CSE 484 Mobile Security Lab

Congratulations! You've just completed your degree from the University of Washington and have now taken a job at the CIA (or the intelligence agency of your choice). Your first assignment: to work in the mobile phone security division. You know that key officials in other countries have Android phones and that they often store significant amounts of secret data on their phones. Even if it's not "secret," a lot of the information can be used for intelligence gathering purposes. For example, who does this government official communicate with frequently? And, are they working on any secret project at the moment?

In addition to providing you your first assignment as a CIA agent, we will also use this lab as an opportunity for everyone to learn more about mobile phone security, and Android security in particular. Since the CIA is big on practical deliverables (and so are we), you'll be showing us a demo of all your work.

Before you begin, read the whole document and especially the Things Not To Do section later in this description

Part 0

Due: Monday, May 21th, 5pm (no extensions allowed)

Your boss at the CIA wants to make sure you're "up to snuff" and has given you a quick learning task. This task will ensure that you get started early and don't lose time to setting up the Android development environment later. Don't fail your boss, we hear spooks are really good at shaming.

Tasks

- Get Eclipse (or your IDE of choice) set up and running the 4.0.X Android emulator.
- Get a sample program (can be an existing program or one you write yourself) running on the provided phone.
- Modify the sample program to display the names of ALL your groups team members.

Turn-in

A zip file containing the following should be turned in through Catalyst (https://catalyst.uw.edu/collectit/dropbox/kohno/20970) by the due date:

- Screenshot showing the IDE and the emulator running.
- Picture of your phone running one of the sample apps (see 'Getting Started' section below). The application should be displaying ALL the team member names in the photo.

- A README text file with:
    - The link to the description page of the app you selected.
    - ALL the team members names.
    - Anything we need to know in order to build and run your app.

Grading

Your boss will mentally give you 5 reputation points for completing Part 0 (and we'll give you 5 points too).

Live Demo

For the demo (more on this later), please show the app starting and your names appearing. This demo will occur after you have completed Part 1.

Part 1

Due: Tuesday, May 29th, 5pm (no late submissions)

The boss is happy with your progress; you're a quick learner! In order to pursue "persons of interest," the boss has asked you to modify some existing Android applications to be more spy friendly... errr... to have more "hidden" features. Specifically, the boss has asked you to modify applications to maliciously gather and exfiltrate sensitive and valuable information (*e.g.,* location, photo, contact list, *etc.*).

After you finish these tasks, you'll be a pro (well, almost) on how the Android operating system permission model works and the risks associated with Android applications.

Tasks

- Task 0: Create a simple web application that can store information sent to it. For example, you could create a web application that will store to a text file anything provided in a URL parameter (you did this for lab 2). This application will be useful in completing further tasks.

  You might refine it as you complete later tasks -- this is fine and expected. During the demo, we will ask you to show us how and where your application stores data. Be ready to explain to us the design choices you've made.

- Task 1: Write or modify an existing application that legitimately needs access to a sensitive resource (see below), but uses it at a time when it does not actually need it. Exfiltrate that information to a server that you have created in Task 0 above. For example, a search application may need camera permissions in order for the user to be able to perform a search by image. You could modify such an application to take a picture of the user (using the front-facing camera) whenever that user performs a text search and upload that image along with the search term to your server. You can see

how spooks might use this kind of information for intelligence or even blackmail, right? At least, that's one of the things the boss was thinking.

For this assignment, the sensitive resource in question must either be one (or more of): photo from camera, contact list, microphone data, or accelerometer data. For the context data (microphone, accelerometer), you must exfiltrate a continuous stream of at least 10 seconds in length. If you would like to exfiltrate other data *instead of* the aforementioned data, please ask first or you may not receive credit. If you would like to exfiltrate other data *in addition to* the aforementioned data -- go ahead! You might even receive extra credit.

During the project demo, we may ask you to demonstrate your app exfiltrating data. For example, if your app exfiltrates images, we may ask you to take an image of us using the phone and then show us the image on your server. Or, if your app exfiltrates sound, we might ask you to exfiltrate our conversation and then play it back on your server. As another example, if your app exfiltrates the contact list, we may add a contact and ask you to show the exfiltrated data reflecting our change on your server application.

- Task 2: Write or modify two applications. One (App 1) has many privileges and needs legitimate access to sensitive resources (e.g., the mail app needs access to the contact list). Another application (App 2) does not have access to that resource (as per the manifest). Add functionality to App 2 (and App 1 as necessary) so that App 2 can access the sensitive resources from App 1 (via APIs/intents). The two applications must run in different processes. Finally, just as you did in Task 1, exfiltrate that data to your server. Sneaky eh?

  For this assignment, the sensitive resource in question is defined just as in Task 1. Similarly, as described in Task 1, expect to be asked to demonstrate these capabilities during the demo.

- Task 3: Write or modify two apps that are signed by the same certificate. The first application (App 1) should have access to a high privilege resource, the other application (App 2) should not (via the manifest). Because of the nature of the Android security model, your boss believes that App 2 should be able to access any high privilege resource accessible by App 1. Modify App 2 to access this sensitive resource and exfiltrate that data to your server.

  For this assignment, the sensitive resource in question is defined just as in Task 1. Similarly, as described in Task 1, expect to be asked to demonstrate these capabilities during the demo.

- Meta Task: For one of the tasks (1-3) above, encrypt and authenticate the data

your application exfiltrated before sending it to your server. You must also write corresponding decryption and verification code, so that you -- as an intelligence agent -- can actually access and read the exfiltrated data (and know that it's authentic). The decryption and verification code can live wherever you wish (e.g., on your server, or on another machine), but you must be able to demonstrate the decryption process during the demo.

- Extra credit opportunities (a.k.a. impress your boss):
    - When malware is disguised or hidden, it's much harder to find and much more likely to stay in the application. Modify your implementation of one (or multiple) above tasks to hide the malicious functionality from an analyst who might have access to the source code. Be ready to explain why what you've done will make it harder for people (and app store maintainers) to discover your "extra" features when looking at the source code.
    - If a binary is obfuscated, it is much harder to reverse engineer. Use an obfuscator to make your application harder to analyze. In your writeup, you should describe the obfuscator that you use, as well as why you think it will make the analysis of your application harder.
    - Modify an application to exfiltrate as much data as possible! Don't be limited by our suggestions as possible! Extra bonus points if the extra malicious functionality is hidden / disguised / hard for an analyst with source code access to detect.
    - Modify one of your apps above (or create a new app) that will install another application. If you need the user to click on anything, then you are allowed to use clever social engineering or any other method you wish to trick the user into doing so. However, the adversarial application must initiate the install process itself; e.g., it is not sufficient to social engineer the user into going to an app store and installing a particular application from that store.
    - Sometimes malicious activity is discovered by analyzing network traffic. Make it harder for network analysts to discover your "hidden features". Enable the communications from your app to the server to use steganographic methods (in addition to the encryption and authentication that you're already applying to the data itself.). You can choose whatever approach you wish for steganography, though you will need to explain why it is secure and would likely thwart network analysis.
    - It's sometime useful to not only be able to exfiltrate data from a device, but to also change the data on the device. For example, you could plant a photo into a photo gallery, change the phone number in the contact list, or delete some

emails before they are read. Develop (and be able to demonstrate) the capability to remotely (from your server) plant new data on the phone, or change existing data on the phone. The boss thinks this can be used to "frame" persons of interest, or prevent them from realizing that they're being monitored (e.g., by deleting emails that you don't want them to read).

- Build a botnet. Create a Service that runs in the background and accepts and responds to remote commands. (Don't root your phone.) Now you can simultaneously control many phones. For example, you could type a command into the command and control channel that will make all phone report their current GPS coordinates.

- Modify an application to exhibit some other kind of malicious behavior not mentioned above. Be creative. But also play nice and follow the rules outlined in the Things Not To Do section. If you think something might not be appropriate to do, ask us first! (And wait for us to reply back with an "OK to do" before actually doing it!)

NOTE: The boss is subjective (like all human beings) and will be impressed based on the perceived amount of engineering effort and the "coolness" factor. If you have a really creative idea but a simple implementation, that can still count as a lot of effort -- sometimes you have to do a lot of thinking to come up with a really creative idea.

Turn-in

- A zip file named *username1-username2-username3*.zip where (*usernameX* is your CSE NetID). The zip file should include:
  - A directory for each task: (task0, task1, task2, task3, meta, extra).
    - Each task directory should have the correct README (see below), the source code (in a source directory) and the built apk (in a binary directory).
    - The READMEs should be named as follows: README-Task1.txt, README-Task2.txt, README-Task3.txt, README-Meta, and README-Extra. The READMEs should include the following and must use exactly one paragraph per bullet below:
      - The word "Description: ", followed by the description of how you implemented each attack/task.
      - The word "Fancy: ", followed by a description of anything 'special/ fancy' that you did for your implementation. This is your chance to convince the boss that you deserve a promotion (and us that you deserve extra credit).
      - The word "Understanding: ", followed by a discussion of why your attack works
    - The README-Meta.txt should include the following and must use exactly one paragraph per bullet below:
      - The word "Encryption: ", followed by an explanation of your

method to encrypt and authenticate the data (where do the keys come from, how do you encrypt, how do you authenticate).
- The word “Decryption: ”, followed by an explanation of your method to decrypt and verify the data (where do the keys come from, how do you decrypt, how do you verify).
- The phrase “Security Analysis:”, followed by an explanation of why your method is secure (in your discussion, be sure to include a description of which attackers is your method secure against and which attackers is your method insecure against).

■ The README-Extra should include the following and must use exactly one paragraph per bullet below:
- The word “Extra #1: ”, followed by a description of the *first* extra credit tasks you completed and what it does. (Omit this if you did not do any extra credit tasks.)
- The word “Extra #2: ”, followed by a description of the *second* extra credit tasks you completed and what it does. (Omit this if you did not do a second extra credit tasks.)
- The word “Extra #3: ”, followed by a description of the *third* extra credit tasks you completed and what it does. (Omit this if you did not do a third extra credit tasks.)
- (Repeat the above “Extra #N” notation for extra credit task 4, 5, 6, and so on.)

○ A directory for your server application.
■ Please include the source for your server application along with a README describing what the code does, how to run it, and how to use it (and anything cool or neat that it does).

Grading

Your boss will mentally give you:
- 5 reputation points for completing Task 0
- 10 reputation points for completing Task 1
- 15 reputation points for completing Task 2
- 15 reputation points for completing Task 3
- 15 reputation points for completing the Meta Task

We’ll follow the boss’ example when computing your grade

Live demos

Your boss is demanding. He or she wants to see a demo of your work -- live and in person. You will be asked to sign up to present a live demo of your Part 1 results on May 30, May 31, or June 1. You will have at most 15 minutes to present, so rehearse. We strongly recommend practicing your demo long before you actually present so that you have time to fix any glitches in advance. A large part of your grade will depend on what we see in the demo -- if we didn’t see it, we’ll assume it doesn’t exist or doesn’t work.

Part 2

Due (Task 1 and Task 2 are required): Friday, June 1st, 5pm (no late submissions)

Due (Tasks 3 is optional/extra credit): Wednesday, June 6th, 5pm (no late submissions)

Background

Two persons of interest were careless in keeping track of their phones. One of our spies has risked his life, but was able to dump data from their phones. These data dumps have been delivered to you for analysis. Get the the data dump here: http://www.cs.washington.edu/education/courses/cse484/12sp/projects/project3/phones.tar.gz

Tasks

You must learn as much as you can from these data dumps.

- Task 1 (required): Extract general information that will help us understand the type of phones these were and will help the agency craft 0-days. Get the:
  - Phone number of the phone
  - Name and gmail account of user
  - Phone Information
    - Model
    - Brand
    - Name
    - Device
    - Board
    - CPU
    - Manufacturer
    - IMEI
    - Serial Number
  - Operating system
    - Type
    - Custom ROM or stock?
    - Version
    - Build date
  - Carrier
  - List of applications installed
    - Which came with the system
    - Which did the user install?

- Task 2 (required): Extract any PII that you can find from the phone. Get the:
  - Phone numbers called from this phone
  - E-mail addresses of people e-mailed from phone
    - What are the contents of the e-mail messages?
  - Numbers to which SMSs have been sent
    - What are the contents of the SMS messages?
  - Recover the list of Wifi Networks to which the device has connected, if there were any passwords used, recover these.

- Task 3 (optional, extra credit): Extract other information:
  - What method does the target use to unlock their phone (PIN, pattern, password, etc...)? Is there an alternate mechanism? What is it? If there is a password salt, what is it?

    - What was the voice volume level?
    - What was the alarm volume level?
    - What was the screen timeout set to?
    - Was the wifi on?
    - Was bluetooth on?
    - If the phone has a second factor application installed (like Google Authenticator), extract all of the key material.

- Additional Extra Credit Task:
    - Crack the screen unlock mechanism (i.e., crack the password/pin/pattern)
    - Location:
        - What locations has the target been to?
        - What terms does the target search for with respect to locations?
    - Media
        - Can you recover any images taken by the camera(s) or downloaded from the internet? (images that are part of the OS or downloaded as part of an application don't count).
    - Wifi MAC address
    - Bluetooth MAC address
    - Anything else you think might be useful

Turn-in (Task 1 and 2)

A zip file named *username1-username2-username3-Part2Task1*.zip where (*usernameX* is your CSE NetID). The zip file should include a directory for each task: just task1 and taks 2 in this case. Each task directory should have a text file the appropriate text file named as follows: ANSWERS-Task1.txt and ANSWERS-Task2.txt. The ANSWERS files should include the following and must use no more than one paragraph per bullet below:

- The word "Question: ", followed by the question you were trying to answer .
- The word "Answer: ", followed by the answer to this question.
- The word "How: ", followed by the explanation of how you obtained the answer.

Turn-in (Tasks 3, and Extra)

A zip file named *username1-username2-username3-Part2Task+*.zip where (*usernameX* is your CSE NetID). The zip file should include a directory for each task: (task3 and extra). Each task directory should have a text file the appropriate text file named as follows: ANSWERS-Task3.txt, and ANSWERS-Extra. The ANSWERS files should include the following and must use no more than one paragraph per bullet below:

- The word "Question: ", followed by the question you were trying to answer .
- The word "Answer: ", followed by the answer to this question.
- The word "How: ", followed by the explanation of how you obtained the answer.

Grading

Your boss will award you 36 reputation points for completing each of the three tasks (12 points per task) -- we'll award you the same. Recall that only tasks 1 and 2 are required.

Things Not to Do

- Don't release / redistribute your malware; we encourage you to keep all your copies encrypted. Don't write or use malware that overwrites the firmware.
- Don't "unlock" the bootloader
- Don't "brick" the phone
- Don't sell the phone
- Don't do anything illegal with the phone
- If you're not sure if you should do it or not, ask first and then only do it if we reply saying that you can.

Getting Started – Android Development.

Follow the following steps to get the Android system up and running on your machine:

- Download the Android SDK: http://developer.android.com/sdk/index.html.
    - Install the following components from the SDK manager:
        - Tools
        - Android 4.0.3 (API15)
        - Google USB Driver
    - NOTE: We had some issues installing the 'Google USB Driver' component on Windows and chose to use a different tool (http://junefabrics.com/android/) to install the required drivers.
- We recommend using Eclipse for development. Other IDE are an option, but we highly recommend Eclipse. Our instructions will continue assuming Eclipse is your choice
    - Install the ADT plugin for Eclipse.
        - Instructions: http://developer.android.com/sdk/eclipse-adt.html.
- You now have everything necessary for development, but you need to make a few updates to the Android phone itself.
    - Enable USB debugging:
        - Click on Settings → Developer options
        - Enable USB debugging option
    - Enable installation of non-market apps:
        - Click on Settings → Security
        - Enable Unknown sources option

The goal for this lab is not to teach you how to design mobile apps. Feel free to download open-source android applications to work with. The following links are either repositories to open-source apps or precompiled projects with a *.apk file. The *.apk file is required to install your android apps on your phones.

The app shown as a demo in the quiz section was pulled from the following link. I would recommend you start from here:

- http://f-droid.org/repository/browse/

Some more apps are located here:

- http://code.google.com/p/apps-for-android/

Open Source Apps (APKs Excluded):

- **Amazed**: A simple but addictive accelerometer-based marble-guidance game. (you'll need to remove R.java in src)
- **BTClickLinkCompete**: Bluetooth compete game.
- **DivideAndConquer**: a game in which you must isolate bouncing balls by creating walls around them. (you'll need to remove R.java in src)
- **Panoramio**: An app that shows you nearby photos and points of interest. (It seems that map is not working due to it been outdated)
- **WikiNotes**: A wiki note pad that uses intents to navigate to wiki words and other rich content stored in the notes.

The following apps have their application packages (apk files) provided:

- **SuperGenPass**: http://staticfree.info/projects/sgp/
- **PedoMeter**: http://code.google.com/p/pedometer/
- **K-9**: http://code.google.com/p/k9mail/
- **ConnectBot(SSHClient)**: http://code.google.com/p/connectbot/

Once you have the source, you can compile and build the open-source projects within Eclipse to obtain a *.apk file within the bin folder of your project. This is the *.apk file that you will need to load onto the phones to run your program. Alternatively you can directly use the provided *.apk files from the compiled apps provided earlier.

Finally, use either Eclipse or the adb tool (located within the ANDROID_SDK_HOME\platform-tools folder) to install your app on your phones. To use the adb tool, run the following in a command window:

- adb devices
- adb install [package name ending in *.apk]

Note: In the event that the app has previously been installed, make sure you either directly uninstall the app from the phone or use the adb tool as illustrated below before installing your newly built app:

- adb uninstall [package name ending in .apk]

REMINDER: The internet is a great place to find help, especially with setting up your environments or getting something running on your phone. You will probably be able to get an answer online faster than you would from the staff (though feel free to email us if you'd like).