locate data directory on an Android device

Objectives

- Verify we can connect an Android device to a forensic workstation
- Attempt to access shell and locate data directories

Instructions

1. Connect an Android device to your VM workstation with USB cable (or startup an AVD)
   a. Use slides on Connecting Device and USB for guidance
2. Verify USB Debugging is enabled on the device
   a. Use slides on USB Debugging for guidance
3. Start adb on your forensic workstation
   a. Use slides on ADB for guidance
4. Using adb shell, locate directories in /data/data
   a. Use slides on ADB Shell for guidance
5. Jot down the name of some interesting directories for further exploration later

6. (Optional) Check for mounted SD cards
   a. Use slide on USB Forensics Precaution for guidance

NOTES:

Exercise 3 - Apply Android forensics knowledge to locate data of interest

Objectives

- Become familiar with common command line utilities for locating data
- Explore most common data directories and databases

Instructions

1. Using adb shell (or /.android if using an AVD), explore an applications shared_prefs within /data/data
   a. Use slides on directories Shared preferences for guidance
2. Use the cat command to open an xml file and review the contents
3. Note anything of interest to share with the class

4. Using sqlite3, explore an applications databases within /data/data
   a. Use slides on SQLite for guidance
5. Use .tables and select commands to gather data of interest, which could identify something specific about the user.
6. Note anything of interest to share with the class

7. (Optional) Run live stream of device messages in terminal, while running an application
   a. Use slides on logcat for guidance

NOTES:

Exercise 4 - Attempt to circumvent passcode and obtain temp root access

Objectives

- Apply rooting techniques, using available tools

Instructions

1. Identify what type (if any) of passcode is enabled on the device
   a. Use Passcode Types slides for guidance

2. Confirm if device is already rooted, or not
   a. Use Temp Root slides for guidance
3. If not rooted, attempt to enable Temp Root (aka Shell Root)
   a. Use SuperOneClick slides for guidance
4. (optional) Apply the Extend, Enable, Disable techniques of a "first responder"
   a. Use Device Acquisition slide for guidance
5. (optional) Verify if user accessible Recovery Mode is on your device
   a. Use Recovery Mode slides for guidance
6. (optional – after verification in #5) Verify if Recovery Mode has root access

NOTES:

Exercise 5 - Logical Acquisition of Data

Objectives

- Extract a logical acquisition from device or AVD
- Document data size extracted

Instructions

1. Execute a logical data extraction of /data with ADB Pull
    a. Use ADB Pull slides for guidance
2. Document number of files pulled and skipped

3. (optional) Using QtADB, run logcat
    a. Use QtADB slides for guidance
4. (optional) Using QtADB, execute the same logical extraction from Step #1.

NOTES:

Exercise 6 - Determine what the user does for work and fun

Objectives

- Explore different commercial and open-source Android forensics products
- Identify data on device which can be used as evidence to identify user activity

Instructions

1. (Group / Individual activity) Now that you have acquired data many different ways, analyze the data using one of the forensics tools (adb, adb shell, Device Seizure, QtADB, etc) to get a fresh data acquisition from your device
2. Look at earlier exercises for commands, as a refresher
3. Explore data in directories like /data/ and /cache/
4. As a forensic analyst, document findings that would help you determine the users profession and hobbies
5. Be prepared to share your findings with the class

Investigators Name(s):

Investigation Date:

Data Extraction File Size:

Recent Photos Detail / include geo-location if available:

Recent GPS details:

Recent SMS / email details:

NOTES:

Exercise 7 - Reverse engineer an app and locate critical data

Objectives

- Explore reversing tools for Android
- Reverse engineer an Android application using available tools
- Locate data within the application

Instructions

1. Use APKInspector
    a. At command line, navigate to "/opt/apkinspector", run command "python startQT.py"
2. Attempt to reverse engineer Facebook or F-Droid .apk, located in Documents directory of forensics workstation (HINT: File > New; locate .apk file to reverse)
    a. NOTE: F-Droid may have issue reversing
3. Be prepared to share your findings with the class

NOTES:

Back Cover - Linux commands

./android	Run Android SDK Manager and AVD Manager
df –h	Display free disk space. –h displays sized in K, M and G. Easier to read.
adb devices	Identifies Android devices running abdb and connected to workstation.
adb kill-server	Kills running adb server. Useful if ‘adb devices’ is not responding properly.
adb pull <remote dir> <local dir>	Pulls data from an emulator/device instance’s data file
adb shell	Opens a shell on an Android device.
apt-get	Advanced Packaging Tool used for installing/uninstalling software via Linux command line
cat	Used to display file contents in shell
dd	Unix program for copying / converting raw data
dmesg	Displays Linux kernel messages. Useful with AVD or adb shell
gconf-editor	Opens Configuration Editor application, similar to registry editor in Windows. For Android forensics, it’s used for enabling / disabling automount for mobile devices.
grep	Used for searching keywords; will become indispensable if using Linux for forensics investigations
lsusb -v	Lists all USB devices. –v displays verbose details. Helpful if needing to identify ‘idVendor’ for updating udev rules.
mount	For mounting a file system (commonly when mounting an Android device to a forensics workstation)
nano	Will follow the path and open that file if it exists. If it does not exist, it’ll start a new buffer with that filename in that directory
sqlite 3 <db name>	Opens SQLite
.tables	lists all tables
CTRL+z	Exits SQLite
sudo	Running in escalated mode; usually as superuser or root; useful for rooted Android devices
sudo nano -w /etc/udev/rules.d/51-android.rules	File for adding USB Vendor IDs
tar xzvf	Unzip / extract package utility; e**x**tract, un**z**ip, **v**erbose, **f**ile.