使用靜態分析與機器學習以偵測 Android 惡意程式

蘇民揚*　　張文銓
銘傳大學資訊工程學系
***minysu@mail.mcu.edu.tw　irons7654321@gmail.com**

摘要

相對於個人電腦上已有的安全防護，智慧型手機是新興的平台，其安全防護措施比電腦薄弱。Android 智慧手機目前的使用者對於行動通訊資訊安全的意識薄弱，往往沒看清楚軟體的要求權限，就跳出視窗按確定，以至於手機被植入病毒都不知道，而在 Android Market 中，也有很多惡意軟體會偽裝成遊戲或者圖片被使用者下載，然後再進行如惡意消費、手機資源消耗、協助犯罪，或者竊取資料等等，所以本篇研究將著重在防範惡意程式被安裝在 Android 智慧型手機上。研究內容為透過 Android 應用程式的許可權(permission)與程式碼函式來分析是否為惡意程式，並利用基於機器學習的加權分析來提高精確度。

關鍵詞：Android、智慧型手機、安全、許可權、機器學習。

1. 前言

這幾年來智慧型手機等行動設備的發展相當迅速，並結合了 Wi-Fi 和 3G 無線網路的技術使得手機的用途大大的提升，但在使用者享受手機所帶來的便利之餘，網路所帶來的威脅也漸漸影響到日漸強大的智慧型手機。

智慧型手機是一種類似個人電腦，具有獨立的作業系統，並且可以通過自行安裝或官方更新各種軟體程式，像是許多工具與遊戲等等。通過這些程式可以不斷的對智慧型手機進行功能擴充，而且也可以通過行動網路(wifi、3G 等)來進行網路行為，這一類型的手機就統稱為智慧型手機。智慧型手機的作業系統有相當多種，常見的作業系統為: Nokia 的塞班(Symbian)、微軟作業系統(Windows phone)，蘋果作業系統(iOS)、Linux 相關作業系統(含 Android, WebOS 等)、黑莓作業系統(BlackBerry OS)。

現今智慧型手機的主流，非 Google 的 Android 和 Apple 的 IOS 莫屬，其中 Android 手機最大特色在於其作業系統屬於開放原始碼，讓各個公司都能自行衍生、創作，且每位開發者所撰寫的程式都能發佈到 Market 供人下載，讓使用者能安裝各種不同的軟體客製化屬於自己的手機。相對其他智慧型手機系統，Android 系統具高度自由，在 linux 或 windows 主機上都可以進行開發，且在實體手機上執行也無須額外花費，只有在打算將完成的程式上架時，才需要購買開發者帳號。Android 開發者帳號只要 25 美金，帳號期限為無限期，間接導致免費軟體較多，其軟體豐富性深受使用者的喜愛，目前為市場占有率最高的作業系統，而且銷售量也不斷的增加。

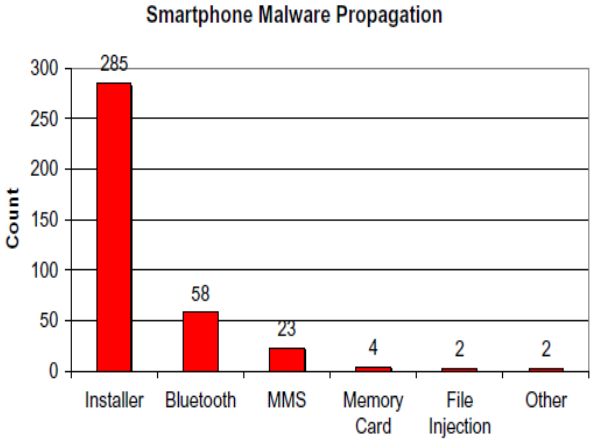

圖 1：智慧型手機惡意程式傳播途徑[1]

本研究的主要目的是要防範惡意程式被安裝在 Android 智慧型手機上，如圖 1 顯示了惡意軟體傳播方式，由此可知，大部分惡意軟體都是使用者在不知道的情況下安裝的。Android 系統的安裝檔為 APK(全名為 Android Package)，任何程式都需要封裝成副檔名為 .apk 的檔案，在經過使用者允許後安裝在手機上，因此即使下載到惡意程式檔案，只要不被安裝到手機上，該惡意程式就不會造成威脅。本篇研究根據許可權(permission)組合來分析使用者欲安裝的程式是否為惡意程式，並利用基於機器學習的分析來提高精確度。這將使得使用者可以在第一時間阻止惡意程式下載與安裝。

2. 相關研究

Android 由於開放原始碼的緣故，惡意程式的數量逐年快速增加，因此吸引許多專家學者從事這方面的研究，像是 W. Enck 等人[2]提出了一個基於核心的 Android 惡意程式行為分析檢查。該系統收集 Linux 層和應用程式的日誌並進行分析，其日誌內有系統紀錄與目標應用程式的事件。這是一種輕量級的偵測惡意程式機制，可以在安裝 Android 應用程式時比對惡意程式特徵。然而他們的研究有個重大的問題就是惡意程式有辦法避免在應用程式日誌上留下紀錄，因此僅能透過 Linux 層日記來協

助判斷。

A. P. Felt 等人[3]認為，Android 許可權限系統的目的是通知使用者安裝應用程式的風險，因此 Android 使用者是否注意且理解各種許可權限是相當重要的，但作者根據 308 位 Android 使用者進行了兩次調查，發現只有 17%的使用者注意到安裝過程中的權限，而只有 3%的使用者能正確回答出三種權限的功能。因此作者指出目前的 Android 應該提高使用者對許可權限的關注和理解，以確保安全。

M. Nauman 等人[4]指出，雖然 Android 有許可權限機制這個安全系統，但是當使用者想要安裝 Android 應用程式，就必須允許所有請求的許可權限，且沒有辦法對特定情況進行權限限制。因此作者提出的 Apex 框架允許使用者有選擇性地授予許可權限，因此可以更加嚴謹與安全的使用許可權限。

M. Zhao 等人[5]的研究使用 SVM 算法建構一個惡意程式檢測框架:AntiMalDroid，其基於特徵碼分析並可以檢測到惡意程式和它們的變種，還可以動態且有效地運行和擴展惡意程式特徵庫。A. Apvrille 和 T. Strazzere[6]提出一個靜態分析方法，採用 39 種不同的標誌，例如 Java API 調用、嵌入可執行文件的存在、代碼大小與網址等等。每個標誌都被分配一個不同的權重，基於統計計算哪些是 Android 惡意程式撰寫者最常使用在他們的程式碼中。

A. Shabtai, Y. Fledel, and Y. Elovici 三人[7]的研究運用機器學習（ML）技術，從 Android 的應用程式文件萃取資訊。例如從 Android 的 Java 字節碼(即特徵萃取，dex 文件）和其他文件類型像是 XML 文件等等。該文中重點擺在 Android 應用類型的其中兩種類型：工具和遊戲，實驗結果成功分類遊戲和工具。這種機器學習方法也可運用於檢測 Android 惡意程式。

3. 研究方法

3.1 Permission 許可權介紹

許可權(也被稱為許可權限、權限)是一種 Android 手機上的安全系統，Android 為避免手機功能遭濫用，透過設置許可權，可以讓使用者有效的知道且管理手機資源。若程式需用到某功能，開發者就要在程式中宣告許可權。目前最新版本中，Android 總共有 130 個許可權，圖 2 中顯示部分許可權。

收集程式許可權限的方法:1. 透過手機自帶的檢驗機制(圖 3)來取得程式的許可種類；2.透過萃取工具來取得程式的許可種類。透過萃取工具的這個方法，是使用一個萃取工具 apktool[8]進行提取 AndroidManifest.xml 檔案(圖 4)裡面的許可權。本篇研究主要是透過此工具取得資料集中所有應用程式的許可權類型。

Category	Android Permissions
Contacts	WRITE_SYNC_SETTINGS, WRITE_CONTACTS, READ_CONTACTS, MANAGE_ACCOUNTS, GET_ACCOUNTS, ACCOUNT_MANAGER
SMS-MMS messages	WRITE_SYNC_SETTINGS, WRITE_SMS, VIBRATE, SET_ORIENTATION, SEND_SMS, RECEIVE_SMS, RECEIVE_MMS, READ_SMS, FLASHLIGHT, BROADCAST_SMS
Call Log	VIBRATE, PROCESS_OUTGOING_CALLS, FLASHLIGHT, CALL_PHONE, CALL_PRIVILEGED
Audio & Video	WRITE_EXTERNAL_STORAGE, SET_ORIENTATION, RECORD_AUDIO, MODIFY_AUDIO_SETTINGS, GLOBAL_SEARCH, FLASHLIGHT, BLUETOOTH, BLUETOOTH_ADMIN, CAMERA
Tasks & Calendar	WRITE_CALENDAR, REORDER_TASKS, READ_CALENDAR, GLOBAL_SEARCH, GET_TASKS, BLUETOOTH_ADMIN
Browser History	WRITE_SYNC_SETTINGS, WRITE_HISTORY_BOOKMARKS, READ_HISTORY_BOOKMARKS
Images	WRITE_EXTERNAL_STORAGE, SET_WALLPAPER_HINTS, SET_WALLPAPER, SET_ORIENTATION, READ_FRAME_BUFFER, GLOBAL_SEARCH, FLASHLIGHT, BLUETOOTH, BLUE-TOOTH_ADMIN, CAMERA

圖 2: 部分 permissions 類型

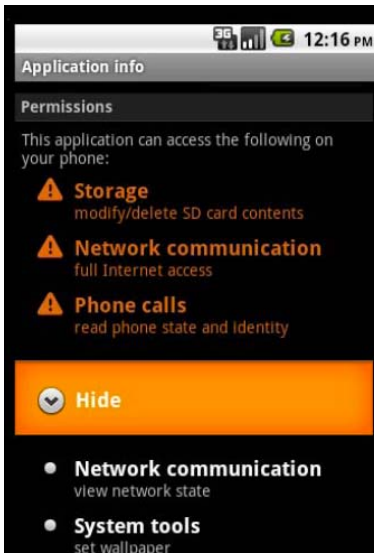

圖 3: 安裝程式時的 permissions 請求

```
<?xml version="1.0" encoding="utf-8"?>
<manifest ...
  xmlns:android="http://schemas.android.com/apk/res/android">
    <uses-sdk android:minSdkVersion="8" android:targetSdkVersion="14" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.BROADCAST_STICKY" />
    <application android:label=...>
        ...
    </application>
</manifest>
```

圖 4： AndroidManifest.xml 檔案

3.2 資料集

本研究資料集總共收集 534 個 Android 應用程

式，其中又分成訓練集與測試集。訓練集中正常程式共 201 個，惡意程式共 60 個，合計 261 個應用程式；測試集中正常程式共 213 個，惡意程式共 60 個，合計 273 個應用程式。表 1 為資料集正常程式的數量與分類，表 2 為資料集惡意程式個數。

正常程式來源為 Google Play[9]中熱門推薦的前幾頁，這些被熱門推薦且排在最前面幾頁的應用程式，絕大部分都是使用者下載數多與評分高，因此本研究將其視為正常程式；而惡意程式的來源是來自 contagio mobile dump[10]這個網站，他們收集的這些惡意程式都有附上防毒公司或防毒網站的檢測報告，因此可以確信其為惡意程式。

將各應用程式取出的許可權限進行統計，若有使用該許可權限，就設為 1；沒有便設為 0，如此就可以建立出相鄰矩陣(adjacency matrix)用來記錄每個應用程式與每個許可權限的對應關係。

3.3 分析方法

對於惡意程式而言，有些 permission 是重要且常常需要的，因此有必要找出這些惡意程式常用的 permissions。例如:INTERNET(將偷取的資料傳送給攻擊者)，或是 SMS_SEND(進行發送大量簡訊的破壞行為)。

本文中將透過機器學習分類的多種演算法來進行偵測惡意程式正確率的比較。

表 1: 資料集正常程式

	訓練集正式程式	測試集正常程式	數量
休閒	18	22	40
益智	20	26	46
工具	24	23	47
社交	24	26	50
通訊	24	24	48
多媒體	23	23	46
程式庫	23	22	45
街機與動作遊戲	20	22	42
生產應用	25	25	50
總數	201	213	414

表 2: 資料集惡意程式

	訓練集惡意程式	測試集惡意程式	總數
數量	60	60	120

機器學習是設計和分析能使電腦可以自動「學習」的演算法，並可通過經驗自動改進的演算法。也就是說，機器學習演算法從數據中自動分析獲得規律，並利用規律對未知數據進行預測。

機器學習演算法通常分為三種不同類型：監督式學習、無監督式學習、半監督式學習。監督式演算法，訓練數據集必須要有正確答案（例如：某一個應用程式是否為惡意程式）。無監督式學習演算法試圖將數據依相似性進行群聚，因此數據不需要正確答案。最後，半監督式學習演算法混合使用具有正確答案和沒有正確答案的數據，相對於無監督學習提高了準確度。因為本研究的資料集皆有標示是否為惡意程式，因此用監督式機器學習進行分析。以下介紹本研究使用的幾種演算法:

1. 貝氏網路為一有向量的非循環圖形，由節點(Nodes)與連結線(Edges)所組成，其中每個節點都有一組狀態機(Condition Probability Tables)。

2. 單純貝氏分類(Naïve Bayes)也稱為樸素貝氏分類或簡單貝氏分類，主要是根據貝氏定理(Bayesian Theorem)來預測分類的結果，是一種簡單且實用的分類方法。

3. 類神經網路使用大量簡單的人工神經元來模仿生物神經網路的能力，由很多非線性的運算單元(我們叫神經元 Neuron)和位於這些運算單元間的眾多連結所組成，這些運算單元通常是以平行且分散的方式在作運算，因此可以同時處理大量的資料，而這樣的設計就可以被用來處理各種需要大量資料運算的應用上。在 weka 工具中，MultilayerPerceptron 就是一種類神經網路，本研究使用它來進行分類。

4. 支持向量機(SVM)對於一群在空間中的資料，希望能夠在該空間之中找出一個超平面(Hyperplan)，並且希望此超平面(Hyperplan)可以將這些資料切成兩群(如:群組 A、群組 B)。屬於群組 A 的資料均位於 Hyper-plan 的同側，而群組 B 的資料均位於 Hyper-plan 的另一側，如此就將不同類別的資料完美的分開。超平面與不同類別的距離愈大愈好。在 weka 工具中 SMO 就是一種支持向量機，本研究使用它來進行分類。

5. k-最鄰近法(k Nearest Neighbor, k-NN)，KNN 是屬於案例學習(Instance -based learning)中的一種方法，KNN 將每個案例(instance)表示為 n 維空間上的一個點。假設空間中存在若干已分類好的樣本案例(sampling instances)，面對一個未知類別的案例『x』時，KNN 演算法會找出樣本案例中與『x』最接近的 k 個案例，其中多數者之類別即判斷為『x』之類別。本研究使用的 IBK 是一種 KNN 的演算法，並設定 k=1, 2, 3 進行分類。

6. 決策樹是一種語意樹(Semantic Tree)，與資料結構中的樹狀結構相仿，皆擁有根(Root)、節點(Node)以及樹葉(Leaf)等結構。而每一節點都有一個

分類的測試條件，就如「IF-THEN」的控制結構，利用測試結果來決定資料將分類於此節點的哪一棵子樹(Branch)，並繼續作為分類的條件和最後的決策。本研究使用的 J48 就是一種決策樹，並且使用 C4.5 演算法[11]。

除了許可權可以當作特徵外，本研究還將進行 android 應用程式檔案反編譯，透過反編譯可得到 smali 檔案。反編譯的過程是 Android 應用程式專案在開發完成後，將編譯打包成 APK，而其中 Java code 在編譯後，變成 classes.dex 檔案。對 classes.dex 進行反編譯後，可得到 smali 檔案。Smali 檔案的特色是可以很清楚看到函式是怎麼被呼叫使用的，包含此函式是哪個類別的或此函式所需傳入的參數等等，因此利用 smali code 來進行提取函式是非常適合的。因此函式也是一種有效的特徵，可與 permission 一起當作特徵增加惡意程式判斷正確率。本研究將利用此 smali 檔案進行函式特徵提取。

4. 實驗結果

表 3 為經由機器學習分類，得到的各演算法實驗結果(AVGTPR, AVGFPR, Accuracy)，而圖 5 是將表 3 繪製成長條圖，可以看出 AVG TPR 是 IBK(KNN, k=3)最高，其值為 0.919，J48(決策樹)最低，其值為 0.853；而 Accuracy 則是 IBK(KNN, k=3)最高，其值為 0.919；J48(決策樹)最低，其值為 0.845。一般而言大家較注重 Accuracy 而不是 AVGTPR。

本節方法相當有效，精確度基本上都有 0.84 以上，下一節實驗為加入新的特徵(將程式碼函式當作特徵)，以許可權和函示特徵一起進行機器學習分析。

表 3：機器學習分類結果

分類演算法	AVG TP Rate	AVG FP Rate	Accuracy
BayesNet	0.857	0.16	0.879
NaiveBayes	0.879	0.166	0.889
MultilayerPerceptron(類神經網路)	0.861	0.327	0.854
SMO(支持向量機)	0.872	0.299	0.867
IBK(KNN, k=1)	0.908	0.229	0.906
IBK(KNN, k=2)	0.912	0.24	0.91
IBK(KNN, k=3)	0.919	0.238	0.919
J48(決策樹)	0.853	0.365	0.845

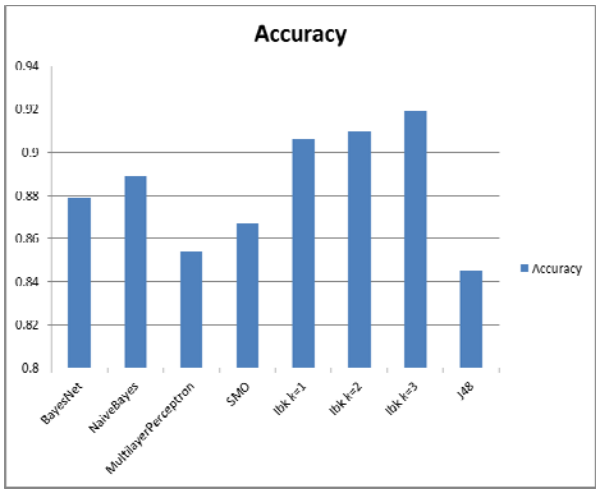

圖 5：偵測惡意程式精確度比較圖

圖 6 為許可權加函式特徵法實驗結果，此結果為使用函式與許可權當作特徵，再利用數種機器學習演算法進行分析得出。為了與圖 5 的精確度做區隔，在此我們將得到的精確度稱為 Accuracy-method。由圖可知 SMO 演算法可得到最好精確度。

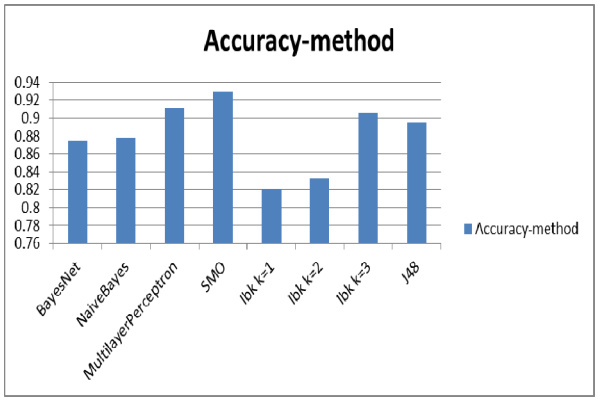

圖 6：透過機器學習分析函式與許可權當特徵的實驗結果

5. 結論與未來目標

通過本文中的方法，可以知道機器學習具有高正確率是較好的選擇，而且也確實得知許可權與函式都可以用來進行特徵分析。然而本研究還有可提升精確度的空間。在本研究中的機器學習是使用 weka 工具，其分類演算法都是使用內建的加權方法，並無法使用自己的，因此之後將自行撰寫可套用加權方法的分類演算法，便可把更好的加權方法配合機器學習來進行分類，相信可以再提升惡意程式偵測正確率。而另一種改良方向為透過應用程式類型個別進行許可權分析，分析出每個程式類型使用許可權的趨勢與差異，如此也可以提升正確率。目前由於用於分析的惡意程式檔案數不夠，因此分類後每個類中的檔案數過少，導致無法進行有效分析，期望將來能有機構或網站收集許多惡意程式，如此分類研究才能完成。

誌謝

感謝行政院國科會專題研究計畫之補助(101-2622-E-130-001-CC3, 101-2221-E-130-016, and 102-2221-E-130-006)，使本論文得以順利完成。

6. 參考文獻

[1] A.-D. Schmidt, H.-G. Schmidt, L. Batyuk, J. H. Clausen, S. A. Camtepe, S. Albayrak, and C. Yildizli, "Smartphone malware evolution revisited: Android next target?," in *2009 4th International Conference on Malicious and Unwanted Software, MALWARE 2009, October 13, 2009 - October 14, 2009*, Montreal, QC, Canada, 2009, pp. 1-7.

[2] W. Enck, M. Ongtang, and P. McDaniel, "On lightweight mobile phone application certification," in *16th ACM Conference on Computer and Communications Security, CCS'09, November 9, 2009 - November 13, 2009*, Chicago, IL, United states, 2009, pp. 235-245.

[3] A. P. Felt, E. Ha, S. Egelman, A. Haney, E. Chin, and D. Wagner, "Android permissions: User attention, comprehension, and behavior," in *8th Symposium on Usable Privacy and Security, SOUPS 2012, July 11, 2012 - July 13, 2012*, Washington, DC, United states, 2012, p. CyLab; National Science Foundation; Microsoft; Nielsen; RIM.

[4] M. Nauman, S. Khan, and X. Zhang, "Apex: Extending Android permission model and enforcement with user-defined runtime constraints," in *5th ACM Symposium on Information, Computer and Communication Security, ASIACCS 2010, April 13, 2010 - April 16, 2010*, Beijing, China, 2010, pp. 328-332.

[5] M. Zhao, F. Ge, T. Zhang, and Z. Yuan, "AntiMalDroid: An efficient SVM-based malware detection framework for android," in *2nd International Conference on Information Computing and Applications, ICICA 2011, October 28, 2011 - October 31, 2011*, Qinhuangdao, China, 2011, pp. 158-166.

[6] A. Apvrille and T. Strazzere, "Reducing the window of opportunity for Android malware Gotta catch 'em all," *Journal in Computer Virology*, vol. 8, pp. 61-71, 2012.

[7] A. Shabtai, Y. Fledel, and Y. Elovici, "Automated static code analysis for classifying android applications using machine learning," in *2010 International Conference on Computational Intelligence and Security, CIS 2010, December 11, 2010 - December 14, 2010*, Nanning, China, 2010, pp. 329-333.

[8] *apk-tool*. Available: http://code.google.com/p/android-apktool

[9] *Google Play*. Available: https://play.google.com/store

[10] *contagio mobile dump*. Available: http://contagiominidump.blogspot.com/

[11] J. Quinlan, *C4.5 programs for machine learning*: Morgan Kaufmann Publishers, 1993.