Android App逆向工程與簽章技術

Joey{27350000 @ hst.tw}

這場演講「三不一沒有」

三不一沒有

- 三不
  - 我不是駭客（我是魯蛇）
  - 我不會Android
  - 這場演講不難，大家都聽得懂
- 一沒有
  - 這場演講沒有太多梗，有的話幫幫忙笑一下

About Me

- Joey Chen
- 目前
  - 國立臺灣科技大學資管所 ~~菸酒生~~ 研究生
  - Hack-Stuff Technology Core member
  - 趨勢科技實習生
- 證照
  - ISO 27001:2013
  - BS 10012:2009
- 經歷
  - 2012年駭客年會第三名
  - 2013年駭客年會第二名
  - 2014年Honeynet CTF 第二名
- 精專於
  - 加解密與數位簽章
  - 逆向工程

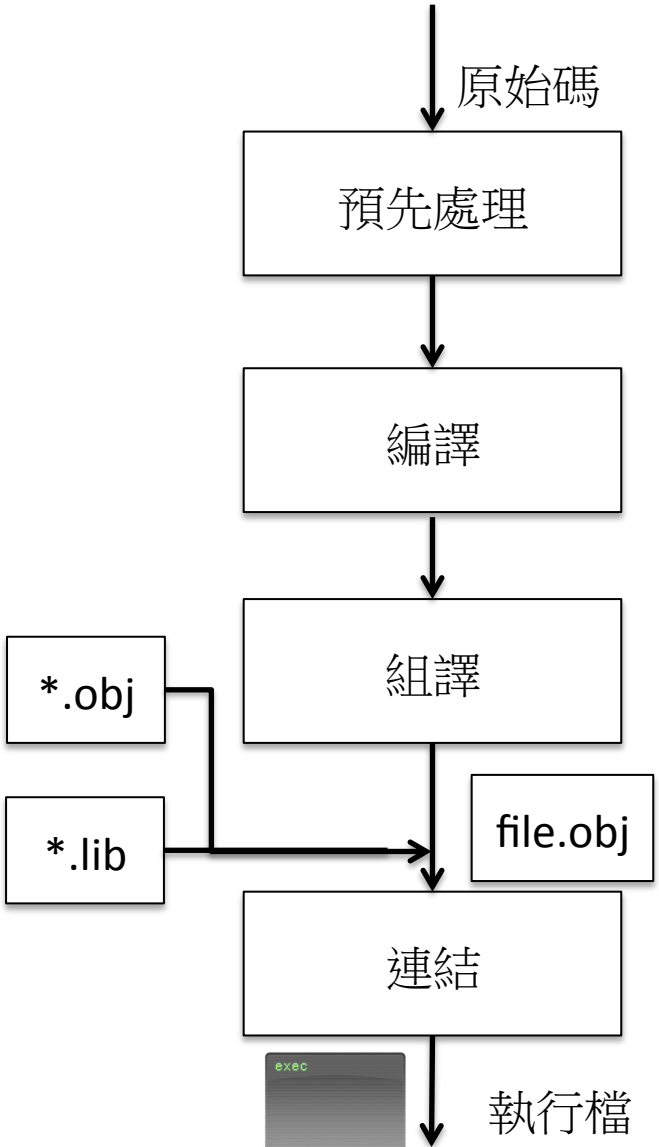
演講重點

- 今日演講不再「技術」
- 演講重點在於「努力」
- 希望大家找到「熱情」

大綱

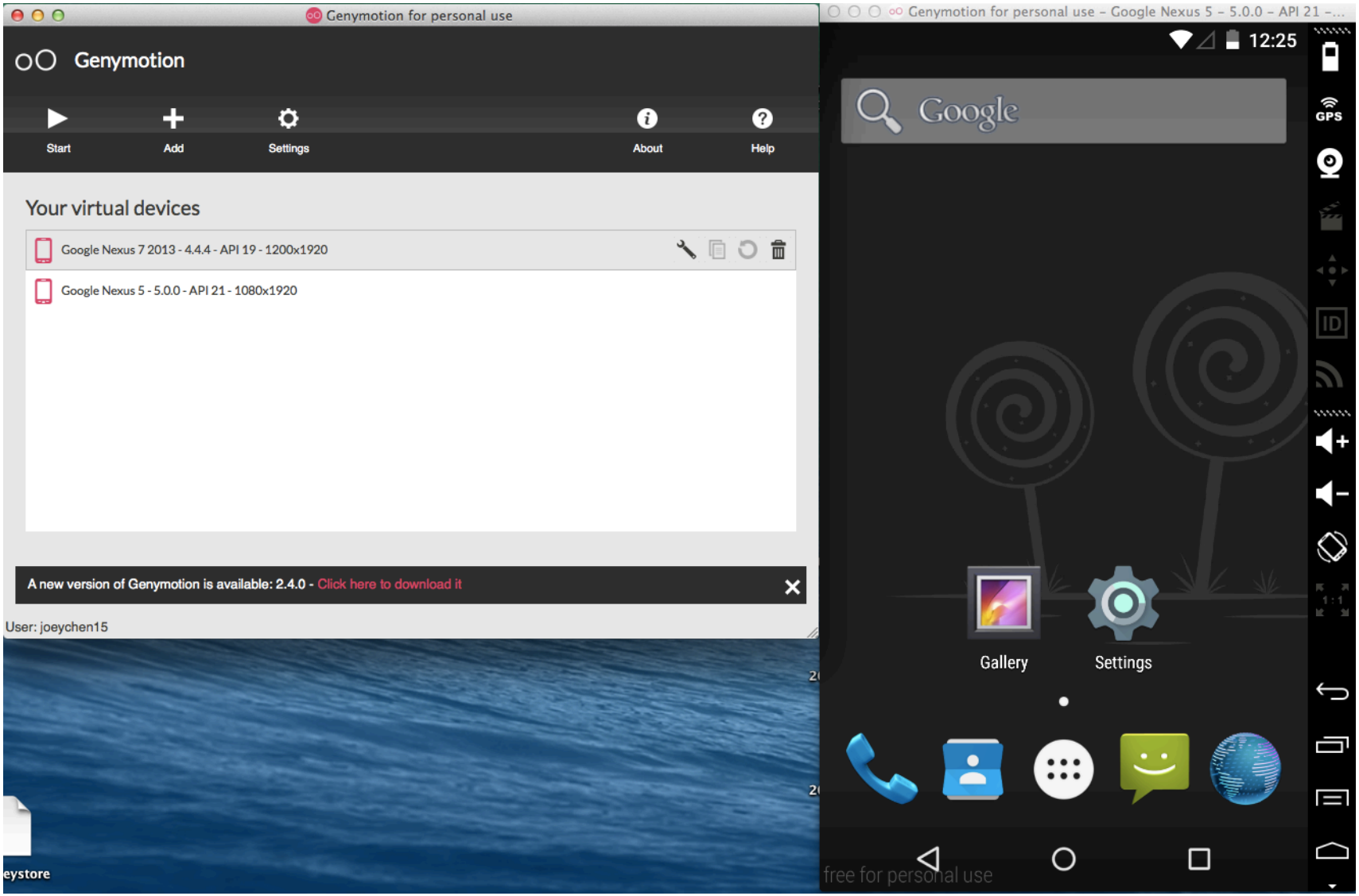
- 逆向工程
- 簽章技術
- 工具介紹
- Android OS 介紹
- 拆解第一個 App
- 修改第一個 App
- 保護 App
- 結論
- Q&A

預備知識(1) - 逆向工程

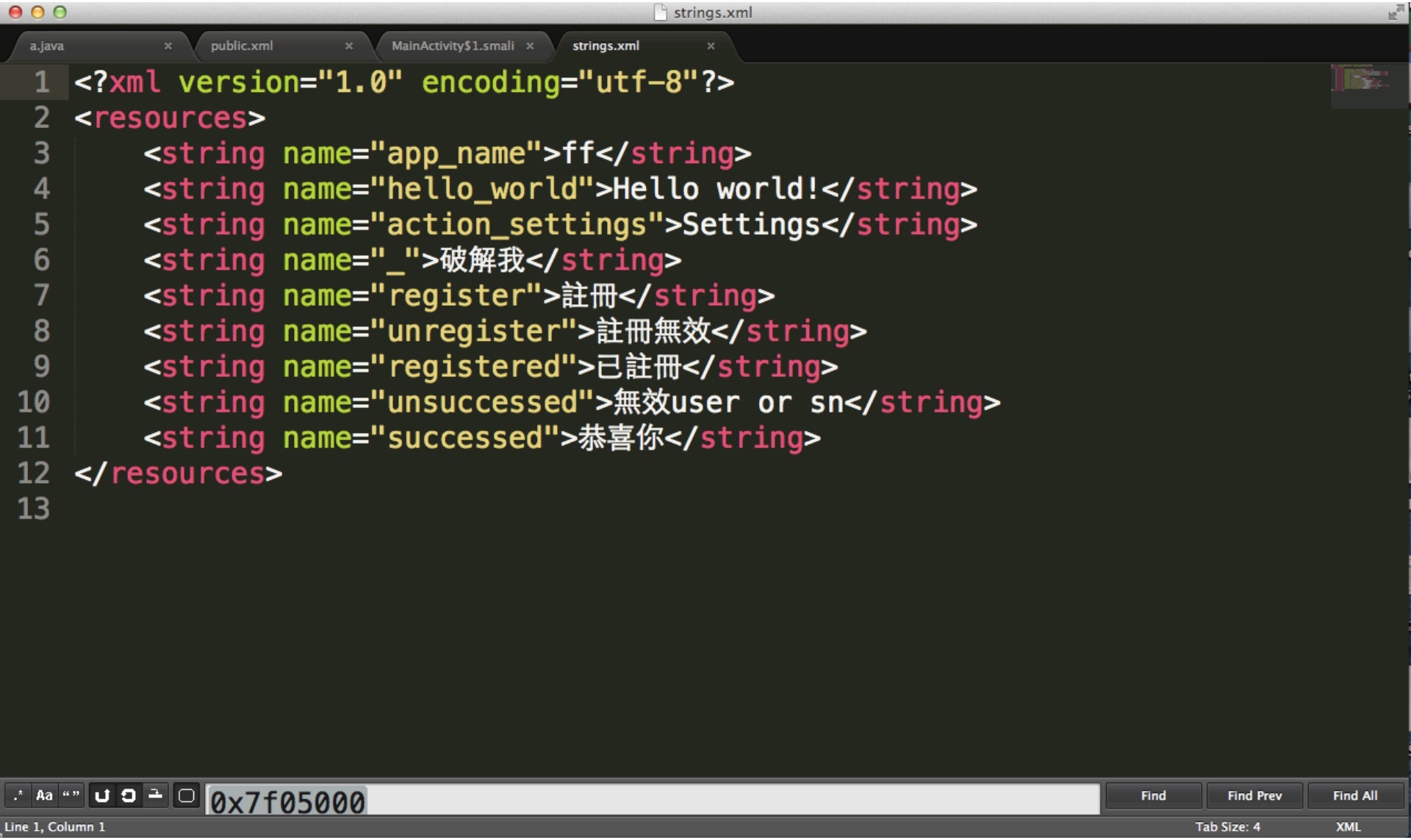


- Compiler V.S Reversing Engineer
- 直譯式 V.S 編譯式
- 程式語言的不同, ex :
  C/C++, .Net , Java…
- File format 的不同, ex :
  *.exe, *.dll, *.jar…

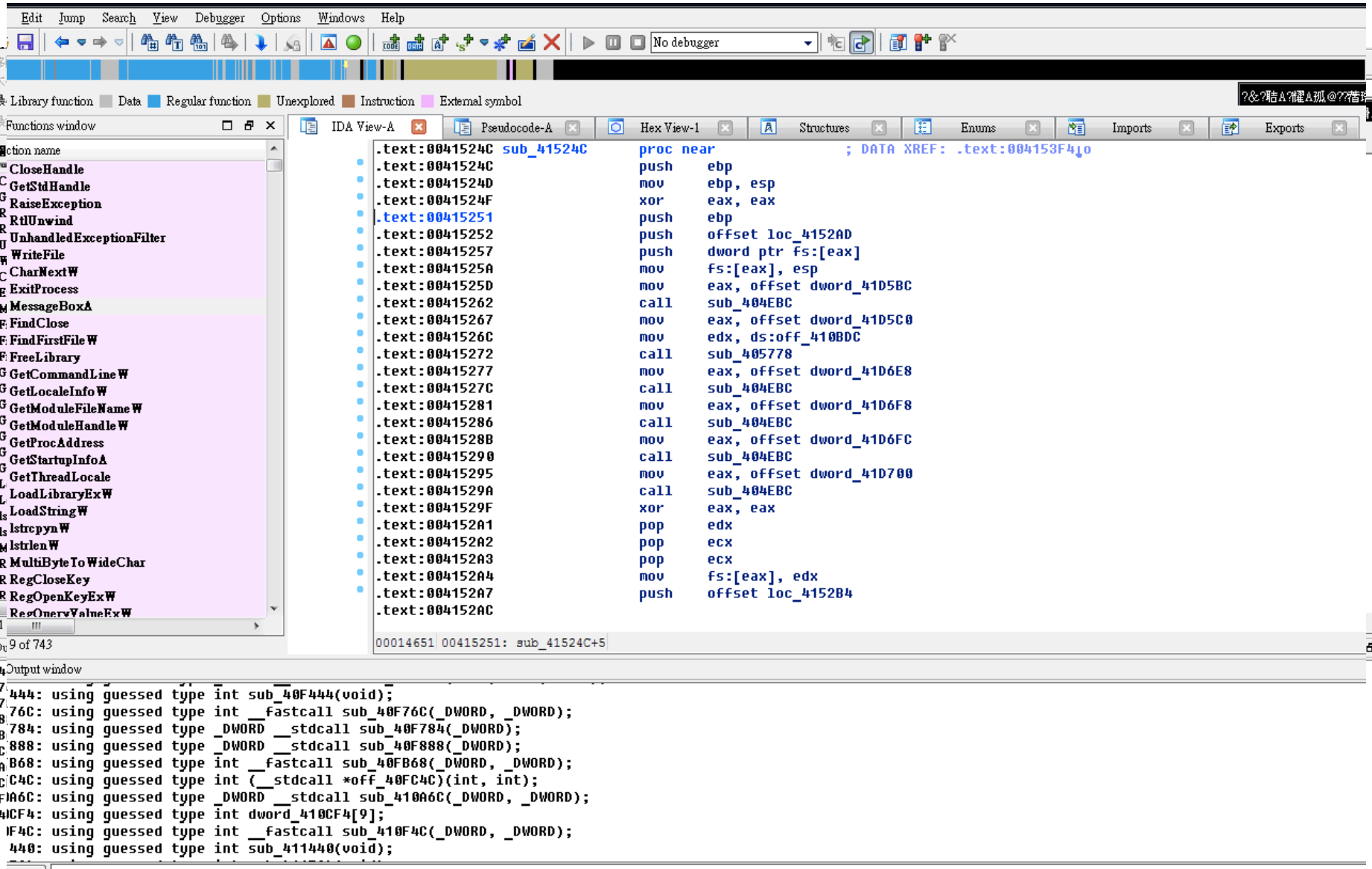
工具介紹(1) - Genymotiom



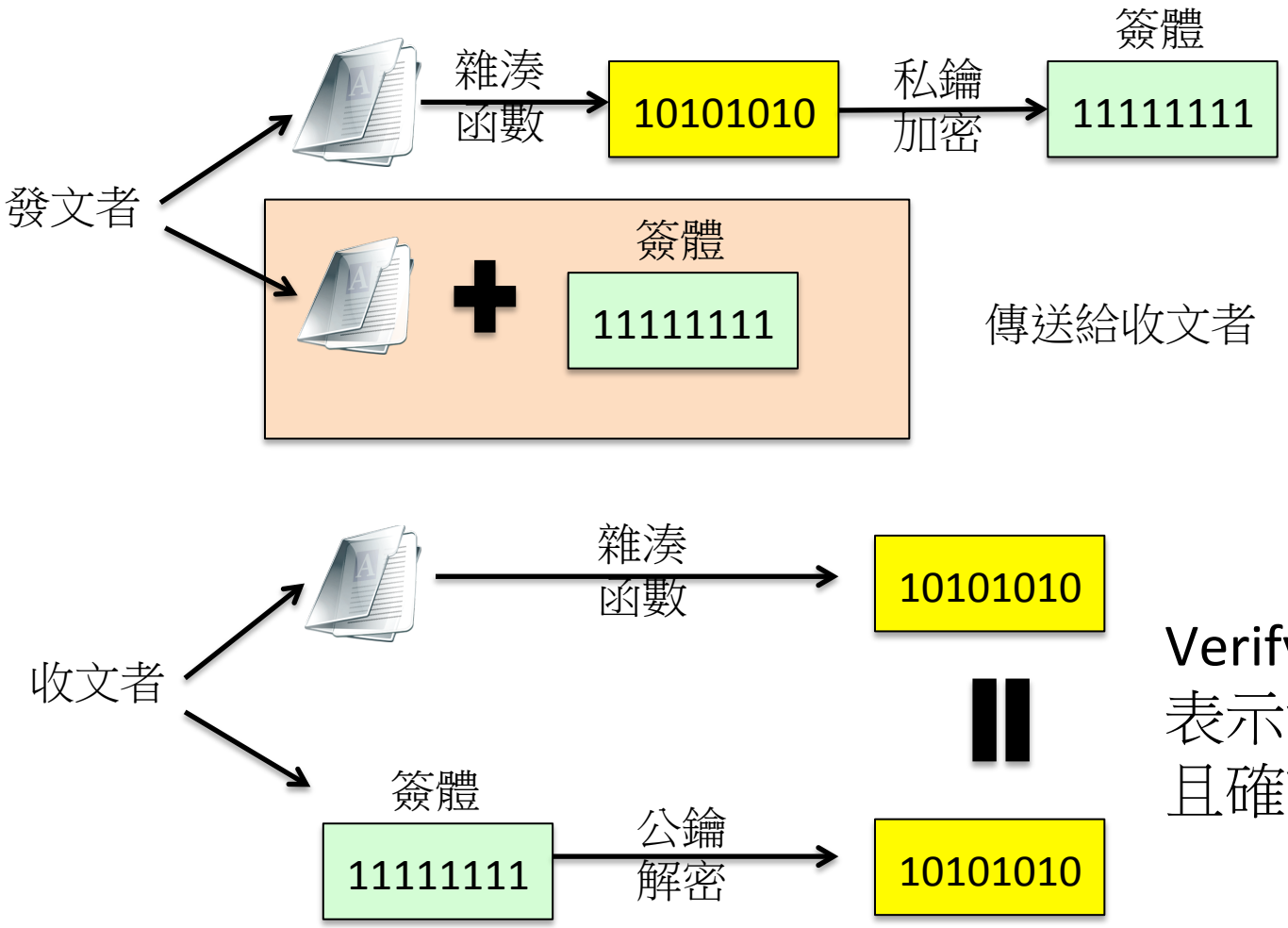
工具介紹(2) - Sublime



工具介紹(3) - IDA pro



預備知識(2) - 數位簽章

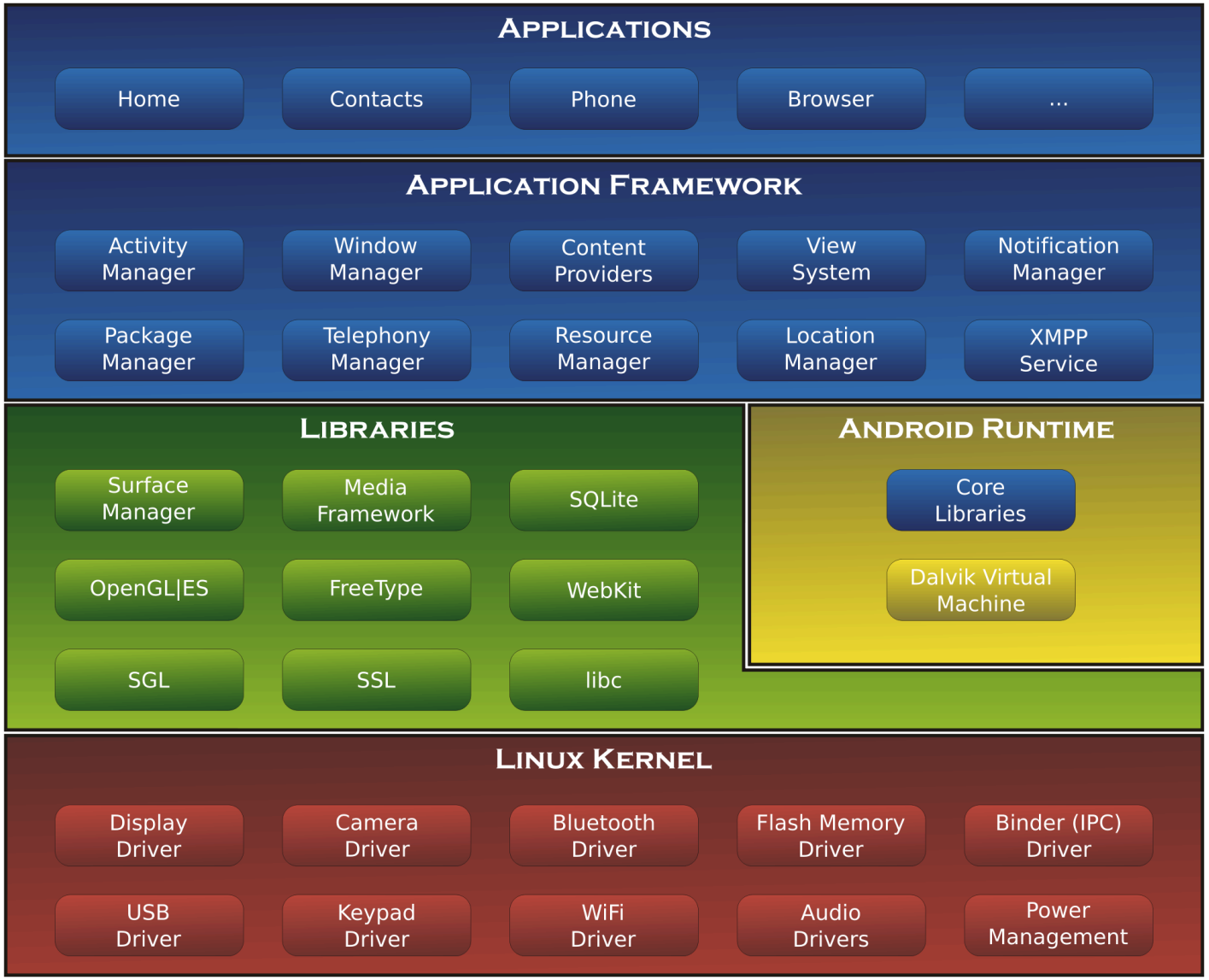


Verify ? Yes : No

表示資料未被篡改

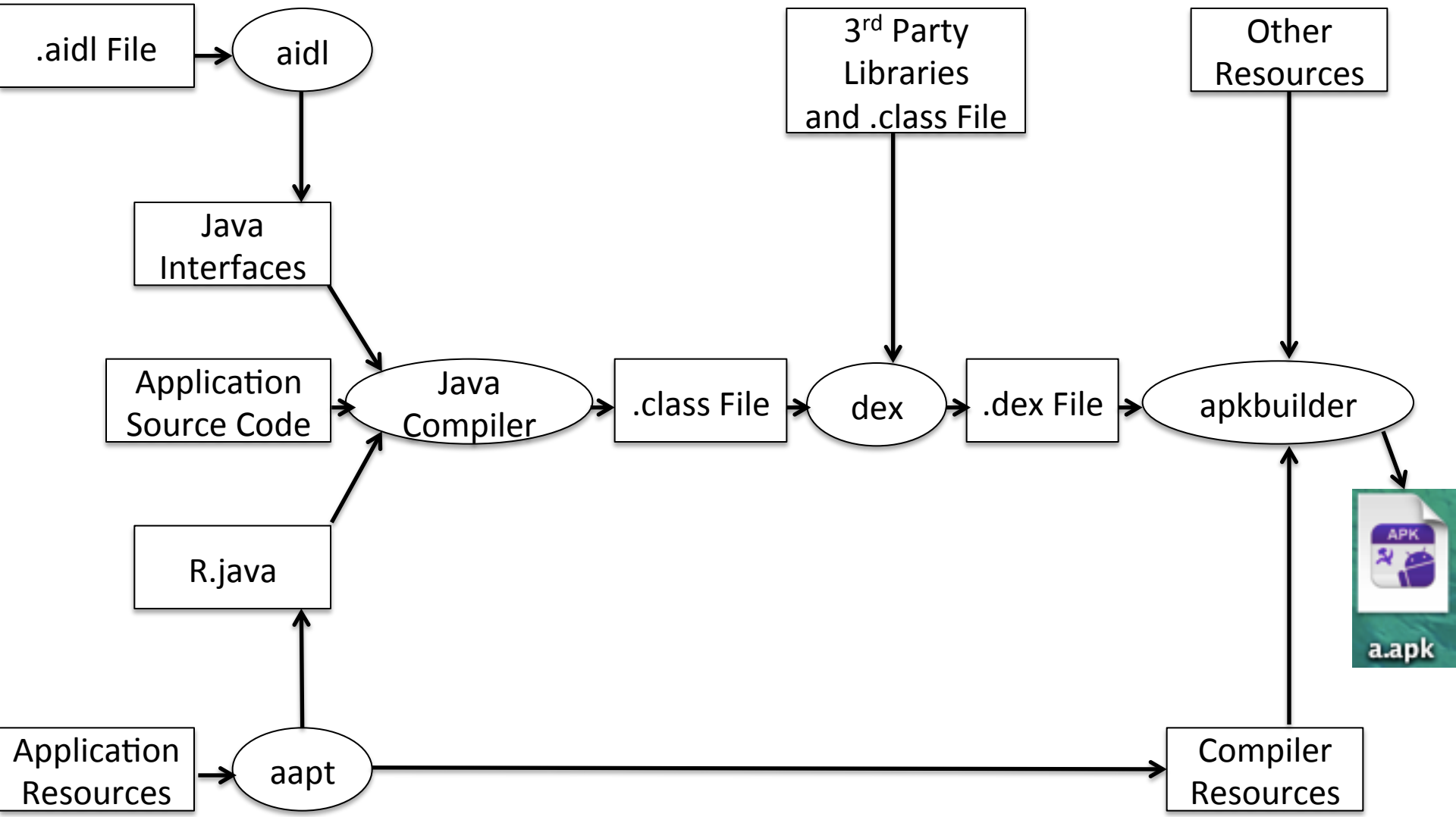
且確認發文者身份

Android OS

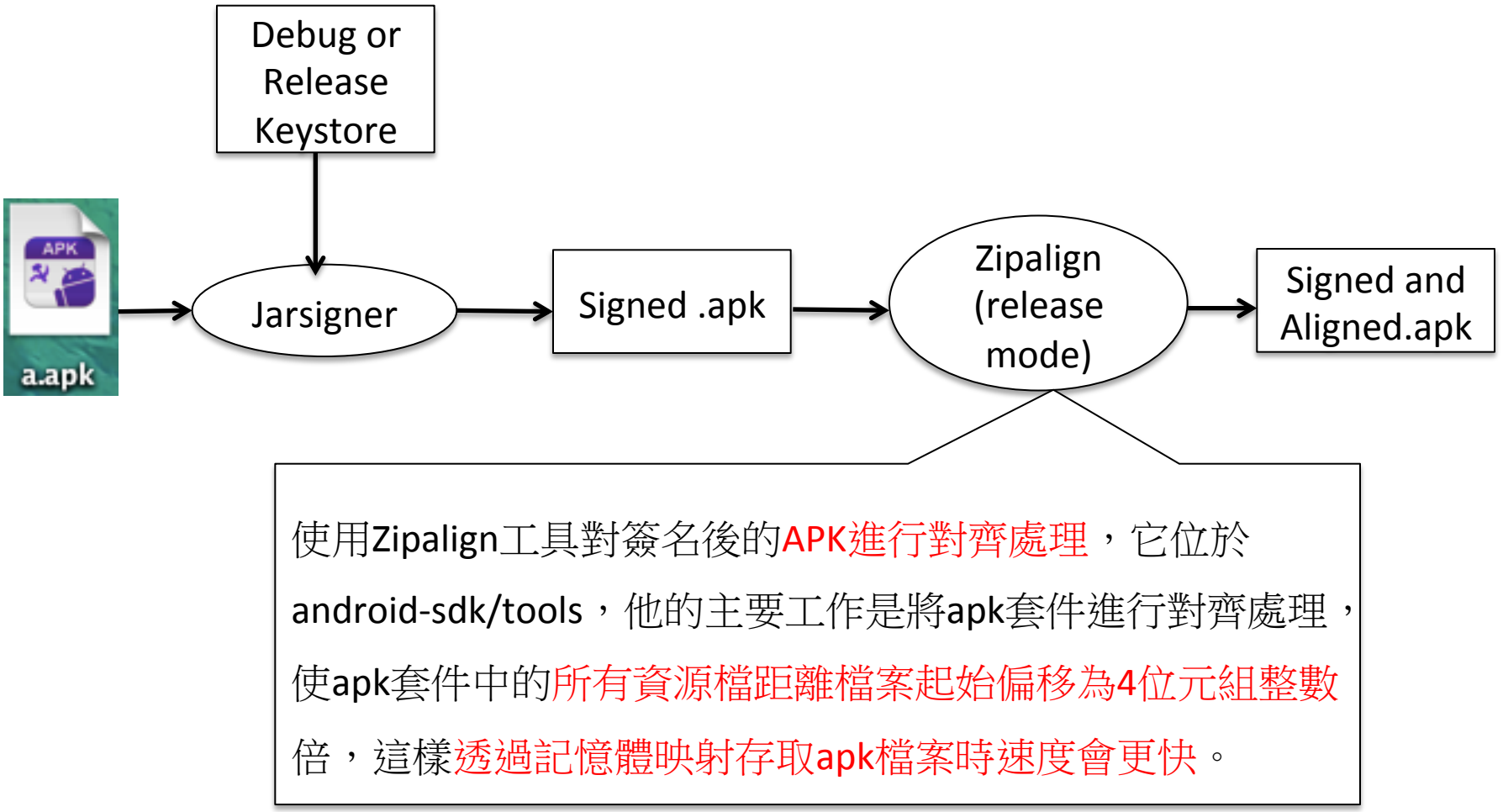


資料來源：http://en.wikipedia.org/wiki/Android_(operating_system)

APK 包裝過程


.aidl File
aidl
3rd Party Libraries and .class File
Other Resources
Java Interfaces
Application Source Code
Java Compiler
.class File
dex
.dex File
apkbuilder
R.java
APK
a.apk
Application Resources
aapt
Compiler Resources

APK 簽章過程


Debug or Release Keystore
APK
a.apk
Jarsigner
Signed .apk
Zipalign (release mode)
Signed and Aligned.apk
使用Zipalign工具對簽名後的APK進行對齊處理，它位於android-sdk/tools，他的主要工作是將apk套件進行對齊處理，使apk套件中的所有資源檔距離檔案起始偏移為4位元組整數倍，這樣透過記憶體映射存取apk檔案時速度會更快。

破解第一支 APP

- 破解工具
  - Apktool, dex2jar, JD-GUI, IDA pro, androguard
- 環境
  - OSX, windows, linux
- 開發工具
  - Eclipse, Android SDK, Android NDK, genymotion
- 先備語言
  - Java, C/C++, .Net

檢視 APK 檔案格式

- Apk : Android 應用程式套件檔案
  - META-INF
    - MANIFEST.MF : 清單資訊（Manifest file）
    - CERT.RSA : 儲存著該應用程式的憑證和授權資訊。
    - CERT.SF : 儲存著 SHA-1 資訊資源清單
  - Lib : 已編譯好的程式
  - Res :不需要被編譯的檔案, ex : *.png, *jpg …
  - Assets :用於存放需要打包到應用程式的靜態檔，以便部署到設備中, ex : 政策條款等等…
  - AndroidManifest.xml :傳統的Android清單檔案，有該應用程式名字、版本號、所需權限、註冊服務、連結的其他應用程式名稱
  - classes.dex : classes檔案通過DEX編譯後的檔案格式，用於在Dalvik虛擬機器上執行的主要代碼部分
  - resources.arsc : 有被編譯過的檔案, ex : 被編譯過的xml

Apktool : A Tool for Reverse APK

- Features
  - Disassembling resources to nearly original form (including resources.arsc, classes.dex, *.png. and XMLs)
  - Rebuilding decoded resources back to binary APK/JAR
  - Organizing and handling APKs that depend on framework resources
  - Smali Debugging
- Requirements
  - Java 7 (JRE 1.7)
  - Basic knowledge of Android SDK, AAPT and smali

資料來源：http://ibotpeaches.github.io/Apktool/

Smali V.S Classes.dex

- *.smali
  - 一種組合語言
  - 極為貼近 Dalvik VM 所接受的 DEX 檔案的組合語言形式
  - 不同於Java
  - 使用apktool 才會出現
- Classes.dex
  - Java 位元的 Binary code
  - Android使用的dalvik虛擬機器與標準的java虛擬機器是不相容的
  - dex檔與class檔相比，不論是檔結構還是opcode都不一樣
  - 使用unzip才會出現

神器：dex2jar, JD-GUI

- Dex2jar
  - 將*.dex轉換成*.jar
  - 其他功能 ex : d2j-apk-sign.sh, d2j-jar2dex.sh, d2j-asm-verify.sh, d2j-jar2jasmin.sh, d2j-decrpyt-string.sh, d2j-jasmin2jar.sh, d2j-dex-asmifier.sh, dex-dump.sh, d2j-dex-dump.sh…
- JD-GUI
  - JD-GUI 可反編譯*.jar，將 java 原始碼還原
  - 透過 code review 理解程式的邏輯與重要的 function

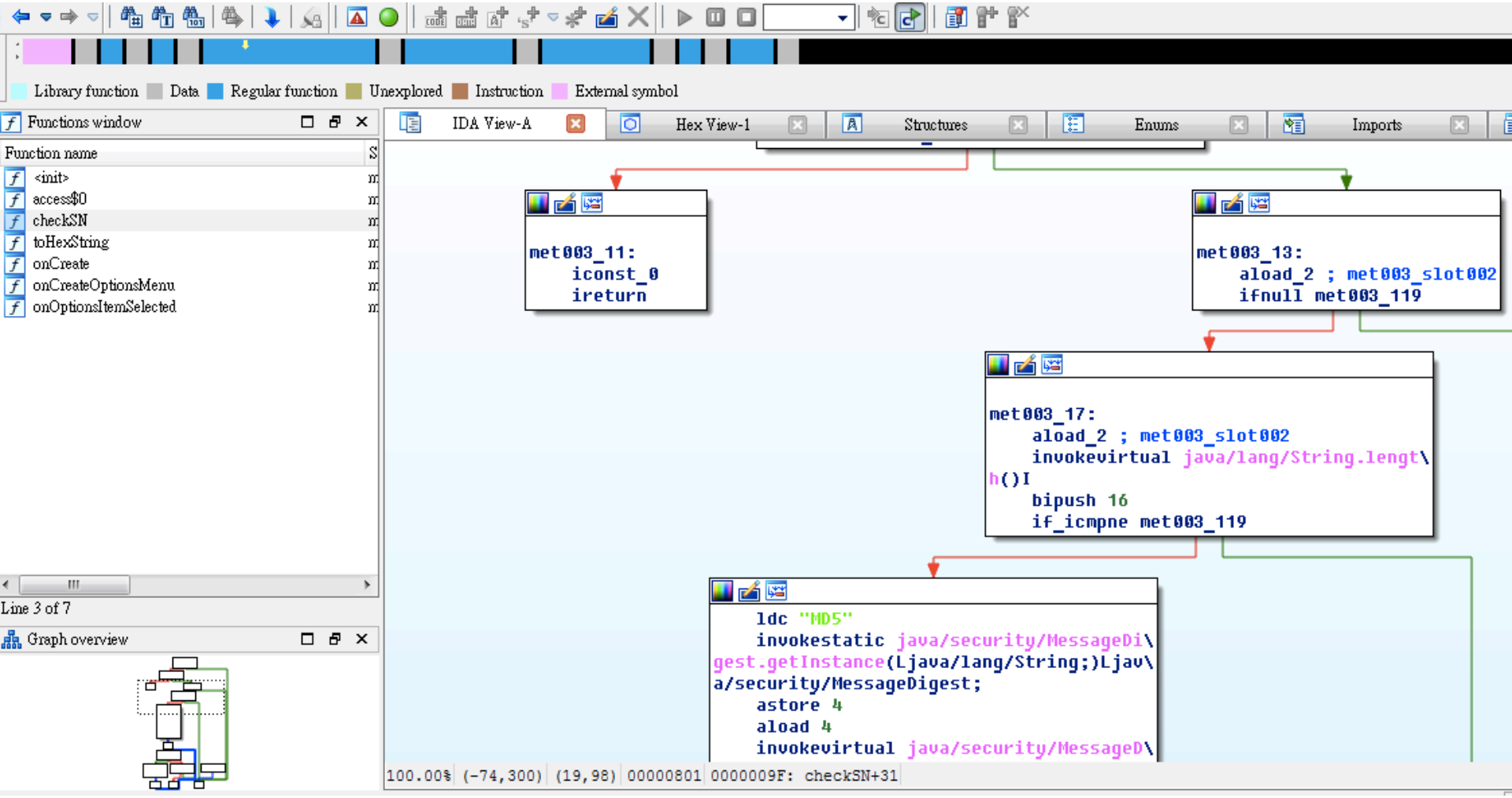
~~篡改~~修改 APK 的方法

- 只能修改 smali, 因為原生語言無法重新編譯回 *.jar ，但是修改 *.smali 他只需要拉進 IDE 修改組合語言的部分，存檔再重新打包 Apk
- 今天不是來教大家看組合語言
- 今天希望大家能夠理解程式邏輯、系統邏輯以及檔案格式

Smali 的跳躍指令

- “if-testz vAA, +BBBB” 條件跳躍指令。拿 vAA 暫存器與0比較，如果比較結果滿足或值為0時就跳躍到 BBBB 指定的偏移處。偏移量 BBBB 不能為0。
- “if-eqz” 如果 vAA 為0則跳躍。Java 語法表示”if(!vAA)”
- “if-nez” 如果 vAA 不為0則跳躍。Java 語法表示”if(vAA)”

程式邏輯(科技始終來自於人性)



重新打包回APK

- Apktool b ff（打包剛剛反編譯出來的資料夾）
- 但是Apk尚未簽章
- jarsigner -verbose -keystore rdss.keystore -signedjar ffx.apk ff.apk rdss

深入淺出 META-INF

- 比較發現 RDSS.SF 比 MANIFEST.MF 多了一個 SHA1-Digest-Manifest 的值，這個值其實是 MANIFEST.MF 文件的 SHA1 接著用 base64 編碼的值，可以手動驗證，也可以從 android/tool源碼分析。
- SHA1-Digest-Manifest 是 MANIFEST.MF 文件的 SHA1 接著用 base64 編碼的結果。

保護APP(1)

- Isolate Java Program
  - 將較敏感或重要的 class 檔案放在 server 中，利用動態載入的方法，來避免駭客去反編譯整支程式
- Encrypt Class File
  - 使用 AES, DES… 去加密重要的 class 檔案，使駭客即使反編譯，也只能看到亂碼
- Convert to Native Codes
  - 利用 Android NDK 在 Project 裡面寫 C/C++ 使得反編譯的易讀大大提升難度，更別提程式中的加密演算法

保護APP(2)

- Code Obfuscation
  - 可以使用”ProGuard”通過移除不用的代碼，用語義上混淆的名字來重命名分類、欄位和方法等手段來壓縮、優化和混淆你的代碼
- Online Encryption
  - http://sourceforge.net/projects/apkprotect/ 這個網站提供了跨語言（Java, C/C++）的保護，可以達到反編譯反組譯的的功能
  - https://dexprotector.com/node/4627 直接針對 *.dex 檔案進行保護，加密檔案、訊息，隱藏 function call ，確保完整性

結論

- 學習 Android 逆向，一步一步來，從根本出發，從檔案、系統、格式都要理解
- 學習工具的使用，再強的高手都要使用工具來幫助自己，減少時間的耗費
- 學習簽章、密碼學，資訊安全最後的一道防線，系統會有漏洞，人為會有疏失，密碼系統雖不保證絕對但增加了一定的難度
- 學習不要心急，經驗與知識是靠累積，大家一起努力，加油

延伸議題 and Q&A

- 一般 Apk 的簽章期限多久？
- Update Apk 需要一樣的簽章嗎？
- 利用 jarsigner 簽的Apk跟之前的簽章相同嗎？
- 重新打包 Apk 會有風險嗎？
- 可以在不反編譯或解壓縮的情況下塞檔案進 Apk 嗎？