CS 577 Cybersecurity Lab

Stevens Institute of Technology

Lab 6 – due 10/23/14 6:16pm

Instructor Georgios Portokalidis

Teaching Assistant Dimitrios Damopoulos

This assignment will help you to build your first android application, decompile/edit/recompile a legitimate android app, reverse engineer malicious android applications, and finally designing a malware detection tool for Android apps.

This lab will be done on your personal computer.

To develop android application, you will need to install into your system: the Java and Android SDK, Eclipse as the developing IDE and the Android Developer Tools.

Android SDK is open source and available for every OS.

Follow the instruction to install the Android SDK into your systems:

http://developer.android.com/sdk/installing/index.html?pkg=adt

Additional tools and information

Apktool: https://code.google.com/p/android-apktool/

Smali debugging: https://code.google.com/p/androidapktool/wiki/SmaliDebugging

dex2jar: https://code.google.com/p/dex2jar/

JD Decompiler: http://jd.benow.ca/

jarsigner or keytool:
https://www.owasp.org/index.php/Signing_jar_files_with_jarsigner

Android Security Overview: https://source.android.com/devices/tech/security/

System Permissions:
http://developer.android.com/guide/topics/security/permissions.html

Android Security

Aim of this deliverable is to understand how basic Android applications are build, how easy is to modify the Android application package (APK) in order to inject malicious behaviors into a legitimate application. Also you have to go one step further and analyze 5 popular Android malware understanding and reporting the malicious functions. Finally using the gained knowledge, you will have to build a signature-based malware detection system, able to analyze statically an application and define if it is malicious or not.

Exercise 1. Build your first app (20%)

Create your first android application. This application will be the typical "Hello world" for the Android OS.

You'll learn how to create an Android project and run a debuggable version of the app. You'll also learn some fundamentals of Android app design, including how to build a simple user interface.

Exercise 2. Decompile/Edit/ Recompile (20%)

Having compiled and packaged your first Android application ("Hello world") into an APK file, in this exercise you have to use one of the available tools and technics to modify it. Once you reverse engineer apk and retrieve the source code, you have to change the "Hello world" phrase into, "Hello world, Your_names". Finally recompile the application, install the application into the emulator and execute it. Report the results.

Document every step of the process. Keep in mind that in this exercise you will ***not*** use the source code you wrote in the 1st exercise, but you will retrieve it directly from your APK.

Exercise 3. Reverse engineering Android applications (30%)

Reverse engineer 3 android malwares to identify and document the malicious code, the required permissions, and report if the malware is able to exploit the android OS and gain root privileges.

Explain the malicious behavior by understanding the source code and not by executing it.

Can these malwares be categorized into families based on the malicious behavior and same source code?

Keep in mind that you don't have to execute the malware. In case you do it, please use a new emulator only for the experiment.

Exercise 4. Signature-based Malware detection systems (30%)

Having identified the malicious source code in exercise 3, create malware signatures. Moreover, design and implement an application able to automatically analyze an APK and search into the source code for the malicious signature. This way you will be able to detect malwares. The application should be analyzed statically and not dynamically.

Using your tool, analyze and define the legitimate and malicious android applications and report the performance and accuracy of you tool.

Please, keep it simple!

Bonus. Employ machine learning to detect malwares (20%)

Expand your previous detection system by providing a detection engine based on machine leaning. Use WEKA in order to build the behavior profile that includes both legitimate and malicious source code.

Explain how you build and evaluate your detection system. Test it with both legitimate and malicious application. Does your system identify zero-day (unknown to the training phase) malwares?

Grading

As always, we will grade your work on quality from both the user's and programmer's points of view. Each program should contain function-level and local comments as appropriate, as well as an explanation of the program's principles of operation. **PLEASE SUBMIT**: ***Documentation***, ***your source code and the 2 APK you create for the first exercise***, ***a report with all the steps and problems you had***.