智慧型手機資安檢測技術規範探討及風險評估

張宏昌

一、緒論

1.1 研究背景

行動裝置為人們帶來許多的便利已經成為人們身邊不可缺少的設備之一，為了滿足人們不同的需求逐漸開發出各式各樣的行動應用程式產品，企業意識到廣大商機逐漸湧入行動裝置市場使得行動裝置市場迅速的發展，不管是購物網站、社群網站、即時通訊軟體、應用程式與遊戲、收發電子郵件、遠端存取資料等多元用途，漸漸改變以往人們在通訊產品使用上的習慣與方式，並且更能快速、容易從世界上任何地點與任何人取得聯繫獲得不同資訊[1]，隨著行動應用程式數量逐年提高，在程式開發過程中開發者容易缺乏安全考量可能造成使用者個資洩漏或財務上的損失，行動裝置成為身邊不可或缺的設備的同時，行動應用程式的安全考量顯得格外重要。

由於程式開發人員對於安全考量不夠周全導致不肖人士利用弱點掃描或是滲透攻擊方式來針對原生環境進行攻擊試圖竊取使用者個資和權限，注入式攻擊在安全漏洞 OWASP 前 10 大漏洞中，於 2013 年排名位居第一[2]，而第 24 屆黑客大會中曝光 QuadRooter 的安全漏洞使得有心人士利用惡意程式繞過所有保安限制，取得手機的完全控制權[3]，隨著智慧手機的盛行，使用者已經漸漸意識到個資保護的重要性，期望各個行動應用程式開發者或擁有行動應用程式的企業廠商能夠遵守使用者意願、最小權限及密不透風的三個安全原則中，從安全需求、安全設計、實作安全、測試安全及上線安全等多方面考量[4]。

1.2 研究動機

根據趨勢科技 2016 年資安預測報告中提到 2016 年底惡意及高風險的行動應用程式達到 2,000 萬個，比起電腦領域的惡意程式累積樣本數量，發現電腦領域總共花費 21 年的時間才達到 2,000 萬個惡意及高風險應用程式，而行動裝置於 2012 年底發現的 35 萬個惡意及高風險的行動應用程式，短短 4 年成長到 2,000 萬個，由此可以發現行動裝置應用程式的成長速度遠遠超過了電腦領域[5]，目前國際組織 OWASP、美國 NIST、中國工業和資訊化部、經部工業局 APP 檢測規範 V2.0、NCC、CSA…等制定行動裝置安全規範，而行動裝置使用者並不瞭解行動裝置安全規範標準重要性以及檢測規範項目是否完善。

國內檢測規範項目對於資料授權、資料儲存、資料保護、傳輸協定、傳輸保護、應用程式執行、應用程式安全、系統執行、系統安全規範不夠完善針對應用程式與系統安全部分的檢測規範存在安全疑慮。

1.3 研究目的

國內安全規範標準及檢測項目存在安全疑慮，如規範標準未嚴格要求開發者上架自行開發的應用程式時在下載前須告知使用者應用程式提取相關權限聲明同意書，而使用者同意應用程式提取相關權限的同時發現下載前權限提取聲明同意書內容與事實不符，導致使用者個資洩漏或財務上的損失。

本研究改善國內行動應用程式檢測標準讓檢測者依據檢測項目以及風險等級評估瞭解目前檢測的行動裝置是否存在安全的疑慮，讓開發人員於開放應用程式前，透過本研究改善後的行動應用程式規範中的檢測項目幫助開發者、使用者於行動裝置安全上層層把關，並參閱風險評估結果給予建議。

二、文獻探討

本章節將介紹國際組織以及各國提出相關的行動裝置規範文獻，章節包括OWASP、NIST、ITU-T、經濟部工業局、NCC、CSA、國內外相關行動裝置安全規範相關文獻整理以及相關研究探討。

2.1 文獻回顧

文獻資料分析透過蒐集、分析、歸納、研究方式將所需要的資料進行客觀而有系統的描述，蒐集相關內容要求豐富廣博，將所有收集來的資料分析之後在進行歸納以及統整。相關文獻資料為國際組織的報告、各國相關行動裝置規範的研究、國內單位組織規範、圖書館中的書籍、論文與期刊、報章新聞等等。分析步驟有四個部分，分別是即閱覽與整理、描述、分類及詮釋(林生傳，2003)。文獻回顧涵蓋許多各種不同的重點、目的、觀點的策略，注重於研究結果、研究方法、理論和應用上，同時也能夠整合其他研究所做和所說、批評過往的學術研究、建立相關領域之間橋樑以及確認在某一領域中的重要中心議題。在理論的回顧上，也包含對已經進行或建議的重要實驗的描述與不同理論的抽象概念重組及整合(Cooper，1998)。

本研究文獻回顧，除了相關領域之理論、學術論述、著作、也參考各行動裝置安全規範機構之網路訊息、年報、相關紀錄文件、作業規範以及國內相關之行動裝置安全規範等，藉此對行動應用程式安全檢測規範訂定更清楚概念並提供參考。

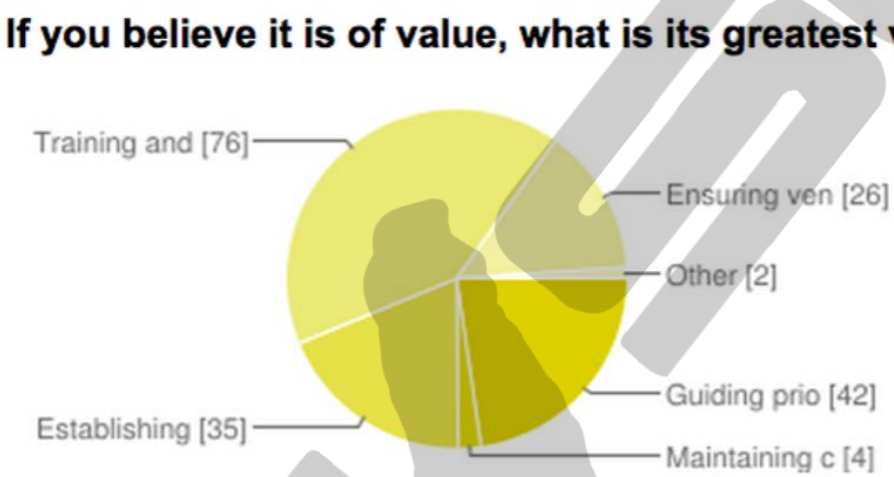
2.2 OWASP

OWASP 全名為開放 Web 軟體安全計畫(Open Web Application Security Project) OWASP 被美國確立為一個不以營利為目的的慈善組織。OWASP 是一個國際組織和 OWASP 所建立的基金會支持在世界各地 OWASP 組織上的開發，致力於幫助企業設計、開發、獲取、操作、維護...等一些可以信任的應用程序，如圖 2 所有 OWASP 使用到的工具、文檔、論壇、分會上所有相關資訊都是免費提供給所有對於應用安全上有興趣並改善應用安全性者[6]。

OWASP 在全球總共有 82 個分會員人數將近萬名，主要為研議協助解決 Web 軟體安全中的標準、工具與技術文件，長時間給予政府適當的協助以及協助企業改善網頁應用程式與網頁服務的安全性。由於應用範圍隨著人們安全意識的抬頭，網頁應用上的安全也逐漸受到重視，並成為安全領域的熱門話題之一，同時駭客也將目標鎖定在網頁應用程式開發過程中產生的漏洞進而攻擊以及破壞。

美國聯邦貿易委員會(FTC)強烈推薦企業應該遵循 OWASP 所發佈的十大 Web 弱點防護守則，其中國際信用卡資料安全技術 PCI 標準更將所列的項目當作必要基礎[7] [8]，同時培養安全意識、提升弱點補救的有效性、建立測試方法...等在行動裝置安全上的保護如圖 3[9]。

圖 1 OWASP Mobile Security Project



Guiding prioritization of vulnerability remediation	**42**	22.7%
Maintaining compliance	**4**	2.2%
Establishing testing methodologies	**35**	18.9%
Training and security awareness	**76**	41.1%
Ensuring vendors think about security	**26**	14.1%
Other	**2**	1.1%

圖 2 2015 OWASP 受益比例

2.2.1 OWASP Top 10

OWASP 開放 Web 應用安全項目每隔一段時間會公布 OWASP Top 10，以下為 Mobile Top 10 2014-Top 10 如表 1、表 2 [10]，Mobile Top 10 2016-Top 如表 3、表 4 所示 OWASP Top 10 對於網路應用安全問題上具有極大的影響力[11]。

表 1 Mobile Top 10 2014-Top 10

Mobile Top 10 2014-Top 10			
M1-Weak Server Side Controls	M2-Insecure Data Storage	M3-Insufficient Transport Layer Protection	M4-Unintended Data Leakage
M5-Poor Authorization and Authentication	M6-Broken Cryptography	M7-Client Side Injection	M8-Security Decisions Via Untrusted Inputs
M9-Improper Session Handling	M10-Lack of Binary Protections		

表 2 Mobile Top 10 2014-Top 10 項目說明

編號	威脅項目	說明
M1	伺服器端的控制薄弱	主要是說明 Mobile 的弱點並不只單存在於 Mobile 端，所開發的 APP 應用程式或雲端系統的程式亦有可能存在弱點，例如：OWASP Cloud Top 10。
M2	不安全的資料儲存	意指敏感性資料未受到適當的保護，一般常見如敏感性資料未加密，或是一些不常用到的暫存資料可能含有敏 感訊息，可能會造成機密性資料損失、憑證外洩、侵犯隱私權等衝擊。
M3	傳輸層保護不足	可攜式行動裝置於傳輸機敏性資料時，很常發生未加密情況，可能會造成駭客使用中間人攻擊(Man-in-the middle attacks)，從中竄改或竊取封包資料，進而造成機敏資料的洩漏。
M4	非故意資料外洩	可攜式行動裝置中的第三方應用程式可能會自動幫使用者儲存一些敏感性資訊，一但攻擊者成功取得行動裝置 權限時，將會侵犯使用者隱私，甚至導致資料洩漏情形。
M5	粗糙的授權與認證	部分可攜式行動裝置的網頁應用程式僅採用永不變的數值來執行身分驗證與授權階段。
M6	加密失效	所謂加密失效分為兩種情況，一種是使用強健的加密 演算法卻遭到破解，另一種為使用過於簡單的加密演算法遭到破解。前者要實現的困難度較高，後者則是相當常見。
M7	客戶端注入	Injection 攻擊即使從傳統電腦轉移到了可攜式行動裝置上的網頁應用程式，若網頁應用程式存有 Injection 弱點，攻擊者仍可利用 SQL Injection 或 XSS 攻擊手法來提升可攜式行動裝置的權限，或是利用網路盜打市話(Toll Fraud)的情況發生。
M8	不受信任的輸入	在各種可攜式行動裝置的平台均會發生，應用程式可能 經由惡意攻擊者精心設計或是應用程式遭攻擊者透過 Client Side Injection 攻擊方式來消耗可攜式行動裝置的硬體資源或提升權限情形。
M9	不適當的會話處理	可攜式行動裝置的應用程式 session 過期時間，一般而言會設定的比較長，原因是對使用者方便存取或使用。
M10	封裝檔案保護不足	此弱點的洩漏方式，乃是指應用程式原始碼中，把輸入 或輸出的相關參數直接寫入在程式碼當中，因此只要攻 擊者能夠取得應用程式的原始碼，若原始程式碼內容含有敏感資訊，可能會造成企業敏感性資料洩漏情況。

表 3 Mobile Top 10 2016-Top 10

Mobile Top 10 2016-Top 10			
M1-Improper Platform Usage	M2-Insecure Data Storage	M3-Insecure Communication	M4-Insecure Authentication
M5-Insufficient Cryptography	M6-Insecure Authorization	M7-Client Code Quality	M8-Code Tampering
M9-Reverse Engineering	M10-Extraneous Functionality		

表 4 Mobile Top 10 2016-Top 10 項目說明

編號	威脅項目	說明
M1	平台使用不當	此類別將涵蓋平台功能或不使用平台的安全控制的濫用，包括 Android 平台上的權限、濫用 Touch ID、金鑰鏈作為行動操作系統的其他安全性的控制。
M2	不安全的數據存儲	此類別將 2014 年行動裝置十大弱點 M2 + M4 的組合這包括不安全的數據存儲和數據洩漏。
M3	不安全的通信	此類別包括較弱的雙方交握、不正確的SSL版本、弱談判，明文通訊的敏感資產...等等。
M4	不安全的認證	此類別將涵蓋認證最終用戶或錯誤的會話管理的概念。包括： 1. 未能在所有識別用戶時應該要求 2. 在需要時未能保持用戶的身份 3. 會話管理薄弱環節
M5	加密不足	此類別適用於加密敏感信息資產。然而加密在某些程度上次是不夠的。請注意任何與 TLS 或 SSL 相關的內容都在 M3 裡，此外，如果應用程式無法使用加密，當它應該屬於 M2。此類別適用於嘗試加密的問題，但未能正確完成。

編號	威脅項目	說明
M6	不安全的授權	此類別適用於捕獲任何授權失敗的類別(例如，客戶端中的授權決定，強制瀏覽等類別)，它與身份驗證問題不同(例如，設備登記，用戶識別等)。如果應用程式在其應該的情況下根本不認證用戶(例如，當需要認證和授權訪問時，授予對某些資源或服務的匿名訪問)，那麼這是認證失敗而不是授權失敗。
M7	客戶端代碼質量	此類別是“透過不可信任輸入的安全決定”為較少使用的類別之一，包含所有的行動 Client 端代碼級的實施問題，這不同於服務器端編碼錯誤，捕獲緩衝區溢位、格式化字符串漏洞、各種其他代碼級錯誤，解決方案是重寫在行動設備上執行的代碼。
M8	代碼篡改	此類別包括二進制補丁、資源修改、方法掛鉤、方法調用與動態存儲器修改，一旦應用程式被傳送到行動裝置代碼和數據資源駐留在其中，攻擊者可以直接修改代碼，動態改變內存中的內容，更改或替換應用程式使用的系統 API 或修改應用程式的數據和資源，這可以為攻擊者提供一種直接的方法來破壞軟件的預期用途，以獲得個人不法收益。
M9	逆向工程	此類別包括最後核心二進制的分析，確認來源代碼、算法、其他資產。軟件如 IDA Pro、Hopper、otool、和其他二進制檢查工具給攻擊者洞察應用程式內部工作原理。可能用於利用應用程式中其他發生中的弱點，以及顯示有關後端服務器的信息、加密常數和密碼、知識產權。
M10	多餘的功能	此類別包括開發人員隱藏的後門功能或其他內部開發安全控制，不打算發佈到生產環境中。例如，開發人員不經意間在應用程序中包含密碼作為註釋。另一個例子包括在測試過程中禁用雙因素身份驗證。

2.2.2 OWASP Mobile Application Security Guide

OWASP 行動安全項目是一個集中的資源，為開發人員以及開發安全團隊建立和維護行動應用程序的安全，透過行動安全項目對行動安全風險進行分類。

行動應用安全指南主要針對應用程式開發人員和安全測試人員，訂定標準化和傳播行動應用測試方法，行動應用安全指南概述描述了一般情況下行動應用測試的方法以滿足安全測試的需要，隨著安全意識的抬頭，也慢慢為每個高水平的平台上進行定制。

2.3 NIST

NIST 全名為美國國家標準技術研究所(National Institute of Standards and Technology)，於物理、生物、工程方面的訂定穩定的基礎，為應用研究、測量技術、測試方法方面的研究提供更完善的標準、參考數據及相關的服務，在國際上享有很高的聲譽。

主要以開發和促進計量、標準和技術，以提高生產率、提高貿易、改善生活質量為主，同時幫助國家建立國家計量的基與標準、提供發展工業及國防服務的測量技術、提供研製與銷售的標準服務、計量檢定和校準服務、同時也參與標準化技術委員會制定的標準，幫助中小型企業開發新的產品並提供技術上的轉讓，在防火、抗地震技術與應用計算技術上有相當的貢獻[12]。

2.3.1 NIST SP 163

NIST SP 163 主要為幫助組織了解審視行動應用程式安全性上的過程、開發應用程式上的安全要求、提供應用程式有可能存在的漏洞類型與檢測相關漏洞的測試方法漏洞，確認相關應用程式是否適合應用於行動設備上，就算擁有最完善的審查程序也有可能會發現潛在的漏洞，而應用程式的特點或行為應具備以下安全性才能被視為所謂的安全[13]。

一般的應用程式的安全性要求包括：

1. 啟用授權功能：

   應用程式必須描述工作，所有的按鈕、選單項、其他接口須工作並且錯誤條件須正常處理。

2. 防止未經授權的功能：

   未經授權的功能，如數據洩露惡意軟體執行。

3. 限制權限：

   應用程式應該給予最小的權限，必要時只授予應用程序所需的權限。

4. 保護敏感數據：

   收集應用程式存儲和傳輸敏感數據，應保護保密性與數據的完整性以及私密性。

5. 保護應用程式代碼的依賴關係：

   應用程式必須使用任何相關性。

6. 測試應用程式更新：

   應用程式的新版本都必須經過測試，以確定任何新的弱點。

2.3.2 NIST SP 164

NIST SP 164 規範中，將行動裝置基礎架構分為硬體、韌體、作業系統、應用程式及資訊內容，特別針對手機硬體的密碼模組功能區塊的檢測措施與管理辦法，參考 NIST 聯邦資訊處理標準 FIPS-140-2 標準來加強手機密碼模組安全功能。而密碼模組是指由硬體、軟體及韌體組合成的模組，用來實現密碼邏輯、程序及密碼演算法[14]。

行動設備應該擁有以下三種安全功能：

1. 設備誠信：

   行動設備可以提供已經維護設備完整性的證據，用於傳達該可信狀態的機制通過一個或設備所有者允許設備向所有者發出信息。

2. 隔離：

   隔離防止對同一信息擁有者之間的互動意外設備。

3. 保護存儲：

   受保護的存儲保存敏感數據的機密性和完整性在休眠時，在使用中（未授權的應用程式的情況下）訪問受保護存儲的項目與撤銷訪問。

2.4 ITU-T

ITU-T 全名為國際電聯電信標準化部門(ITU Telecommunication Standardization Sector)，各個研究組匯集了來自世界各地的專家，而 ITU-T 主要為制定國際標準的建議書，這些國際標準是全球信息通信技術（ICT）基礎設施的訂定，標準對 ICT 的互連互通非常重要，無論是語音、視頻通信還是數據消息交換，可確保各國的 ICT 網絡和設備使用相同的語言，而達到全球通信的目標。

國際 ICT 標準為了避免市場爭鬥，對於新加入市場的公司而言，標準的訂定可以為新市場提供完整公平的競爭環境，而對於發展中的國家建設基礎設施、鼓勵經濟發展有實質性的幫助，在規模經濟產生後，各方可因標準帶來的費用降低受益如製造商、運營商、消費者。

ITU-T 積極參與文稿引導、基於共識的標准開發方法，所有國家和公司，無論大小，在影響 ITU-T 建議書的開發方面都享有平等權利，ITU-T 從最初制定國際電報交換標準的機構，發展至今結合 ICT 環境，為全球標準化提供世界上的最佳設施，作為世界上僅有的真正全球性 ICT 標準機構[15]。

2.5 經濟部工業局

經濟部工業局以產業界需求為導向，提供相關產業完整服務幫助，為台灣工業創新升級、轉型，積極輔導廠商來強化經營同時提高產業間的生產力與國際之間的競爭力，協助企業因應產業環境的變化，主要為幫助工業政策發展、策略與措施擬訂同時推廣工業升級相關計畫、協助工業區開發管理與工業發展有關財稅金融措施擬訂、工業污染防治相關安全輔導及工廠管理[16]。

2.6 NCC

NCC 全名為國家通訊傳播委員會(National Communications Commission)，為了確保相關通訊傳播產業市場的有效競爭擬訂相關規範，資通訊產業在國家產業發展中扮演著舉足輕重位置，為了因應全球性數位匯流發展革新趨勢與整合現在所有相關的通訊及傳播分散的狀況，政府依資訊與電信策略會議（SRB 會議）建議，規劃成立電信資訊傳播整合監理機關。

NCC 主要為通訊傳播監理政策之訂定、法令之訂定、擬訂、修正、廢止及執行、協助通訊傳播事業營運監督管理以及證照核發、系統及設備的審驗、相關工程技術規範訂定、內容分級制度與相關法律規定相關事項規範、競爭秩序維護、資源管理、相關資通安全技術規範與管制、國際交流合作處理、調查及裁決[17]。

2.7 CSA

CSA 全名為雲端安全聯盟(Cloud Security Alliance)為全球非營利組織，CSA Taiwan Chapter 自 2011 年底於台灣成立為設立民間協會組織，在國內屬於雲端服務安全等新興資安議題的推動者之一，為了幫助提高國內雲端服務對於使用者信任關係，搭配全球雲端服務安全證書 CSA STAR 的推動，並透過第三方的驗證方式來建立公正客觀的量測標準，提高使用者對於雲端服務平台的信賴，雲端安全聯盟所發表的雲端安全指南和雲端運算領域已經成為目前熱門的重要文件之一如圖 4 所示，CSA 主要為提供使用用戶和供應商相關對於雲端運算的安全需求、推廣雲端運算安全最佳做法以及正確使用雲端運算與相關解決方案、建立有關雲端安全保證問題方針[18]。

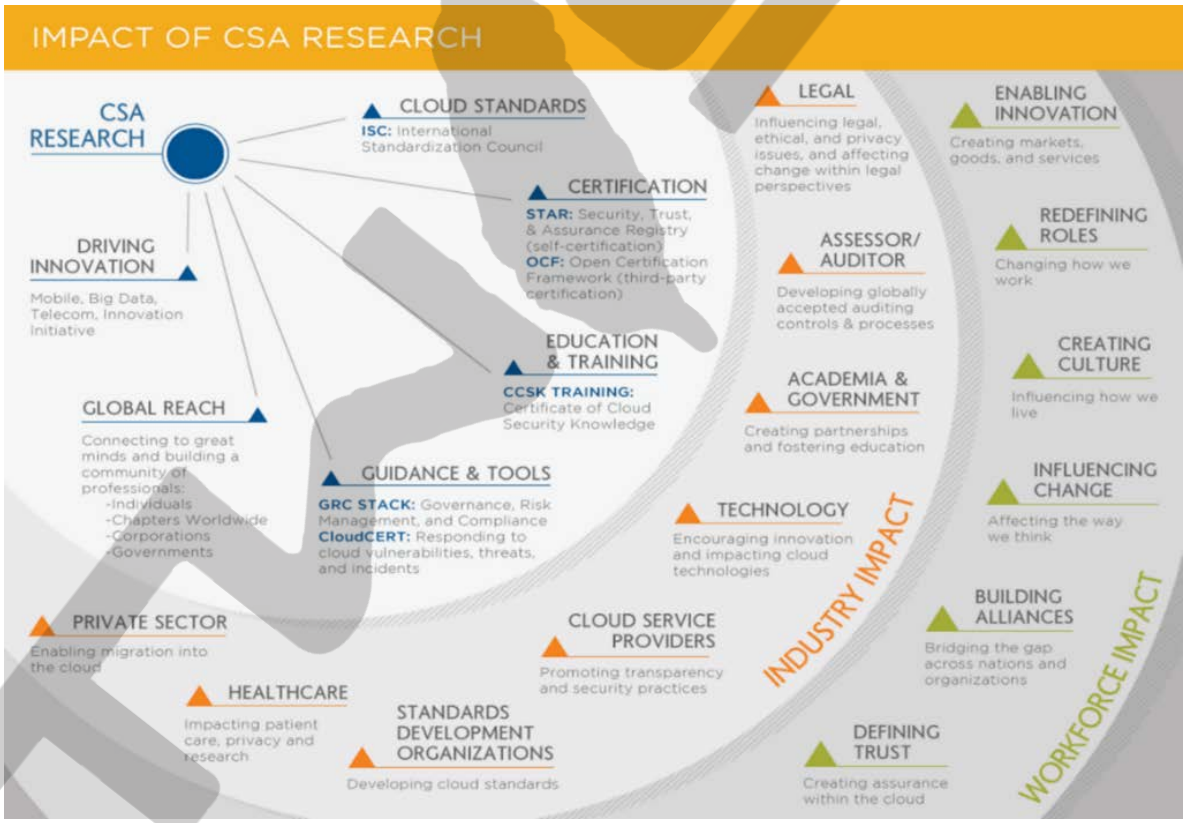


圖 3 CSA 研究影響圖

2.8 相關文獻整理

相關行動裝置安全規範文獻為國際組織、美國、歐盟、中國、台灣、英國、日本、新加坡、韓國行動裝置檢測規範相關發布文獻如表 5 所示。

表 5 國內、國外行動裝置安全規範相關文獻整理表

編號	地區	發布單位	相關發布文件
1	國際組織	A. Open Web Application Security Project(OWASP)	1.OWASP Mobile Security Project - Top Ten Mobile Risks(2014)
			2.OWASP Mobile Security Project-Mobile Test cases
		B. International Organization for Standardization(ISO)	1.ISO / IEC 27001:2013[19]
			2.ISO / IEC 27002:2013[20]
2	美國	A. U.S. Chief Information Officer and the Federal CIO Council	Government Mobile and Wireless Security Baseline(2013)[21]
		B. National Institute of Standards and Technology(NIST)	1.SP800-53 Rev 4, Security and Privacy Controls for Federal Information Systems and Organizations(2013) [22] 2.SP-800-124 Rev 1, Guidelines for Managing the Security of Mobile Devices in the Enterprise(2013) [23] 3.SP 800-142, Practical Combinatorial Testing(2010) [24] 4.SP 800-164, DRAFT Guidelines on Hardware-Rooted Security in Mobile Devices(2012) [25] 5.SP 800-163,Vetting the Security of Mobile Applications (2015) [26]
		C. Department of Homeland Security(DHS)	Cyber Threats to Mobile Phones(2011) [27]
		D. Department of Defense(DoD)	1.Commercial Mobile Device Implementation Plan(2013) [28] 2.Mobile Device Strategy(2012) [29] 3.Mobile Operating System Security Requirements Guide Version 1, Release 3 (2013) [30]
		E. Federal Communications	1.Ten Steps to Smartphone Security(2012) [31]

編號	地區	發布單位	相關發布文件
		Commission (FCC)	2.Ten Steps to Smartphone Security (Android)(2012) [32] 3.Ten Steps to Smartphone Security (iOS) (2012) [33] 4.Ten Steps to Smartphone Security (Windows Phone)(2012) [34] 5.Ten Steps to Smartphone Security (BlackBerry)(2012) [35]
		F. National Security Agency(NSA)	1.Mobility Security Guide v2.3(2013) [36] 2.Security Requirements for Mobile Operating Systems v1.0 (2013) [37]
3	歐盟	A. European Union Agency for Network and Information Security (ENISA)	1.Smartphone secure development guidelines for app developers(2011) [38] 2.Top Ten Smartphone Risks(2014) [39] 3.Smartphones：Information security risks, opportunities and recommendations for users(2010) [40]
		B. European Commission	Opinion 02/2013 on apps on smart devices(2013) [41]
4	中國	A. 工業和資訊化部	1.YD/T 2407-2013 移動智慧終端機安全能力技術要求(2013) [42] 2.YD/T 2408-2013 移動智慧終端機安全能力測試方法(2013) [43]
5	台灣	A. 中華民國國家標準	1.CNS 27000 資訊安全管理系統－概觀及詞彙[44]
			2.CNS 27001 資訊安全管理系統－要求事項[45]
			3.CNS 27002 資訊安全管理之作業規範[46]
			4.CNS 27003 資訊安全管理系統實作指引[47]
			5.CNS 27004 資訊安全管理－量測[48]
			6.CNS 27005 資訊安全風險管理[49]
			7.CNS 27006 資訊安全管理系統稽核與驗證機構要求[50]
		B. 中華民國資訊軟體協會	行動應用 App 安全開發指引草案[51]
		C. 經濟部工業局	行動應用 App 基本資安檢測基準 2.0[52]
		D. 雲端安全聯盟	行動 App 安全項目[53]

編號	地區	發布單位	相關發布文件
		E. 國家通訊傳播委員會	104 年手機系統內建軟體資安檢測[54]
6	英國	A. Communications-Electronics Security Group(CESG)	End User Device Strategy：Security Framework & Controls(2013/2) [55]
		B. Information Commissioner's Office (ICO)	Personal information online small business checklist[56]
		C. Office of Communication(Ofcom)	Safer smartphones–keeping your device secure(2013) [57]
7	日本	A. Japan Smartphone Security Forum (JSSEC)	Security Guideline for using Smartphones and Tablets(2011) [58]
		B. Information-technology Promotion Agency (IPA)	10 Major Security Threats (2013) [59]
8	新加坡	A. Monetary Authority of Singapore	Technology Risk Management Guidelines(2013) [60]
9	韓國	A. Korea Communications Commission	KCC announces 'Comprehensive Plans for Smart Mobile Security'(2010) [61]

2.9 相關研究探討

William Enck,Ongtang, M. and Mcdaniel, P. (2009)提到當應用程式需要擁有相關應用程式權限宣告列表，同時要求用簡短的訊息描述相關的權限並設置給予使用者同意和取消按鈕。使用者可以選擇同意將宣告列表上所描述的權限是否同意，單單只是同意或拒絕這樣的權限保護的保障機制是不夠的，對於使用者同意將上續的相關權限授予時對於之後的安全上存在疑慮[62]。

許博學(2013)提到為授予權限狀況下可以間接存取系統元件，惡意應用程式利用以間接存取系統元件的方式，來執行本身未授予的權限下執行的行為，這是因為 Android 系統未全面對於相關權限管制不夠嚴謹管制因素之一。

使用者應強制檢查授權者權限，避免系統元件被未受與權限者執行，應用程式雖然沒有相關權限，但它可以利用意圖物件呼叫系統瀏覽器元件行為，以間接方式間接地達成通訊目的[63]。

2.10 風險評估

依據國內手機系統內建軟體資安檢測中的風險評估方法，針對行動裝置應用程式隱私安全、原生安全、防護安全檢測項目提出風險評估包含評估方式、等級與結果。

2.10.1 資訊安全與風險定義

資訊安全主要三項要素稱為資訊安全三原則 CIA，有以下三項特性[64]：

- 機密性(Confidentiality)：資料傳遞與儲存的資料，應避免未經授權使用者存取。
- 完整性(Integrity)：資料傳輸與儲存過程中，應避免未授權使用者或處理程式竄改。
- 可用性(Availability)：確保行動裝置的相關資源保持於可用狀況。

參考國際資訊安全管理標準 ISO/IEC 27000:2009 系列及 ISO 31000:2009 系列，在於提供風險管理的原則及指導綱要，透過其架構化、系統化的方法，將無形難以具體描述的各類型不確定性，以更透明的方式加以管理。

2.10.2 風險評估方法

ISO 31010:2009 風險管理—風險評鑑技術(Risk management -- Risk assessment techniques)，風險評估步驟為識別風險及其發生的原因、確定風險發生後的後果、再次確定風險發生的概率、識別降低風險後果或可能性的因素[65]。如表 6 所列風險評鑑方法，依據風險評鑑之過程，區分為風險識別、風險分析(含衝擊後果、發生可能性、風險等級)及風險評估等階段，標示所列風險評鑑方法之適用性，SA：極為適用　NA：不適用　A：適用。

表 6 風險評鑑方法

方法名稱	風險識別	後果分析	機率分析	風險等級	風險評估	備註
腦力激盪法	SA	NA	NA	NA	NA	
結構或非結構化訪談法	SA	NA	NA	NA	NA	
Delphi 德爾非法	SA	NA	NA	NA	NA	
檢查表法	SA	NA	NA	NA	NA	
主要危險分析法(PHA)	SA	NA	NA	NA	NA	
危害與可操作性分析法（HAZOP）	SA	SA	A	A	A	
危害分析重要管制點法（HACCP）	SA	SA	NA	NA	SA	
環境風險評鑑法	SA	SA	SA	SA	SA	
結構化 SWIFT 法	SA	SA	SA	SA	SA	
情境分析法	SA	SA	A	A	A	
營運衝擊分析法	A	SA	A	A	A	●
根本原因分析法(RCA)	NA	SA	SA	SA	SA	
失效模式與影響分析法(FMEA)	SA	SA	SA	SA	SA	
故障樹分析法	A	NA	SA	A	A	
事件樹分析法	A	SA	A	A	NA	
原因及後果分析	A	SA	SA	A	A	
原因及影響分析法	SA	SA	NA	NA	NA	
保護層及分析法(LOPA)	A	SA	A	A	NA	
決策樹分析法	NA	SA	SA	A	A	
人員可靠性分析法	SA	SA	SA	SA	A	

方法名稱	風險識別	後果分析	機率分析	風險等級	風險評估	備註
Bow tie analysis	NA	A	SA	SA	A	
可靠性為中心的維修 (RCM)	SA	SA	SA	SA	SA	
潛行迴路分析法 (Sneak circuit analysis)	A	NA	NA	NA	NA	
馬爾可夫轉移矩陣法 (Markov analysis)	A	SA	NA	NA	NA	
蒙特卡羅方法 (Monte Carlo simulation)	NA	NA	NA	NA	SA	
貝氏網路法 (Bayesian statistics and Bayes Nets)	NA	NA	NA	NA	SA	
ROC 曲線(FN curves)	A	SA	NA	NA	SA	
風險指標法(Risk indices)	A	SA	SA	A	SA	
後果可能性矩陣 (Consequence/probability matrix)	SA	SA	SA	SA	A	
成本利潤分析法 (Cost/benefit analysis)	A	SA	A	A	A	●
多目標決策分析 (Multicriteria decision analysis)	A	SA	A	SA	A	

註記	SA：非常適用 Strongly applicable NA：不適用 Not applicable A：適用 Applicable CNS/ISO/IEC 27005:2011 建議之資訊安全風險評鑑風法 高階風險評鑑法=營運衝擊分析法 詳細風險評鑑法=後果可能性矩陣

參考來源：ISO 31010 附錄 A

依據 ISO/IEC 27005 附錄 E 資訊安全風險評估作法分為高階風險評鑑作法(High-Level Risk Assessment)、詳細風險評鑑作法(Detailed Risk Assessment Approach)。

(一)高階風險評鑑作法

組織內部預算、人力、時間、資源等相關原因與限制，應先對於內部資訊系統與資產做初步的高階風險評鑑從中找出各個資訊系統於組織營運之價值以及高階衝擊的影響為開始，如高階風險評鑑後屬於較低風險/較低衝擊影響的資訊系統，可自行對於其選擇適用的風險鑑定方式或風險處理[66]，使用鑑別機制高階後果衝擊評鑑中理論基礎與方法論，對於資訊系統進行分類分級，分別為低、中、高三個級別，而資訊系統的高階風險評鑑等級分別為低風險、中風險、高風險，並針對不一樣的結果進行相對應的處理[67]。

高階風險評鑑作法之特點如下：

- 可簡單快速區隔，低風險/衝擊、中風險/衝擊、高風險/衝擊有效建構適合的資安風險計畫，幫助整體組織資安規劃以及後續維護。
- 可有效的運用組織資源與預算。

(二)詳細風險評鑑作法

詳細風險評鑑作法主要分為風險分析(Risk Analysis)與風險評估(Risk Evaluation)針對組織內部透過系統化方式搜尋所有資訊系統與資訊資產中優先處理的資料資產所對應的風險，當中包含深入的識別資產與價值、對資產威脅與脆弱性的評鑑，並根據 CIA 衝擊及其可能性，採取適當的防護措施[68]。

詳細風險評鑑作法之特點如下：

- 資訊系統化方式識別出該資訊資產所對應的風險，並採取適當的防護措施。
- 詳細風險評鑑作法之結果可當作往後安全變更異動管理的參考。

三、研究方法

由前述得可知，行動裝置的應用程式數量與日俱增，應用程式開發增加的同時惡意開發者間接利用應用程式竊取未經同意盜用隱私資料的可能性大幅提升，目前國內的行動裝置檢測項目並不完善未依據 OWASP、NIST、ITU-T、經濟部工業局、NCC、CSA 作為檢測項目的基礎，無法有效提供使用者完整的檢測項目避免使用者個資洩漏或財務上的損失。

本章節說明本研究透過文獻回顧法探討行動應用程式安全檢測項目規範，以 OWASP 十大弱點、OWASP 行動應用安全指南、NIST SP800-163.164、ITU-T YD/T 2407、CSA 行動應用程式安全測試為基準，依照各檢測項目中關鍵字部分採用歸納法方式分類，探討目前國內相關研究符合安全議題檢測項目，依據可能造成 CIA 衝擊程度等級分別為風險等級分為低、中、高三級，藉由改善後的行動應用程式安全檢測規範幫助開發者、使用者於行動裝置安全上把關。

3.1 行動應用程式規範

本研究方法透過國際規範 OWASP 十大弱點、OWASP 行動應用安全指南、NIST SP800-163.164、ITU-T YD/T 2407、CSA 行動應用程式安全測試來當作本研究行動應用程式檢測項目規範的基準，並針對規範中所提到的內容分別依照資料授權儲存安全、傳輸協定安全、應用程式執行安全、系統執行安全來找出文件規範目標方向，藉以瞭解各個國際組織規範中的目標方向。

由附錄一中可以看出 OWASP 十大弱點整體不管是資料授權儲存安全、傳輸協定安全、應用程式執行安全、系統執行安全規範方向都有考量，NIST SP800-163 主要針對資料授權儲存安全、傳輸協定安全方向來進行規範相對於應用程式執行安全、系統執行安全部分 NIST SP800-163 有明顯不足，NIST SP800-164 主要是針對資料授權儲存安全、傳輸協定安全方向來進行規範不同於 NIST SP800-163 的地方在於 NIST SP800-164 有對系統執行安全來做一些規範，相對於應用程式執行安全部分有明顯不足，ITU-T YD/T 主要是針對資料授權儲存安全、傳輸協定安全、系統執行安全來進行規範，相對於應用程式執行安全部分有明顯不足[54]，藉此以 NIST SP800-163.164、ITU-T YD/T 2407 作為資料授權儲存安全、傳輸協定安全規範基準，藉由 OWASP 十大弱點、OWASP 行動應用安全指南、CSA 行動應用程序安全測試規範來彌補應用程式執行與應用安全部分的不足之處，並依照檢測規範類別列入相關的各個檢測項目如表 7 所示，分別為隱私安全、原生安全、防護安全類別，而隱私安全類別分別包含權限提取、隱私資料安全檢測項目，原生安全類別分別包含 API/函式庫原生安全、應用程式資料安全、原生環境混淆，防護安全類別分別包含傳輸協定與加密強度、資料儲存安全。

表 7 類別及檢測項目說明

類別	檢測項目	說明
隱私安全	權限提取不當	行動應用程式應完整宣告行動裝置資源與權限用途說明，並具備使用者授權之相關授權安全機制。
	隱私資料安全	由於開發流程中並未有嚴謹全面的安全性考量，導致在使用應用程式相關的隱私資料暴露在危險之中。
原生安全	API/函式庫原生安全	應用程式開發時，時常使用到相關的 API 或者相關的SDK 等套件來達到開發者所需的功能，而這些引用的相關套件並未取的安全性的驗證，導致存在漏洞造成原生安全威脅。
	應用程式資料安全	應用程式安裝後執行會產生相關所需的資料，而這些資料透過特地狀況或者開發時所設計漏洞造成資料安全問題。
	原生環境混淆	利用產生特定的方式來破壞原生環境來達到未預期的機制，造成安全上的風險。
防護安全	傳輸協定與加密強度	應用程式在連線時會進行伺服器相關資料存取服務，在傳輸的過程中，針對傳輸協定或通道做完整加密防止傳輸資料遭到竊取。
	資料儲存安全	應用程式在執行的過程中所產生資料，分為暫時性資料與永久性資料會儲存在執行的應用程式設備上，針對應用程式開發時是否有對於資料儲存做完整安全考量。

3.2 研究限制

本實證Web軟體安全計畫簡稱OWASP、美國國家標準技術研究所簡稱NIST、雲端安全聯盟簡稱CSA具有高證據等級文獻，國內外包含許許多多個案研究篩選文獻後發現未具良好證據等級，因此未採納。文獻回顧著重於國內相關之行動裝置安全議題相關研究，往後研究可留意納入國外相關文章或研究性文章。目前只納入國內中、英文文章來分析故文章收集廣度較適合國內行動裝置應用程式檢測規範採用，未來可納入其他國家與研發技術之文獻進行分析，可使檢測規範更具效度。

3.3 文獻搜尋

初步建立中文關鍵字”安卓”、”應用程式”、”檢測”、”分析”、”機制”、”隱私”，英文關鍵字”Android” or ”App” or ”detection” or ” analysis” or ” Mechanism” or ”Privacy ”執行文獻搜尋，中文資料庫包含台灣碩博士論文系統，以上續關鍵字中英文名詞搜尋文獻。

3.4 篩選標準條件

研究者以瀏覽方式針對中英文文獻標題.摘要篩選納入標準，在以排除條件刪除相關文獻，採用人工方式閱讀文獻標題、摘要，若無摘要則提取文章內容，透過關鍵字篩選內容須符合 OWASP 安全議題、全球雲端安全重要聯盟 CSA 安全議題，進行安全檢測的目標項目。

3.5 檢測項目歸納

如表 8 所示，以 OWASP 安全議題、CSA 安全議題為基礎，依照檢測項目中關鍵字部分採用歸納法方式歸類，探討目前國內相關研究符合安全議題檢測項目。

表 8 檢測項目歸納表

類別	檢測項目	關鍵字
隱	權限提取不當	權限欺騙、權限表、權限、越權、資源、授權。

私安全	隱私資料安全	隱私安全性、洩漏資料、資料外洩、隱私資料、私密資料、隱私機制、機敏資訊、敏感資料、個人隱私、機密性、存取、竊取、洩露、個資。
原生安全	API/函式庫原生安全	逆向工程、反向工程、呼叫序列、反組譯、apk、編譯、API、SQL 指令、System Call、API call。
	應用程式資料安全	共謀型攻擊、攔截程式碼、系統漏洞、竊取、傳送、外流、漏洞、弱點、隱藏。
	原生環境混淆	指令碼、包裝、封裝、混淆。
防護安全	傳輸協定與加密強度	不可偽造性、網際網路、封包擷取、雙向鑑別、中間人、重送、加密強度、加密協定、加密程序、密碼參數、暴力破解、安全強度、加密技術、密碼學、傳送、認證、協定、金鑰。
	資料儲存安全	資料保護、身分鑑別、保護措施、簡訊認證、儲存。

3.6 風險評估框架

如表 9、表 10 所示，根據國內手機系統內建軟體資安檢測中的風險評估方法與風險等級，採用高階風險評鑑作法，以 Mobile Top 10 2016-Top 10、Cloud Security Alliance 作為風險評鑑威脅分析與評估行動裝置於隱私安全、原生安全、防護安全，為風險評估框架評估可能造成 CIA 衝擊程度等級，將風險等級分為低、中、高三級。

表 9 風險評估框架

Mobile Top 10 2016-Top 10										類別區分			風險等級		
M1 平台使用不當	M2 不安全的數據存儲	M3 不安全的通信	M4 不安全的認證	M5 加密不足	M6 不安全的授權	M7 客戶端代碼質量	M8 代碼篡改	M9 逆向工程	M10 多餘的功能	隱私安全	原生安全	防護安全	低	中	高

表 10 風險等級

風險等級	說明
低	行動裝置應提供使用者基本相關行動裝置安全與個人資料保護，如完整宣告行動裝置資源與權限用途，並給予相關保護。
中	行動裝置應提供使用者資料及隱私方面完整保護機制，如資料傳輸過程與資料的儲存的安全機制，確保使用者在使用、傳輸、儲存整體過程中的安全保護。
高	行動裝置應確保核心層不被惡意竄改及破壞，避免相關資料洩露。

四、結果與討論

4.1 檢測項目細項說明

本研究依據各國提出的行動應用程式規範項目來改善國內行動應用程式規範不足之處，檢測項目細項說明如表 11 所示，權限提取不當檢測項目中於行動應用程式發布時完整說明用途並取得使用者同意、拒絕機制、未經使用者授權准許使用非相關授權的函數，規範出 6 項檢測細項。隱私資料安全檢測項目中應用程式程式碼封裝之檔案內容不應存放敏感訊息且應使用密碼強度策略防止資料遭竊，規範出 6 項檢測細項。API/函式庫原生檢測項目中應用程式應避免應用程式開發時產生逆向工程與注入式安全漏洞造成原生安全威脅，規範出 7 項檢測細項。應用程式資料安全檢測項目中應用程式安裝後執行會產生相關所需的資料不應存在於記憶體傾印中且應於設定時間內不活動時是否具自動關閉應用程式或鎖定避免資料透過特地狀況或者開發時所設計漏洞造成資料安全問題，規範出 5 項檢測細項。原生環境混淆檢測項目中應用程式意外所產生未預期的狀況或再包裝與混淆技術造成的安全風險，規範出 4 項檢測細項。傳輸協定與加密強度檢測項目中應用程式於傳輸過程應對傳輸協定或通道做完整加密採用安全之加密演算法及相關驗證機制防止傳輸資料遭竊，規範出 14 項檢測細項。資料儲存安全檢測項目中應用程式應於儲存敏感性資料時應提供資料加密且將帳號密碼儲存於作業系統保護區內或以加密方式儲存避免相關敏感性資料以明文方式存在於執行檔中避免遭不正當方式取得敏感性資料，規範出 4 項檢測細項。

表 11 本研究行動應用程式規範檢測項目細項說明

檢測項目	安全需求	
A.1 權限提取不當	行動應用程式應完整宣告行動裝置資源與權限用途說明，且取得使用者授權之相關授權機制。	
檢測項目	檢測編號	檢測細項
A.1 權限提取不當	A.1.1	行動應用程式在發布時應完整說明存取敏感性資料、行動裝置資源及宣告權限用途。
	A.1.2	行動應用程式存取與個人資料相關敏感性資料應檢查行動應用程式是否提供相關身分授權機制。
	A.1.3	行動應用程式應於存取敏感性資料前，取得使用者同意。
	A.1.4	行動應用程式未經使用者授權准許使用非相關授權的函數。
	A.1.5	行動應用程式經使用者設定拒絕存取敏感性資料後，應用程式不得使用其他方式存取相關敏感資料。
	A.1.6	行動應用程式經使用者使用聯絡人或發送接收刪除訊息等功能伺服器必需紀錄，測試者應該嘗試存取另一個用戶的功能，以便驗證是否可以存取不應該被用戶的角色/特權所允許（但可能被允許作為另一個用戶）的功能。

檢測項目	安全需求	
A.2 隱私資料安全	由於開發流程中並未有嚴謹全面的安全性考量，導致在使用應用程式相關的隱私資料暴露在危險之中。	
檢測項目	檢測編號	檢測細項
A.2 隱私資料安全	A.2.1	行動應用程式開發過程中修改密碼執行不當動作所造成的漏洞。
	A.2.2	行動應用程式開發過程中是否有註銷功能藉此減少會話劫持攻擊的可能性。
	A.2.3	行動應用程式中程式碼或其他封裝之檔案內容，是否存放敏感訊息。
	A.2.4	行動應用程式是否使用密碼強度策略防止隱私資料遭竊取。
	A.2.5	行動應用程式有可能造成安全性漏洞功能是否設定為開/關。
	A.2.6	行動應用程式是否可在 Rooted / Dev Unlocked / Jailbroken….等環境下存取。

檢測項目	安全需求	
B.1 API/函式庫原生安全	應用程式開發時，時常使用到相關的 API 或者相關的 SDK 等套件來達到開發者想要的功能，而這些引用的相關套件並未取的安全性的驗證，導致存在漏洞造成原生安全威脅。	
檢測項目	檢測編號	檢測細項
B.1 API/函式庫原生安全	B.1.1	行動應用程式是否存在 SQL 攻擊的風險。
	B.1.2	行動應用程式是否具延伸標記語言攻擊字串的風險。
	B.1.3	行動應用程式是否具逆向工程攻擊的風險。
	B.1.4	行動應用程式代碼是否以常量的方式將敏感資訊書寫在原始碼。
	B.1.5	行動應用程式是否具 Session Fixation 攻擊的風險。
	B.1.6	除錯設定設定值是否安全。
	B.1.7	是否具 FTP 協定注入式攻擊。

檢測項目	安全需求	
B.2 應用程式資料安全	應用程式安裝後執行會產生相關所需的資料，而這些資料透過特地狀況或者開發時所設計漏洞造成資料安全問題。	
檢測項目	檢測編號	檢測細項
B.2 應用程式資料安全	B.2.1	行動應用程式在設定時間內不活動時是否具自動關閉應用程式或鎖定功能。
	B.2.2	行動應用程式是否具LDAP注入攻擊的風險。
	B.2.3	行動應用程式是否具OS命令注入攻擊的風險。
	B.2.4	行動應用程式所產生的敏感訊息是否存在於記憶體傾印中。
	B.2.5	行動應用程式透過惡意方式所產生漏洞導致資料安全暴露在危險的風險。

檢測項目	安全需求	
B.3 原生環境混淆	利用應用程式使用時檢核或產生意外所產生一些維持及測試的程式或呼叫，導致未預期的狀況，造成相當的安全風險。	
檢測項目	檢測編號	檢測細項
B.3 原生環境混淆	B.3.1	行動應用程式是否具 XSS 攻擊的風險。
	B.3.2	行動應用程式是否具惡意檔案上傳的風險。
	B.3.3	行動應用程式是否存在 IOS 快照漏洞。
	B.3.4	行動應用程式透過惡意再包裝與混淆技術造成安全上的風險。

檢測項目	安全需求	
C.1 傳輸協定與加密強度	應用程式在連線時會進行伺服器相關資料存取服務，在傳輸的過程中，針對傳輸協定或通道做完整加密防止傳輸資料遭到竊取。	
檢測項目	檢測編號	檢測細項
C.1 傳輸協定與加密強度	C.1.1	行動應用程式透過網路傳輸敏感性資料時，是否使用加密傳輸以確保敏感性資料安全。
	C.1.2	行動應用程式是否具交談識別碼遭重送攻擊的風險。
	C.1.3	行動應用程式與付費功能伺服器間之資料加密傳輸，是否使用安全之加密演算法。
	C.1.4	行動應用程式與伺服器間之資料加密傳輸，是否使用安全之加密演算法。
	C.1.5	行動應用程式是否具離線狀態繞過認證的風險。
	C.1.6	行動應用程式目前狀態是否驗證 MSISDN 號碼。
	C.1.7	行動應用程式是否具被繞過二級身份驗證的風險。
	C.1.8	行動應用程式內是否清除 SSL Tunnel 中的文件資訊。
	C.1.9	行動應用程式是否具被繞過客戶端驗證風險。
	C.1.10	行動應用程式中 SSL 憑證/ SSL / TLS 密碼/協議/密鑰無效風險。
	C.1.11	行動應用程式中是否具敏感訊息以明文形式暴露的風險。
	C.1.12	CAPTCHA 沒有使用至行動應用程式的公共頁面/登錄頁面上。
	C.1.13	是否具敏感訊息作為查詢字串參數繞過安全機制風險。
	C.1.14	檢查是否具網址參數修改攻擊風險。

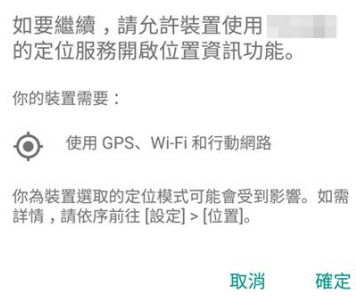
檢測項目	安全需求	
C.2 資料儲存安全	應用程式在執行的過程中都會產生資料，這些資料分為暫時性資料與永久性資料會儲存在執行的應用程式設備上，針對應用程式開發時是否有對於資料儲存做安全考量。	
檢測項目	檢測編號	檢測細項
C.2 資料儲存安全	C.2.1	行動應用程式應將帳號密碼儲存於作業系統保護區內或以加密方式儲存。
	C.2.2	行動應用程式於儲存敏感性資料時應提供資料加密功能，以避免遭不正當方式取得敏感性資料。
	C.2.3	行動應用程式與遠端伺服器溝通之帳號密碼不應以明文方式存在於執行檔中，以避免遭不正當的方式存取。
	C.2.4	應用程式使用加密強度不足。

4.2 檢測步驟與改善方式

本研究提出改善後的行動應用程式規範，依照檢測項目中檢測步驟與各國行動應用程式規範提出的改善方式進行改善建議如表 12 所示，將權限提取不當、隱私資料安全、API/函式庫原生、應用程式資料安全、原生環境混淆、傳輸協定與加密強度、資料儲存安全，共 46 項檢測規範。

依檢測細項列出檢測步驟、範例截圖與改善方式，使用者、開發者參照各檢測項目檢測步驟進行實際檢測，同時依圖片式的範例方式讓使用者、開發者能夠瞭解檢測項目的目的，並給予改善方式藉以降低行動裝置安全所造成的風險。

表 12 本研究行動應用程式規範檢測步驟與改善方式

檢測編號：A.1.1
測試步驟： 行動裝置應用程式需在發布時完整說明應用程式存取相關敏感性資料、資源、宣告權限用途，依照 API 使用到的 android 權限列表內容完整呈現。 一般權限與危險權限這兩種權限同樣都要在 AndroidManifest.xml 中使用<uses-permission>宣告，不同的是，危險權限除此之外，在執行應用程式時，如果程式碼中存取了危險權限，還須出現請求權限的視窗，要求使用者允許存取資料。
範例截圖： 下圖為安裝前宣告存取權限：    下圖為存取危險權限訊息視窗： 
改善方式： 測試人員應嘗試檢測相關敏感性資料、資源、宣告權限用途是否符合該應用程式要求，如使用危險權限請求時需告知使用者。

檢測編號：A.1.2

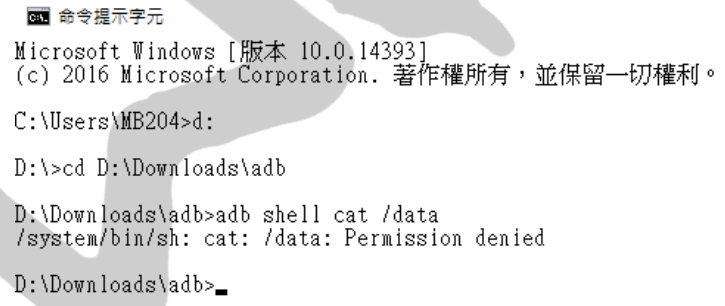
測試步驟：

透過滲透測試的方式，對行動應用程式進行越權與不同身分權限的提權存取嘗試，以檢測該行動應用程式使用者相關身分類別。

範例截圖：

下圖為 Root 權限狀態存取：

下圖為非 Root 權限狀態存取嘗試：



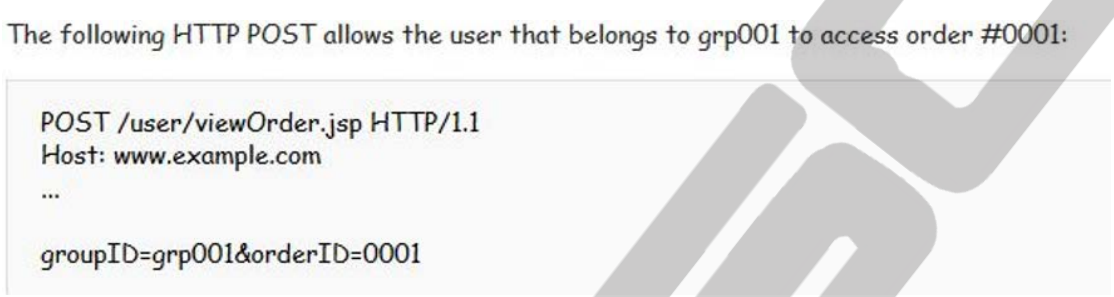
改善方式：

測試人員應透過對行動應用程式進行不同身分權限的提權存取測試是否符合使用者身分授權。

檢測編號：A.1.3
測試步驟： 檢查該行動裝置應用程式隱私權部分使用聲明中，是否如 A.1.1 檢測項目相對應資源說明給予使用者同意授予權限機制。
範例截圖： 
改善方式： 行動裝置應用程式應於安裝前提供授予同意權限機制用取得使用者授予權限。

檢測編號：A.1.4
測試步驟： 檢查該行動裝置應用程式隱私權部分使用聲明中，是否如 A.1.1 檢測項目相對應資源使用說明給予使用者權限列表上的權限授權，當給予授權時表示授權與該應用程式相關權限使用預設為授權狀態或直到應用程式移除為止，但需檢測是否為未經使用者授權准許使用非相關授權的函數。
範例截圖： 下圖為隱私權部分使用聲明：　　下圖為未經使用者授權准許預設權限：    
改善方式： 測試人透過反組譯工具查看 AndroidManifest.xml 中相關使用<uses-permission>宣告權限得知是否有使用未經使用者授權准許使用非相關授權的函數。

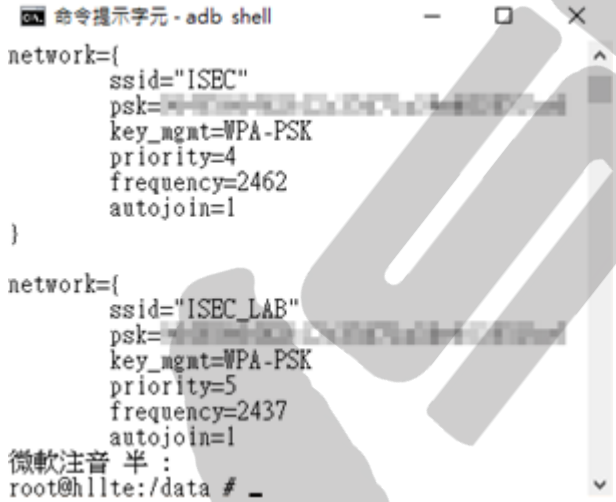
檢測編號：A.1.5
測試步驟： 檢查該行動裝置應用程式隱私權部分使用聲明中，是否如 A.1.1 檢測項目相對應資源使用說明給予使用者拒絕安裝該應用程式相關機制，同時讓使用者拒絕授予權限與行動裝置上相關資源權限權力。
範例截圖： 
改善方式： 行動裝置應用程式應於安裝前提供使用者拒絕安裝該應用程式相關機制，給予使用者拒絕授予權限與行動裝置上相關資源權限權力。

檢測編號：A.1.6
測試步驟： 行動裝置中每個應用程式，應記錄使用者在數據庫中產生的相關資訊和接收、刪除資訊等相關功能。測試人員應嘗試使用不同使用者來驗證是否可以取得用戶角色/權限不應該擁有的相關功能與權限。
範例截圖：    驗證不屬於 grp001 的用戶是否可以修改“groupID”和“orderID”的參數值來獲得該特權資訊訪問的權限。
改善方式： 測試人應驗證是否可透過修改參數值方式來取得不該擁有的權限。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：A.2.1
測試步驟： 行動裝置中應用程式於行動裝置應用程式密碼變更時檢測人員應檢查除管理員以外的使用者是否可以更改或重置除自己以外的帳戶密碼、密碼更改或重置過程是否容易受到任何類型的偽造、重置驗證密碼是否隨機生成、更改密碼前，重置密碼功能是否要求身分驗證。
範例截圖： 
改善方式： 密碼更改或重置功能需要一定程度上的保護機制，例如要求用戶在過程中重新驗證或與用戶在次確認的相關機制。

檢測編號：A.2.2
測試步驟： 檢測人員應檢查行動裝置應用程式中註銷功能應具備手動註銷的可控介面、沒有任何動作逾時應終止會話、伺服器端會話狀態是否安全、所有行動應用程式都應具備註銷按鈕於使用者能夠快速識別的位置，藉以減少會話劫持攻擊的可能性。
範例截圖：    會話劫持，對局域網中的主機通過實施 ARP 欺騙，進行 cookie 欺騙，從而達到劫持會話的效果[69]。
改善方式： 行動裝置應用程式中應具備完善的會話管理機制避免遭受會話劫持攻擊。

檢測編號：A.2.3
測試步驟： 檢測人員應檢查行動裝置應用程式中使用相關日誌;用於敏感信息，如用戶名/密碼、密碼、IP 地址、會話資訊，任何用戶相關資訊等。使用 ADB，Xcode，VisualStudio 和 BB SDK 等工具和調查日誌，Logcat，LogExpert 等相關其他工具檢測相關封裝內容是否存放敏感資訊。
範例截圖： 
改善方式： 應確保用戶端或伺服器端的相關日誌中不存儲敏感資訊。

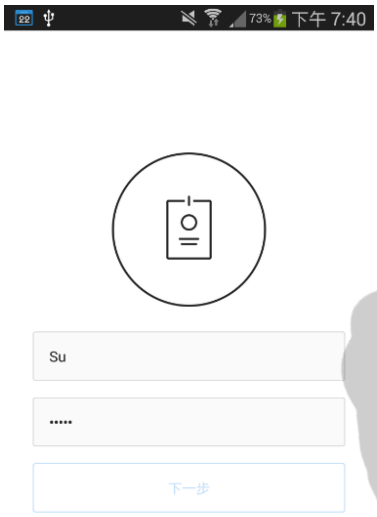
檢測編號：A.2.4

測試步驟：

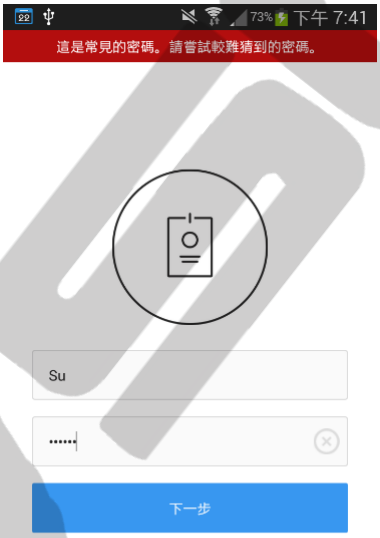
檢測人員應檢查行動裝置應用程式中，使用者是否使用不同字符符集，如小寫和大寫字母，數字和特殊符號、使用者是否可以頻繁變更密碼、密碼變更間隔限制、嘗試過多錯誤時鎖定、過往密碼歷史資料是否保留、驗證密碼是否使用包含用戶名帳號或其他帳戶相關資訊當作密碼使用。

範例截圖：

下圖為密碼長度限制：



下圖為密碼複雜度機制：



改善方式：

使用雙因素身分驗證或使用強密碼策略確保密碼長度及複雜性，藉以降低密碼破解的風險。

檢測編號：A.2.5
測試步驟： 檢查 form autocomplete 設定是否為"on/off"。主要瀏覽器覆蓋任何使用 autocomplete =“off 的密碼表單，預設不應該建議該功能為 On 狀態。這樣可能無意間使用於存儲在應用程式/瀏覽器上。
範例截圖： 一般的情況 IE5/6/7, FireFox 預設是將 AutoComplete 打開 <form> <input type="text" name="user_name"> </form> 但有時候並不需要 AutoComplete，例如需要使用者自己再次輸入密碼而非自動完成或者用 AJax 將後端整理好的資料讓使用者挑選時於上述的情況時，autocomplete 都會干擾使用者的操作，若讓特定的表單欄位不要自動完成(autocomplete)只要將此表單輸入元素的 autocomplete 屬性設置為 off 即可[70]。 <form> <input type="password" name="Password" autocomplete="off"> </form> 如果所有表單元素都不想使用 autocomplete 功能, 則可以用下面的方法： <form autocomplete = "off"> <input type="text" name="user_name"> <input type="password" name="Password" > </form>
改善方式： 建議將 Autocomplete 設置為 off 較為安全。

檢測編號：A.2.6

測試步驟：

應用程式在越獄/Root 環境下應檢測運行存取特殊檔案與特殊權限的過程。

應用程式在非越獄/Root 環境下特殊檔案與特殊權限嘗試應禁止。

範例截圖：

下圖為越獄 Root 環境下存取：

下圖為非越獄 Root 環境下存取：

改善方式：

建議測試人員應確保非越獄 Root 環境下不可存取相關敏感訊息。

檢測編號：B.1.1
測試步驟： 像 SQLite 可以像 Web 應用程式一樣遭受注入式攻擊。當應用程式包含許多不同的用戶付費/解鎖相關內容時，可透過此類的注入方式檢測資料庫的風險。
範例截圖： SELECT * FROM TABLE_NAME WHERE USERNAME = ' ' AND PASSWORD = ' '; Passing it to SQLiteDatabase.rawQuery();   
改善方式： 設計 SQLite 查詢時，確保使用者提供的數據被傳遞給參數化查詢。 透過查詢使用的格式說明符發現一般惡意的使用者提供的數據會插入“%@”而不是“?”的參數化查詢說明符，處理動態查詢或內容提供者確保使用者正常使用參數化查詢。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.1.2

測試步驟：

XML 可以像 Web 應用程式一樣注入，攻擊者可透過修改 XML 資料格式，增加新的 XML 節點藉此對資料處理流程產生影響。

範例截圖：

一般註冊後的資料紀錄

```
<?xml version="1.0" encoding="UTF-8"?>
<USER role="guest_role">
    <name>user1
    </name>
    <email>user1@a.com
    </email>
</USER>
```

攻擊者輸入自己 email 時，可以輸入如下代碼：

```
user1@a.com</email></USER><USERrole="admin_role"><name>lf</name><email>user2@a.com
```

最後註冊成功資料變成：

```
<?xml version="1.0" encoding="UTF-8"?>
<USER role="guest_role">
    <name>user1
    </name>
    <email>user1@a.com</email>
</USER>
<USER role="admin_role">
    <name>lf</name>
    <email>user2@a.com
    </email>
</USER>
```

中間多出了一條注入的 role=“admin_role”的管理員 lf，達到攻擊目的[71]。

改善方式：

對關鍵字符串進行轉義：& → &;< → <;> → >;” → ”;’ → ’，在 XML 中對資料部分，單獨做轉義。

檢測編號：B.1.3

測試步驟：

逆向工程 Android Mobile App（APK 文件）：

1. 選擇要進行逆向工程的 APK，使用 7zip 解壓縮軟體。
2. 可以看到一個為 classes.dex 的文件。
3. 使用 tooldex2jar 將 classes.dex 轉換為可讀的 jar。
4. 使用任何 Java 反編譯器，打開新轉換的文件“classes_dex2jar”，研究使用 jdgui 免費的工具。
5. 之後可以看到裡面的所有的文件。
6. 在代碼中查找編碼的敏感信息(註:如果代碼沒有被模糊處理)。
7. 檢查 BuildConfig.class 以查看應用程式是否以 DEBUG 模式釋放。此外，還可以檢查其他安全控制。
8. 打開 AndroidManifest.xml 文件，檢查 Android 應用程式在安裝過程中所需的權限。

逆向工程 IOS Mobile App（IPA 文件）：

1. 選擇 IPA 文件，並使用 7zip 解壓縮。
2. 可以看到有效載重文件夾和 PList 文件。
3. 檢查有效載重文件夾中的敏感資料和腳本。
4. 可以使用 Hopper Tool 逆向工程的 IOS 應用程式，對於有效分析使用 XCodeon MacBook。

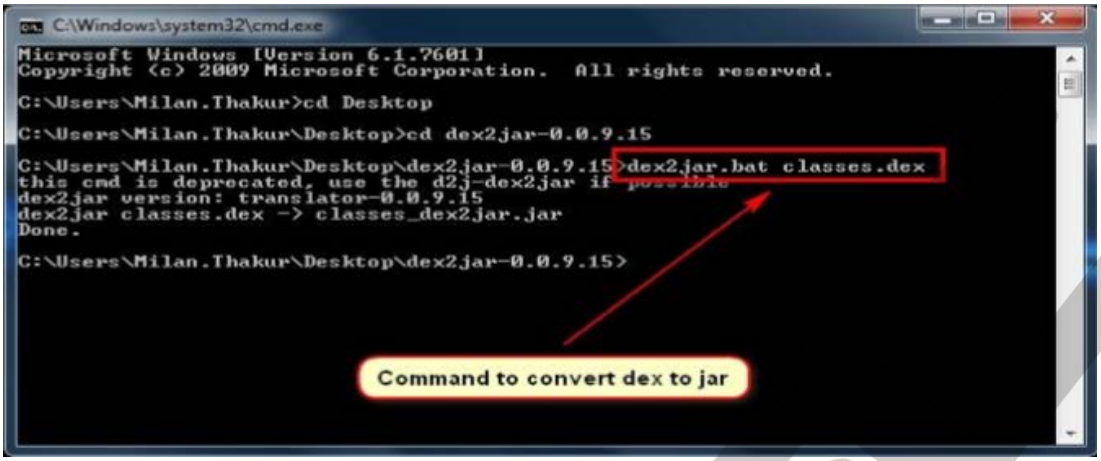
逆向工程 Windows 8 Mobile App（XAP File）：

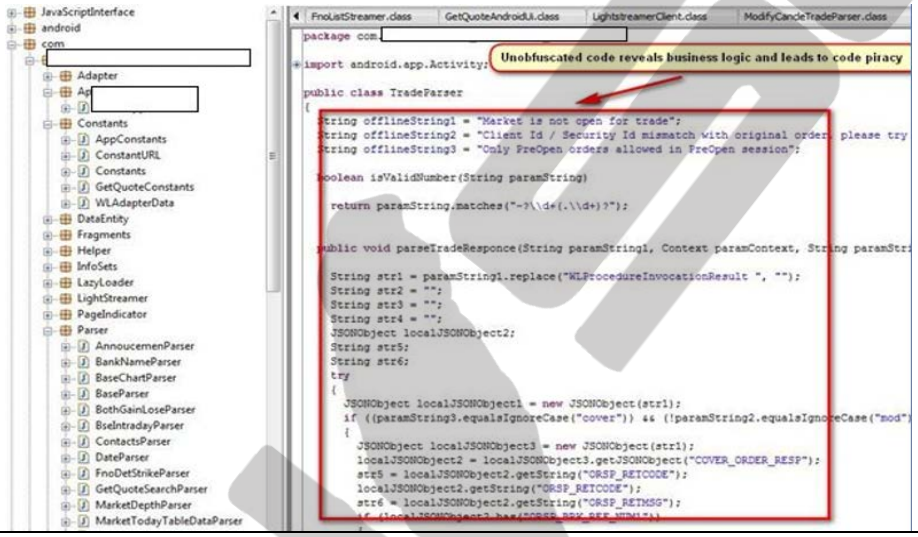
1. 選擇 XAP 文件並使用 7zip 解壓縮。
2. 目前擁有 DLL 列表。
3. 使用 DLL 反編譯器打開 DLL 文件，研究使用 JetBrainsDotPeekby。
4. 打開 DLL 後，可以看到所有和其他部分的明文代碼。
5. 要進一步分析，可以在具有 Windows Mobile SDK 的 Windows8 64 位筆記本電腦上安裝 Visual Studio 2013。

逆向工程黑莓手機應用（COD 文件）：

1. 使用 BlackBerry JDE 可以打開 COD 文件。

範例截圖：





改善方式：

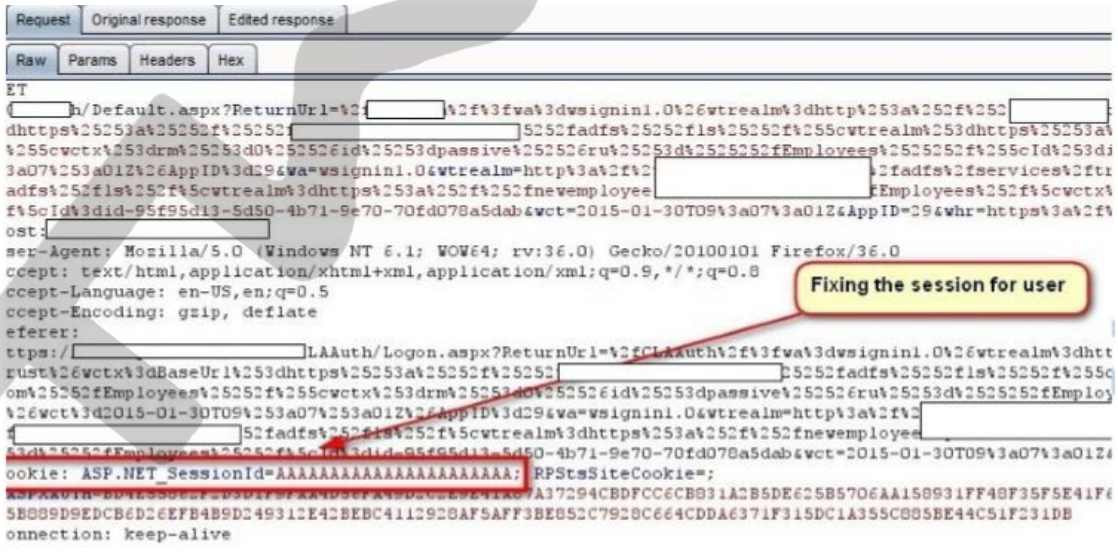
這是一種防止代碼逆向工程控制的方法，提升整體的安全同時攻擊應用程式被攻擊所需花費的時間。

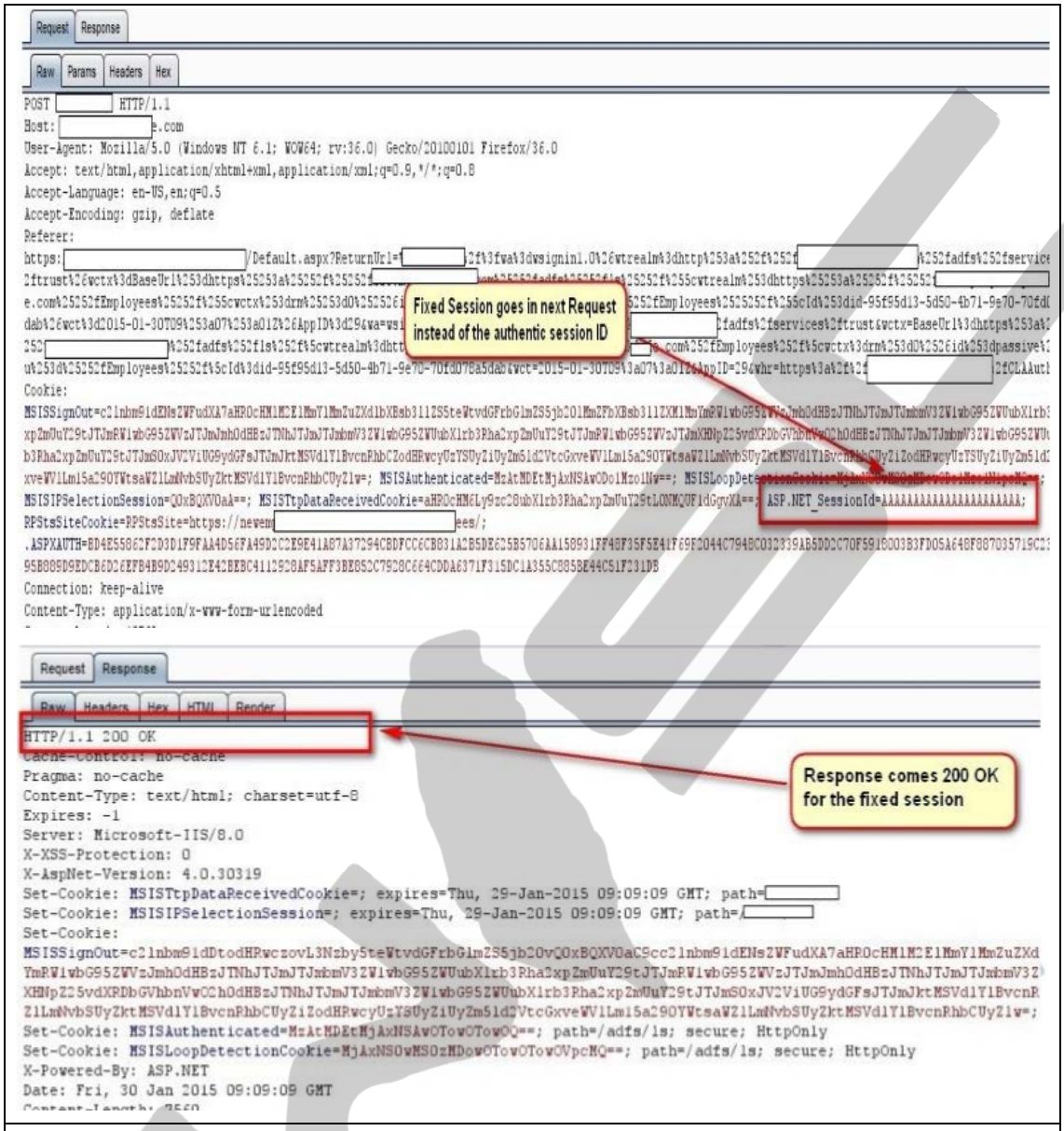
- 抽象敏感軟體中靜態的 C 函式庫。
- 通過使用第三方商業軟體或開源解決方案執行自動化代碼模糊化程式，在可行的情況下模糊所有敏感的應用程式代碼。
- 對於包含敏感資訊的應用程式實施防調試技術（例如防止調試器從附加到過程; android：debuggable =“false”）。
- 確保日誌記錄被禁止，日誌可能會被詢問具有 readlogs 權限的其他應用程式（例如，在重新啟動之前，任何其他應用程式都可以讀取 Android 系統日誌）。
- 應用程式開發用於支持（iOS 4.3 及更高的版本，Android 4.0 及更高的版本）架構，利用地址空間佈局隨機化（ASLR）方式隱藏，防止應用程式內存的轉儲。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.1.4
測試步驟： 測試者應對行動裝置中應用程式進行逆向工程，並檢查以下內容： 支付出入口編碼憑證應用程式編碼服務器和應用程式的具體細節開發人員名稱與註釋解釋代碼片段。
範例截圖：無
改善方式： 測試者應透過安全代碼審查或逆向工程方法驗證，以確保沒有敏感信息的編碼使用於應用程式代碼中。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.1.5
測試步驟： 1. 登錄行動應用程式。 2. 攔截含“set-cookie value”伺服器的回應。 3. 透過攻擊者定義的值更改 cookie，轉發對應用程式的回應。 4. 查看應用程式是否繼續使用攻擊者定義值進行進一步的通訊。 附加檢查： 1. 未能正確旋轉小型文字檔案。 2. 缺乏足夠的逾時保護。 3. 不安全建置的代碼。
範例截圖： 



改善方式：

需要完善的會話處理，確保行動應用程式代碼在使用者的行動裝置應用會話的生命週期內正確建置維護和銷毀會話代碼。

TrackAuthentication 狀態變更為類似事件：

- 從匿名用戶切換到登錄用戶。
- 從任何登錄用戶切換到另一個登錄用戶。
- 從普通用戶切換到特權用戶。
- 逾時。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.1.6
簡述： 使用調試器來處理應用程式變數在執行時使用， Android 應用程式可以解壓縮、修改、重新組合與轉換，藉以取得底層應用程式代碼的存取權限。 調試器是一種掛鉤附加到特定應用程式代碼的技術。當達到特定的代碼段，執行就會暫停，使我們能夠分析局部變數，轉儲類值，修改值，並且通常與程式狀態進行交互，當我們準備好後可以恢復執行。 如果已經完成了與 Android 應用程式的任何工作，則不需要所需任何工具： 應用程式的安裝包 Java SDK Android SDK
測試步驟： 應用程式的.apk 中包含的 AndroidManifest.xml 實際上具有一個 android：可調試器設置，允許應用程式調試。所以需使用 APK Manager 來解壓縮安裝包，並添加 android：debuggable =“true”。
範例截圖： JDB 命令用於調試部分： stop in [function name]-Set a breakpoint next-Executes one line step-Step into a function step up-Step out of a function print obj-Prints a class name dump obj-Dumps a class print [variable name]-Print the value of a variable set [variable name] = [value]-Change the value of a variable
改善方式： 需注意除錯設定中的設定值是否安全。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.1.7
測試步驟： 惡意使用者可以透過該漏洞借助 XXE 或 SSRF 等其他漏洞發送未經授權的電子郵件也可以利用該漏洞繞過防火牆，從而打開並連接一個 TCP 端口。該漏洞主要原因為 Java 和 Python 在解析用戶發送的 FTP URL 時導致惡意使用者可透過惡意的 URL 來注入並執行 FTP 命令。
範例截圖： Klink 演示相同類型的漏洞可用來欺騙 Java runtime 向遠程伺服器發起 FTP 連接，URL 形式是： ftp://user:password@host:port/file.ext 結果發現 Java FTP 客戶端並不會過濾特殊的 CR 和 LF 字符，而會解析這些字符。CRLF 的意思就是回車(CR, ASCII 13, \r) 換行(LF, ASCII 10, \n)。在 FTP URL 中插入這些字符會導致 Java FTP 客戶端執行這些命令，甚至還可以執行 SMTP (Simple Mail Transfer Protocol)命令，因為 SMTP 與 FTP 語法很相似[72]。
改善方式： 建議防火牆廠商在 Java 和 Python 工程師修復漏洞之前關閉傳統的 FTP 轉換模式，用戶應該禁用瀏覽器的 Java 插件並且取消.jnlp 文件與 Java Web Start 的關聯，同時對 Java 和 Python 應用應該審計 SSRF 和 XXE 漏洞。

檢測編號：B.2.1
測試步驟： 1. 將應用程式安裝於行動裝置上。 2. 使用偽造憑證多次嘗試登錄。 3. 在需要離線存取資料的情況下，在 X 次無效密碼嘗試（例如 10 次）後執行帳戶/應用程式鎖定/應用程式資料刪除。 4. 在 X 分鐘不活動（例如不活動 10 分鐘）後，自動應用程式關閉/鎖定。
範例截圖： 
改善方式： 確保嘗試登錄次數（不超過 10 次嘗試）和應用程式使用者界面不活動時（不超過 3、5 分鐘）應用程式應具自動關閉應用程式或鎖定功能。

檢測編號：B.2.2

簡述：

輕量級目錄訪問協議（LDAP）用於存儲有關用戶，主機和其他對象的資訊。LDAP 注入式伺服器端攻擊，允許公開修改或插入 LDAP 結構中，導致用戶和主機的敏感資訊暴露於危險之中，透過操作輸入參數方式傳遞到內部搜索新增或修改功能，Web 應用程式可以使用 LDAP 來讓用戶在企業結構中驗證或搜索其他用戶的信息。LDAP 注入攻擊的目標是將 LDAP 搜索過濾器的元字符放入應用程式執行的查詢中。

LDAP 搜索過濾器以前綴符號構造，搜索過濾器上的虛擬碼條件如下所示：

find("cn=John & userPassword=mypass")

被表示：

find("(&(cn=John)(userPassword=mypass))")

使用以下字符可以應用 LDAP 搜索過濾器上的布林條件和組聚合：

Metachar 含義

& Boolean AND
| Boolean OR
! Boolean NOT
= Equals
~= Approx
>= Greater than
<= Less than
* Any character
() Grouping parenthesis

以下允許測試者利用 LDAP 注入漏洞來實現以下目的：

1. 未經授權內容的存取。
2. 避免應用限制。
3. 收集未經授權的資訊。
4. 新增或修改 LDAP 結構中的內容。

測試步驟：

以下給出的例子進行描述

範例截圖：

例 1：搜索過濾器

假設有一個使用以下的搜索過濾器中 Web 應用程式：
searchfilter="(cn="+user+")"
是由以下的 HTTP 請求實例：
http://www.example.com/ldapsearch?user=John
如果值''''替換為'*'，通過發送請求：
http://www.example.com/ldapsearch?user=*
過濾器如下所示：
searchfilter="(cn=*)"
每個'cn'屬性都等於任何東西。
如果應用程式遭受 LDAP 注入攻擊，會顯示部分或全部用戶屬性取決於應用程式的執行流程和 LDAP 連接用戶的權限。
測試者可以透過插入參數'('，'|'，'&'，'*'和其他字符，檢查應用程式是否有錯誤。
例 2：登錄
Web 應用程式使用 LDAP 在登錄過程中檢查用戶憑證，容易遭受 LDAP 注入攻擊，可透過一直注入為 true 的 LDAP 查詢來繞過驗證檢查（以類似於 SQL 和 XPATH 注入的方式）。
假設 web 應用程式使用過濾器來配對 LDAP 用戶/密碼。
searchlogin=
"(&(uid="+user+")(userPassword={MD5}"+base64(pack("H*",md5(pass)))+"
))";
通過使用以下值：
user=*)(uid=*))(|(uid=* pass=password
搜索過濾器會導致：
searchlogin="(&(uid=*)(uid=*))(|(uid=*)(userPassword={MD5}X03MO1qnZd
YdgyfeuILPmQ==))";
透過一直為 true 的方式測試者是否可取得登錄狀態。

改善方式：
建議避免使用 LDAP 搜索過濾器的元字符於應用程式中。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.2.3

測試步驟：

命令注入攻擊方式目標是透過容易遭受攻擊的應用程式在主機操作系統上執行任意命令。當應用程式把惡意使用者提供的不安全數據（例:表單 Cookie、HTTP 頭等）傳遞到系統 shell 時，來進行命令注入攻擊，攻擊時攻擊者提供的操作系統命令透過遭受攻擊應用程式的權限執行，命令注入攻擊主要是因為輸入驗證機制不完善。命令注入攻擊與代碼注入不同，代碼注入允許攻擊者新增自己的代碼讓應用程式執行。代碼注入中攻擊者擴充應用程式中默認功能，不需要執行系統命令。

範例截圖：

範例 1

以下代碼是 UNIX 命令 cat 的包裝，將文件內容印出標準輸出內容。

```
int main(int argc, char **argv) {
 char cat[] = "cat ";
 char *command;
 size_t commandLength;

 commandLength = strlen(cat) + strlen(argv[1]) + 1;
 command = (char *) malloc(commandLength);
 strncpy(command, cat, commandLength);
 strncat(command, argv[1], (commandLength-strlen(cat)) );
system(command);
 return (0); }
```

正常使用時，輸出僅僅是請求的文件的內容：

```
$ ./catWrapper Story.txt ...
```

加上分號和另一條命令由此處結束後，catWrapper 執行命令：

```
$ ./catWrapper "Story.txt; ls"
 Story.txt          doubFree.c         nullpointer.c
 unstosig.c         www*               a.out*
 format.c           strlen.c           useFree*
 catWrapper*        misnull.c          strlength.c        useFree.c
 commandinjection.c      nodefault.c     trunc.c    writeWhatWhere.c
```

catWrapper 設置具有比標準用戶更高的權限，固擁有更高的權限執行任意命令。

範例 2

以下簡單程式接受文件作為命令行參數，將該文件的內容顯示回用戶。此程式安裝了 setuid root，允許系統管理員檢查特權系統文件，而不會給予修改或破壞系統的能力。

```
int main(char* argc, char** argv) {
 char cmd[CMD_MAX] = "/usr/bin/cat ";
 strcat(cmd, argv[1]);
 system(cmd);
 }
```

由於程式以 root 用戶權限運行，system ( ) 以 root 權限執行。用戶指定標準文件名稱，則呼叫按預期的工作。但如果攻擊者傳遞“; rm -rf /”形式的字符串，則由於缺少參數，system ( ) 無法執行 cat，然後依次遞歸刪除 Root 分區的內容。

範例 3

需要擁有特權程式使用以下代碼環境變量$ APPHOME 確定應用程式的安裝目錄，在該目錄中執行初始化腳本。

```
...
 char* home=getenv("APPHOME");
char* cmd=(char*)malloc(strlen(home)+strlen(INITCMD));
 if (cmd) {
 strcpy(cmd,home);
 strcat(cmd,INITCMD);
 execl(cmd, NULL);
 }
 ...
```

與範例 2 中一樣，此範例中的代碼允許攻擊者使用應用程式來提升權限執行任意命令。在範例中，攻擊者可以修改環境變數$ APPHOME 來指定包含惡意版本的 INITCMD 的不同路徑。由於該程式沒有驗證從環境讀取的值，透過控制環境變數，攻擊者可以混淆應用程式執行惡意代碼。

攻擊者正在使用環境變數來控製程式調用的命令，在這個範例中環境的影響是顯而易見的。後續將把注意力轉移到攻擊者改變命令解釋方式時可能發生的情況。

範例 4

以下代碼來自 Web 的 CGI 實用程式，允許用戶更改密碼。NIS 下的密碼更新過程包括在/ var / yp 目錄中運行 make。注意由於程式更新密碼記錄，所以已經安裝了 setuid root。

使用 make 如下：

```
system("cd /var/yp && make &> /dev/null");
```

與以前的範例不同，此範例中的命令是虛擬碼，因此攻擊者無法控制傳遞給 system（）的參數。由於程式沒有指定 make 的絕對路徑，並且在使用命令之前不刪除任何環境變數，所以攻擊者從 shell 提示可以修改$ PATH 變數以指向名稱為 make 的惡意二進制文件。由於程式已經安裝了 setuid root，所以攻擊者的版本現在以 root 權限執行。

環境在程式中執行系統命令方面發揮了重要作用，像 system（）和 exec（）使用的程式環境函數，因此攻擊者可能影響這些使用的行為。

Java 的 Runtime.exec 與 C 的系統功能完全相同，兩者都允許使用新的程式/進程。然而，C 的系統函數將參數傳遞給 shell（/ bin / sh）以進行解析，而 Runtime.exec 嘗試將字符串拆分成一個單詞數組，並使用剩餘的單詞執行數組中的第一個單詞作為參數。 Runtime.exec 不會嘗試在任何時候使用 shell。關鍵區別在於可導致錯誤（使用“&”，“&&”，“|”，“||”等等）重定向輸入和輸出）的 shell 提供的大部分功能將簡單地結束作為參數傳遞給第一個命令，可能導致語法錯誤或無效參數拋出。

改善方式：

行動裝置應用程式應提供完善的輸入驗證機制。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.2.4

測試步驟：

檢查內存轉儲中的用戶或會話相關資訊，下列提供工具可幫助進一步分析：

- 開源 Android 應用及框架。
- Android 資料提取器 Lite。
- BitPim，允許查看和控制 LG，三星，三洋等許多 CDMA 手機上的資料。
- LiME（以前的 DMD）是一個可加載核心模塊（LKM）允許從 Linux 和 Linux 設備（如由 Android 供電的設備）中取得揮發性記憶體存儲器。
- iPhone-Backup-Analyzer-2。
- 使用 Drozer。
- 使用 Eclipse 一部分的內存分析器。

範例截圖：

Android：

因為 Java 的 Garbage Collection(GC)的特性是，開發人員無法確認何時資料會 真的被清空，而 String 是 immutable，基本上無法主動清空，就算指向 null，還 是需要等待 GC 完成後，它才不會存在於記憶體中，最有效的方式為改以 byte array 來存放，在使用完之後，直接將每一個 byte 值覆寫為 0，如此一來立刻就能 將資料從記憶體中清空，防止惡意人士從記憶體中盜取 [51]。

```
SecretKey key = KeyGenerator.getInstance("DES").generateKey();
 byte[] data = key.getEncoded();//處理完相關作業之後立即清空，以避免存在於 RAM
for (byte oneByte:data){      oneByte=0;
}
```

改善方式：

確保行動應用程式不洩漏任何通過內存轉儲的敏感信息。

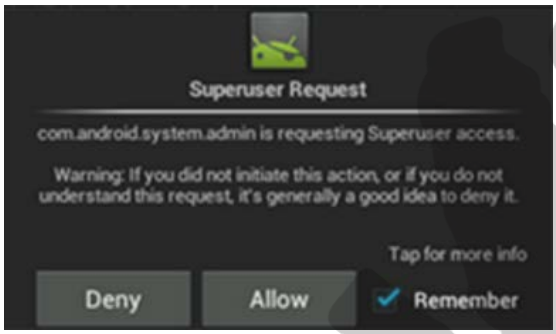
檢測編號：B.2.5

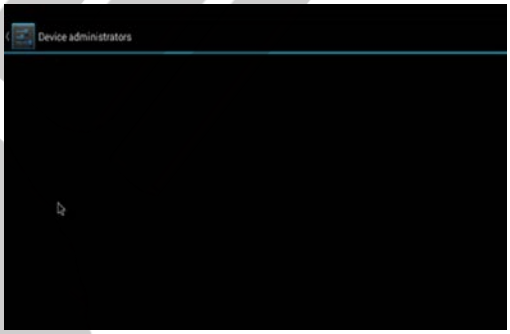
測試步驟：

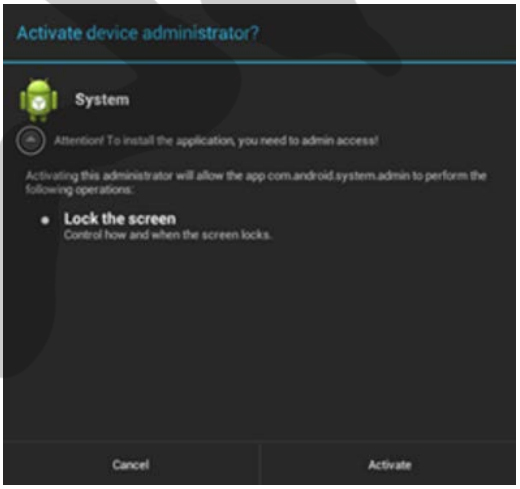
行動應用程式透過惡意方式所產生漏洞導致資料暴露在危險中，如透過木馬程式植入到行動裝置中，導致使用者的行動裝置產生漏洞造成使用者資料暴露在危險中。

範例截圖：

ANDROIDOS_OBAD，此類木馬程式會竊取手機資訊，並可未經使用者同意擅自訂購高付費服務，導致電信帳單爆增。ANDROIDOS_OBAD 具備隱形以及防止被移除的功能，當使用者啟動該木馬後，會被要求授予手機 root 以及設備管理員等權限，使用者若不同意，只要開啟裝置就會不停跳出要求使用者同意的視窗；一旦取得手機的設備管理員權限，其將可以在手機桌面和設備管理員管理畫面上隱形，無法被刪除，在未解除管理員權限的情況之下，使用者或是資訊安全軟體將無法清除此隻木馬程式[73]。







改善方式：

建議測試人員使用滲透檢測方式監控行動裝置中惡意程式的行為動作。

檢測編號：B.3.1

測試步驟：

1. 安裝應用程式。
2. 在應用程式中找輸入或搜索選項。
3. 透過製作一個搜索查詢：“> <SCRIPT> var + img = new + Image ( ) ; img.src =” http：// hacker / “%20 +%20document.cookie; </ SCRIPT> 。
4. 在應用程式中查找隱藏的 iframe。
5. 使用<iframe src = “http://www.evilsite.com” />。

範例截圖：

改善方式：

確保所有 UIWebView 不執行沒有正確輸入的驗證，可能為危險的 JavaScript 字符應用過濾器，在給予前使用白名單超過黑名單字符方式。呼叫 Safari，而不是在可以存取應用程式的 UIWebkit 內部放置。

驗證任何 WebViews（通常是默認值）禁止 JavaScript 和 Plugin 支援。

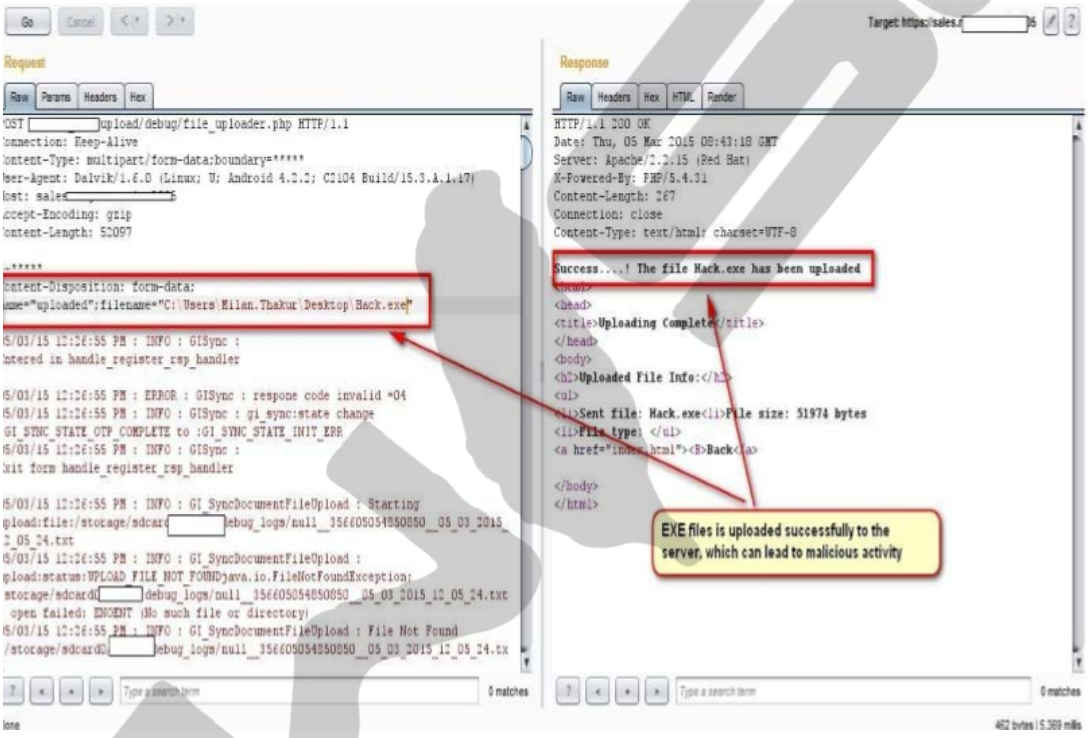
參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.3.2

測試步驟：

1. 安裝應用程式。
2. 找尋文件上傳功能。
3. 嘗試上傳任何惡意文件，如.exe、.msi 或其他可執行文件。
4. 應用程式 UI 限制上傳某些文件類型，則使用任何中間代理（如 Burp / ZAP）攔截流量。
5. 攔截上傳請求後，使用 LFI（本地文件包含）或 RFI（遠程文件包含）方法將惡意文件上傳到伺服器。

範例截圖：



改善方式：

使用標題中“Content-Type”或採用文件類型識別器保護文件的黑名單或白名單但無法都透過這種方式對於此漏洞進行保護。每個應用程式都必須能夠讓用戶傳送文件，應具有一個完整的機制來驗證上傳的文件是否不包含惡意代碼，上傳的文件不應該存儲在用戶或攻擊者可以直接存取位置。可使用黑名單方式、白名單方式、使用檔頭文件中的“Content-Type”與文件類型識別器來改善。

Ref:https://www.owasp.org/index.php/Unrestricted_File_Upload

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.3.3
適用範圍進行測試： 大多數行動裝置應用程式漏洞發生於開發人員將敏感訊息安全的存儲於應用程式中或用戶端控制來強制實施服務器安全性。應用商店擁有超過 1,000,000 個應用程式，而 IOS 漏洞連開發人員也未注意這麼簡易的功能，什麼時候功能成為一個漏洞？為了讓應用程式之間切換時提供無縫的視覺轉換，IOS 使用了一種緩存技術。當使用者雙擊主頁按鈕時會彈出最近使用的應用程式列表，如下所示。  這些未使用的應用程式處於暫停狀態，因此不會佔用不必要的系統資源。 IOS 緩存應用程式的最後一個屏幕的屏幕截圖，當使用者點擊它時，應用程式將恢復執行，工作原理是屏幕截圖首先被加載，並且應用程式於不久之後重新啟動。這種緩存技術為用戶提供了他們的應用程式立即恢復的印象，手機在應用程式之間切換時非常快。但如果丟失手機或被盜時是否會造成安全性上風險。
測試步驟：無
範例截圖：無
改善方式：本研究規範主要對於 Android 部分，該項目屬於 IOS 存在風險之一，Android 可能也存在相關風險。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：B.3.4
測試步驟： 測試人員可透過反組譯方式針對混淆規則進行修改來查閱該應用程式是否造成安全性上的威脅。
範例截圖： 惡意程式將自己 APK 加密[74]  惡意程式將自己的 Shellcode 加密  惡意程式使用 Process Watcher(執行程式監控程式)確保程式正常執行     惡意程式將 APK 包裝檔重新命名 
改善方式： 為了保護應用程式原始碼安全性，應盡可能隱藏和保護其流程，使用者須注意以下做法： 1. 於官方應用商店下載、安裝、更新。 2. 安全的 Android 設備與解決方案須不斷更新，以防止最新的威脅，並在安裝之前阻止惡意程式。

檢測編號：C.1.1

測試步驟：

測試人員可透過中間人攻擊方式攔截傳輸訊息的內容。嘗試監聽未加密訊息傳遞，包括基於 Linux 系統的防火牆 iptables 設置阻擋 SSL port443 連線的規則，測試是否會在 port 443 受阻檔的情況下改經由 port 80 進行明碼傳輸，之後使用封包監聽軟體來監聽通訊連線，並利用 Wireshark 進行分析。

範例截圖：

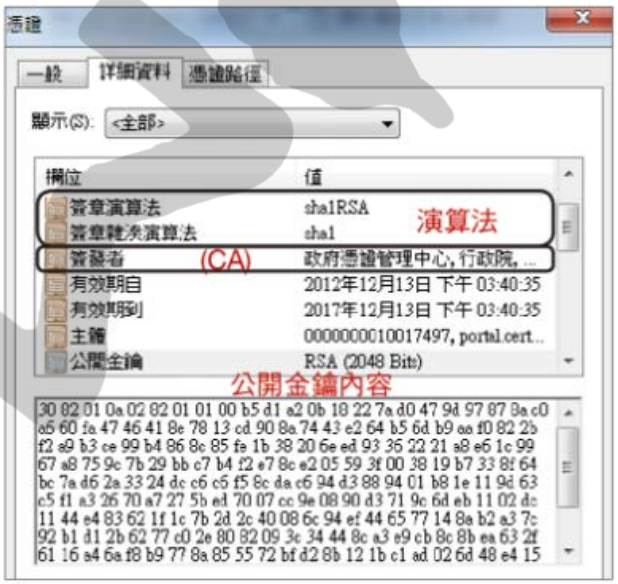
使用擷取工具擷取網路介面封包，針對攔截訊息之內容進行分析[75]。

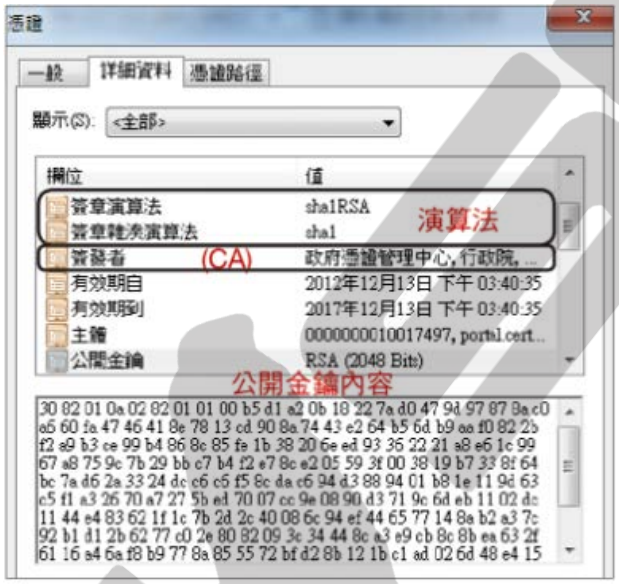
```
.036924 IP 192.168.0.1.110 > 192.168.0.2.1336: P 1:19(18) ack 1 win 5840
0x0000:  4500 003a 23f0 4000 4006 952b c0a8 0034  E..:#.@.@..+...4
0x0010:  c0a8 001e 006e 0538 3fe1 7bf7 3484 b47c  .....n.8?.{.4..|
0x0020:  5018 16d0 f6fb 0000 2b4f 4b20 646f 7665  P.......+OK.dove
0x0030:  636f 7420 7265 6164 792e                 cot.ready.
.156500 IP 192.168.0.2.1336 > 192.168.0.1.110: . ack 19 win 44982
0x0000:  4500 0028 d9dc 4000 8006 9f50 c0a8 001e  E..(..@....P....
0x0010:  c0a8 0034 0538 006e 3484 b47c 3fe1 7c09  ...4.8.n4..|?.|.
0x0020:  5010 afb6 d3e9 0000 0000 0000 0000       P..............
.156566 IP 192.168.0.1.110 > 192.168.0.2.1336: P 19:21(2) ack 1 win 5840
0x0000:  4500 002a 23f2 4000 4006 9539 c0a8 0034  E..*#.@.@..9...4
0x0010:  c0a8 001e 006e 0538 3fe1 7c09 3484 b47c  .....n.8?.|.4..|
0x0020:  5018 16d0 5fbc 0000 0d0a                 P..._.....
.156924 IP 192.168.0.2.1336 > 192.168.0.1.110: P 1:14(13) ack 21 win 44980
0x0000:  4500 0035 d9dd 4000 8006 9f42 c0a8 001e  E..5..@....B....
0x0010:  c0a8 0034 0538 006e 3484 b47c 3fe1 7c0b  ...4.8.n4..|?.|.
0x0020:  5018 afb4 d5e2 0000 5553 4552 2063 6172  P.......USER.car
0x0030:  7269 650d 0a                             rie..
.156969 IP 192.168.0.1.110 > 192.168.0.2.1336: . ack 14 win 5840
0x0000:  4500 0028 23f4 4000 4006 9539 c0a8 0034  E..(#.@.@..9...4
0x0010:  c0a8 001e 006e 0538 3fe1 7c0b 3484 b489  .....n.8?.|.4...
0x0020:  5010 16d0 6cc1 0000                      P...l...
.157145 IP 192.168.0.1.110 > 192.168.0.2.1336: P 21:26(5) ack 14 win 5840
0x0000:  4500 002d 23f6 4000 4006 9532 c0a8 0034  E..-#.@.@..2...4
0x0010:  c0a8 001e 006e 0538 3fe1 7c0b 3484 b489  .....n.8?.|.4...
0x0020:  5018 16d0 ec57 0000 2b4f 4b0d 0a         P....W..+OK..
.157326 IP 192.168.0.2.1336 > 192.168.0.1.110: P 14:27(13) ack 26 win 44975
0x0000:  4500 0035 d9de 4000 8006 9f41 c0a8 001e  E..5..@....A....
0x0010:  c0a8 0034 0538 006e 3484 b489 3fe1 7c10  ...4.8.n4...?.|.
0x0020:  5018 afaf 6d8a 0000 5041 5353 2032 3735  P...m...PASS.123
0x0030:  3134 300d 0a                             456..
.172331 IP 192.168.0.1.110 > 192.168.0.2.1336: P 26:40(14) ack 27 win 5840
0x0000:  4500 0036 23f8 4000 4006 9527 c0a8 0034  E..6#.@.@..'...4
0x0010:  c0a8 001e 006e 0538 3fe1 7c10 3484 b496  .....n.8?.|.4...
0x0020:  5018 16d0 4e57 0000 2b4f 4b20 4c6f 6767  P...NW..+OK.Logg
0x0030:  6564 2069 6e2e                           ed.in.
```

改善方式：

透過網路傳輸敏感性資料時訊息皆須加密，對於憑證也需有充分地進行驗證來抵抗防止中間人攻擊，確保傳輸過程中敏感性資料的安全性。

檢測編號：C.1.2
測試步驟： 使用滲透工具監控該檢測行動裝置與遠端連結伺服器主機傳送過程中的封包，透過滲透工具傳送交談識別碼來達到重送攻擊測試，查看是否存在重送攻擊風險。
範例截圖： 攻擊者於竊聽網路後，劫取並記錄了通訊雙方傳送的憑證資訊，之後便重送憑證資訊以假冒某個特有共用金鑰的使用者，以達到存取系統的入侵目的。
改善方式： 1. 金鑰設定採用較短有效時間限制 2. 採用完整雙向驗證的安全性協定

檢測編號：C.1.3
測試步驟： 使用通訊加密演算法檢測軟體，檢查受測軟體所存取之付費功能伺服器，使用加密演算法為 FIPS 140 核准的加密編譯演算法或由申請者提供同等安全性之佐證資料。
範例截圖： 查閱憑證簽章演算法為核准之安全演算法[76] 
改善方式： 使用 FIPS 140 核准的加密編譯演算法或由申請者提供同等安全性之佐證資料提高通訊過程的安全性。

檢測編號：C.1.4
測試步驟： 使用通訊加密演算法檢測軟體，檢查受測軟體所存取之伺服器，使用的加密演算法為 FIPS 140 核准之加密編譯演算法或由申請者提供同等安全性之佐證資料。
範例截圖： 查閱憑證簽章演算法為核准之安全演算法[76]。 
改善方式： 使用 FIPS 140 核准的加密編譯演算法或由申請者提供同等安全性之佐證資料提高通訊過程的安全性。

檢測編號：C.1.5

測試步驟：

在行動應用程式處於“離線”模式時執行二進制攻擊。透過攻擊測試人員將強制應用程式繞過離線驗證，執行需要離線驗證的功能，此外，測試人員應測試透過從行動應用程式功能的任何 POST/GET 請求中刪除任何會話代碼來匿名執行任何後端伺服器功能，在行動裝置內進行授權決策時授權要求更為薄弱。

範例截圖：

下圖為透過 adb 刪除 gesture.key，繞過驗證：

改善方式：

開發人員應該假設惡意用戶可繞過所有用戶端授權和身份驗證控制，必須在伺服器端重新執行授權和認證控制。

由於離線使用要求，可能需要行動應用程式在代碼中執行本地身份驗證或授權檢查，這種情況下，開發人員應在代碼中進行本地完整性檢查，以檢測任何未經授權的代碼更改。

檢測編號：C.1.6

測試步驟：

MSISDN 號碼可用於驗證來自行動應用程式的關鍵交易。

1. 用戶必須在行動資料上，如果用戶正在使用 Wi-Fi，則不會收到 MSISDN 資訊。
2. 用戶的行動網路必須支持在 HTTP 中傳遞 MSISDN。

查看您的標題以尋找以下任何內容，是否於 MSISDN 列表中;

此為行動開發過程中遇到的問題：

- X-MSISDN
- X_MSISDN
- HTTP_X_MSISDN
- X-UP-CALLING-LINE-ID
- X_UP_CALLING_LINE_ID
- HTTP_X_UP_CALLING_LINE_ID
- X_WAP_NETWORK_CLIENT_MSISDN

Ref:

http://developer.android.com/reference/android/telephony/TelephonyManager.html#getLine1Number %28%29

https://mobiforge.com/design-development/useful-x-headers

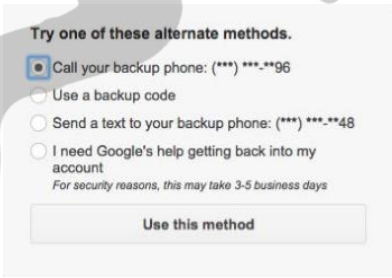
範例截圖：

Header Name	Content
HOST	[redacted].org
USER-AGENT	Mozilla/5.0 (X11; U; Linux armv7l; en-US; rv:1.9.2alpre) Gecko/20090928 Firefox/3.5 Maemo Browser 1.4.1.15 RX-51 N900
ACCEPT	image/png,image/*;q=0.8,*/*;q=0.5
ACCEPT-LANGUAGE	en
ACCEPT-ENCODING	*
ACCEPT-CHARSET	ISO-8859-1,utf-8;q=0.7,*;q=0.7
REFERER	http://[redacted].org/blog/
X-UP-SUBNO	1233936xxx-346677xxx
X-UP-FORWARDED_FOR	10.248.240.209
X-FORWARDED_FOR	10.248.240.209
X-UP-CALLING-LINE-ID	491522852xxxx
X-UP-SUBSCRIBER-COS	System,UMTS,SX-LIVPRT,A02-MADRID-1BILD-VF-DE,Vodafone,Prepaid, Rot
MAX-FORWARDS	10
VIA	1.1 rn2wwpsv161-nc1-0.wwp.vodafone.de
CONNECTION	close
REMOTE_ADDR	139.7.146.41

改善方式：

建議不要於服務器上存儲 MSISDN 號碼。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：C.1.7
簡介： 假設他們可以欺騙使用者的電話公司或安全服務本身讓攻擊者進入則實際上不需要使用者的身份認證 token。附加身份驗證雖然不夠完善，但使用雙因素身份驗證會造成攻擊者在技術上障礙遇到很多困擾。
測試步驟： 許多網站上的雙重認證系統透過有人嘗試登錄時使用短信向使用者的手機發送訊息才能執行後續服務。即使使用者在手機上使用專用應用程式來產生代碼，很大的風險使得其他使用者通過向手機發送短信代碼方式登錄或者在確認使用者擁有存取權限後配置為恢復電話號碼的電話號碼時，該服務允許使用者從帳戶中刪除雙因素身份驗證保護。 使用者在丟失手機或獲得新手機後，將現有的手機號碼移動到新的 SIM 卡上，可以通過手機或透過線上瀏覽方式知道使用者經常使用的手機號碼。這時攻擊者可以假裝是本人打電話給使用者的手機公司的客服部門，得知一些關於使用者的個人資訊(例如信用卡號碼，SSN 的後四位數字…等)，攻擊者可以嘗試將使用者的電話號碼轉移到手機上。
範例截圖： 於電話中設定轉接，讓接收到的語音電話轉發到攻擊者的手機，而不會到達使用者的電話。攻擊者可不需要存取使用者的完整電話號碼，攻擊者可以存取使用者的語音郵件，嘗試使用者未使用的情況下於凌晨 3 點登錄網站，從使用者的語音信箱中獲取驗證碼。使用者的手機公司的語音信箱系統並沒有想像中的安全。    使用者的電話號碼可能成為薄弱的環節，允許使用者的攻擊者通過短信或語音電話從帳戶中刪除兩步驗證或者接收兩步驗證碼。當使用者意識到問題時攻擊者已訪問相關內容。
改善方式： 建議行動裝置應用程式中，應採用雙重認證系統降低繞過驗證的風險性。

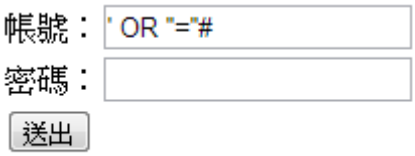
參考來源：OWASP Mobile Security Project Mobile Test cases

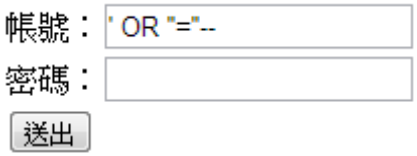
檢測編號：C.1.8
簡介： SSL 證書有效性－用戶端與伺服器 透過 HTTPS 協議訪問 Web 應用程式時，用戶端和伺服器之間建立安全通道。通過數位證書建立一個（伺服器）或雙方（用戶端和伺服器）的身份。因此，一旦確定了密碼套件，“SSL Handshake”將繼續交換證書： 伺服器發送其證書消息，如果需要用刺端認證，還會向用戶端發送 CertificateRequest 訊息，伺服器發送 ServerHelloDone 訊息並等待用戶端響應。在收到 ServerHelloDone 訊息後，用戶端將驗證伺服器的數位證書的有效性。
測試步驟： 為了建立安全通訊，必須對證書進行一些檢查於 SSL 和基於證書的身份驗證，將重點介紹確定證書有效性的主要標準： ● 檢查證書頒發機構（CA）是否為可信任的機構。 ● 檢查證書當前是否有效。 ● 檢查點的名稱和證書中報告的名稱是否匹配。
範例截圖： Ref: https://developer.android.com/training/articles/security-ssl.html
改善方式： 建議取得可信任證書頒發機構（CA）、證書當前有效、相關證書內容名稱符合該證書資訊。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：C.1.9
測試步驟： 檢查以下幾點： ● SQL 注入式：處理動態查詢或 Content-Providers 時，確保使用參數化查詢。 ● JavaScript 注入（XSS）式：驗證是否為任何 WebViews（通常是默認值）禁用 JavaScript 和 Plugin 支援。 ● 本機文件包含：驗證文件系統訪問是否禁用任何 WebViews（webview.getSettings（）。setAllowFileAccess（false）;）。 ● 試圖注入/模糊：驗證操作和資料通過 Intent Filter for All 進行驗證
範例截圖： 以此例說明：

當帳號輸入"' OR ''='' #" 或是 "' OR ''='' -- "(雙引號內字串)就可以進行非法登入[77](如下圖)





原理：在 SQL 指令中 "#"和"-- "代表注釋

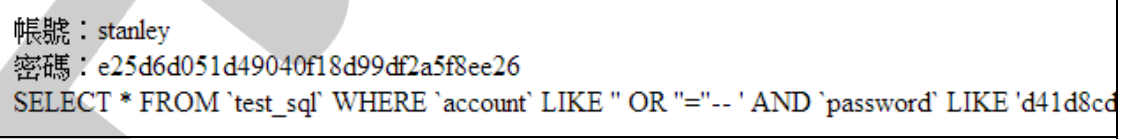
因此原本輸入的查詢是

"SELECT * FROM ‘test_sql‘ WHERE ‘account‘ LIKE '".$account."' AND ‘password‘ LIKE '".$password."'";

會變成 account 欄位等於空 或是 ''='' ，後面的密碼則被注釋掉了

"SELECT * FROM ‘test_sql‘ WHERE ‘account‘ LIKE '".$account."'' OR ''='' ";

而後者條件成立，SQL 指令就會開始執行(如下圖，此例取出第一筆資料)



改善方式：

建議測試人員透過滲透測試方式驗證是否存在相關風險。

檢測編號：C.1.10

測試步驟：

檢查用於避免容易被破解加密的 SSL 配置非常重要，基於 SSL 的服務不應該提供選擇弱密碼套件，密碼套件由加密協議（例如 DES，RC4，AES），加密密鑰長度（例如，40,56 或 128 位）以及用於完整性檢查的散列算法（例如 SHA，MD5）。

密碼套件需要擁有以下幾點：

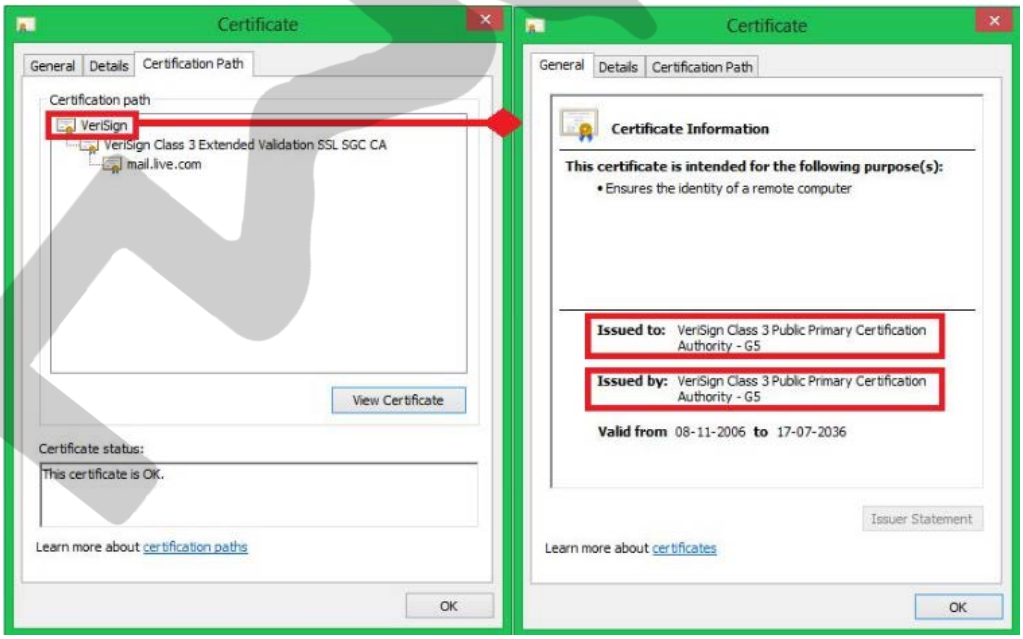
用戶端向伺服器發送一個 ClientHello 消息，其中指定協議能夠處理的密碼套件等訊息。用戶端通常是 Web 瀏覽器（目前最常使用的 SSL 用戶端），但它可以任意支持 SSL 的應用程式。

伺服器使用 ServerHello 訊息進行回應，其中包含將用於會話中所選的協議和密碼套件（通常伺服器選擇用戶端和伺服器支持的最安全的協議和加密套件）。可以（例如，通過配置指令）來指定伺服器遵守的密碼套件，使用這種方式，控制與用戶端的對話是否僅支持 40 位加密。

伺服器發送其證書訊息，如果需要用戶端認證，還會向用戶端發送 CertificateRequest 訊息，伺服器發送 ServerHelloDone 訊息並等待用戶端回應，在收到 ServerHelloDone 訊息後，用戶端將驗證伺服器的數位證書的有效性。

範例截圖：

此為 SSL 證書，需要與網站的伺服器建立一條安全連接進行通訊，避免他人攔截或篡改瀏覽器與伺服器之間傳遞的數據[78]。



改善方式：

建議參閱測試步驟檢查用於避免容易被破解加密的 SSL 配置降低存在風險。

檢測編號：C.1.11

簡介：

必須保護各種類型的訊息於應用程式中以明文形式傳送。透過 HTTP 而不是 HTTPS 來檢查這些訊息或者使用弱密碼，查看更多關於不安全傳輸的訊息 Top 10 2013-A6 敏感資料曝露或傳輸層保護不足 Top 10 2010-A9 傳輸層保護不足。

測試步驟：

例 1：透過 HTTP 進行基本身份驗證

典型的例子是使用 HTTP 的基本身份驗證。使用基本身份驗證時，用戶憑證被編碼而不是加密，並作為 HTTP 傳送。 以下例子中，測試人員使用 curl 來測試此問題。 需要注意應用程式如何使用基本身份驗證 HTTP 而不是 HTTPS

```
$ curl-kis http://example.com/restricted/
HTTP/1.1 401 Authorization Required
Date: Fri, 01 Aug 2013 00:00:00 GMT
WWW-Authenticate: Basic realm="Restricted Area"
Accept-Ranges: bytes Vary:
Accept-Encoding Content-Length: 162
Content-Type: text/html

<html><head><title>401 Authorization Required</title></head>
<body bgcolor=white> <h1>401 Authorization Required</h1>  Invalid login credentials! </body></html>
```

例 2：透過 HTTP 執行表單的身份驗證

另一個例子是透過 HTTP 傳輸用戶認證證書的認證表單。以下例子中，可以看到表單的“action”屬性中使用 HTTP 透過使用攔截代理檢查 HTTP 流量也可以看到這一點。

```
<form action="http://example.com/login">
 <label for="username">User:</label> <input type="text" id="username" name="username" value=""/><br />
 <label for="password">Password:</label>
 <input type="password" id="password" name="password" value=""/>
 <input type="submit" value="Login"/>
</form>
```

例 3：透過 HTTP 發送 Cookie 包含會話 ID

會話 ID Cookie 必須通過受保護的通道傳輸。如果 cookie 沒有設定安全標誌，則會允許應用程式透過未加密方式傳輸，注意下列的 cookie 的設定是沒有安全標誌的，整個登錄進程是在 HTTP 而不是 HTTPS 中執行的。

```
https://secure.example.com/login

POST /login HTTP/1.1
Host: secure.example.com
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0)
Gecko/20100101 Firefox/25.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: https://secure.example.com/
Content-Type: application/x-www-form-urlencoded
Content-Length: 188

HTTP/1.1 302 Found
Date: Tue, 03 Dec 2013 21:18:55 GMT
Server: Apache
Cache-Control: no-store, no-cache, must-revalidate, max-age=0
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Pragma: no-cache
Set-Cookie: JSESSIONID=BD99F321233AF69593EDF52B123B5BDA; expires=Fri,
01-Jan-2014 00:00:00GMT; path=/; domain=example.com; httponly
Location: private/
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN
Content-Length: 0
Keep-Alive: timeout=1, max=100
Connection: Keep-Alive
Content-Type: text/html
-------------------------------------------------------
http://example.com/private

GET /privateHTTP/1.1
Host: example.com
```

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.9; rv:25.0) Gecko/20100101 Firefox/25.0 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Referer: https://secure.example.com/login Cookie: JSESSIONID=BD99F321233AF69593EDF52B123B5BDA; Connection: keep-alive HTTP/1.1 200 OK Cache-Control: no-store Pragma: no-cache Expires: 0 Content-Type: text/html;charset=UTF-8 Content-Length: 730 Date: Tue,25 Dec 2013 00:00:00 GMT
改善方式： 透過測試人員進行網路封包分析檢測敏感訊息是否為明文形式傳輸之外也應注意開發過程中相關敏感訊息以明文方式存在。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：C.1.12

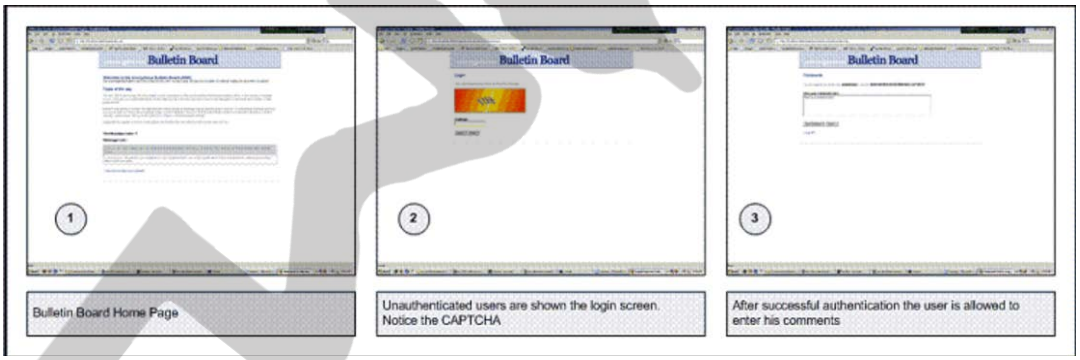
測試步驟：

檢查以下幾點：

1. 產生的人機識別系統有薄弱環節。
2. 用戶端存儲和隱藏字段。
3. 選擇 CAPTCHA 記事本攻擊。
4. 算術 CAPTCHA 和有限集 CAPTCHAs。
5. 選擇 CAPTCHA 標識符攻擊。
6. 可能是固定的 CAPTCHA。
7. 會話中的 CAPTCHA 暴力破解。
8. 識別用戶端到伺服器發送的所有參數（除了解碼的 CAPTCHA 值），以檢查參數是否包含已解碼的 CAPTCHA 和 CAPTCHA ID 號碼的加密或散列值。
9. 測試使用舊的 CAPTCHA ID 發送舊的解碼的 CAPTCHA 值（如果應用程式接受它，則容易遭遇重放攻擊）。
10. 測試使用舊會話 ID 發送舊的解碼 CAPTCHA 值（如果應用程式接受它，則容易遭受重放攻擊）。
11. 找出類似的 CAPTCHA 是否已經被破解了。
12. 驗證 CAPTCHA 的可能答案是否受到限制。

範例截圖：

使用 CAPTCHA 方式保護才能進入[79]。



改善方式：

大多數 CAPTCHA 圖像可在 1-15 秒內解開，因此 CAPTCHA 只能在有限的時間內阻止攻擊者，於安全關鍵應用中更適合使用備用驗證通道（SMS 認證，OTP 等）。

檢測編號：C.1.13
測試步驟： 行動應用程式使用 HTTP 請求輸入來回應，攻擊者可能篡改 HTTP 請求包括 url、querystring、標題、cookie、單域和隱藏字段，嘗試繞過安全機制。必須檢查與會話用戶及活動相關的敏感資訊的 Querystring 參數，根據“正規”規範驗證參數： 1. 資料類型（字符串、整數等）。 2. 允許的字符集。 3. 最小和最大長度。 4. 是否允許 null。 5. 參數是否需要。 6. 是否允許重複。 7. 數值範圍。 8. 具體法律價值（枚舉）。 9. 具體模式（正則表達式）。
範例截圖： 網頁主機： 驗證輸入長度、格式.NET 使用 RegularExpressionValidator，使用正規表示式來驗證使用者輸入的格式[51]。 例如：檢查英數字輸入 6 至 10 位 ValidationExpression="[a-zA-Z]{6,10}"
改善方式： 使用防火牆安全設備可提供參數驗證服務。但必須對設備的每個參數的有效位置進行嚴格的定義，包括正確保護 HTTP 請求中的所有類型的輸入內容，包括 URL、表單、Cookie、查詢字符串、隱藏字段和參數。OWASP 篩選項目是以多種語言生產可重複使用的組件，藉此幫助防止各種形式參數篡改，OWASP 還針對 J2EE 環境開發了 Stinger HTTP 請求驗證引擎（http://sourceforge.net/projects/stinger）。

檢測編號：C.1.14

測試步驟：

檢查 URL 重定向、參數修改攻擊、未經驗證的重定向，在這種情況下可能會出現網絡釣魚攻擊（重定向輸入未被驗證）。

以下 Java 代碼從'url'GET 參數接收 URL，並重定向到該 URL。

```
response.sendRedirect(request.getParameter("url"));
```

類似的 C#.NET 漏洞代碼範例：

```
string url = request.QueryString["url"];
Response.Redirect(url);
```

範例截圖：

Android： 此範例說明如何將加密儲存於 property file 的密碼來和使用者輸入的密碼做安全的比對[51]。

```
public class MainActivity extends ActionBarActivity {
    String flag = "com.ereach.helloworld";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


String encryptPassword = loadPassword(); //因此加密方式為不可逆，如需要比較密碼時，請將使用輸入的密碼加密後比對
// if(encrypt(userInputPassword).equals(encryptPassword))
    }
public static String encrypt(String s){
        MessageDigest sha = null;
        try{
            sha = MessageDigest.getInstance("SHA-256");
            sha.update(s.getBytes());
        }catch(Exception e){
            e.printStackTrace();
            return "";
        }
        return byte2hex(sha.digest());
    }
    private static String byte2hex(byte[] bytes){
```

```
        String hs="";
        String stmp="";
        for (byte myByte: bytes){
            stmp=(java.lang.Integer.toHexString(myByte & 0XFF));
            if (stmp.length()==1) hs=hs+"0"+stmp;
            else hs=hs+stmp;
        }
        return hs.toUpperCase();
    }
 // 讀取資料的方法
private String loadPassword(){ //建構 properties 物件
Properties properties = new Properties();
    try{
        FileInputStream stream = this.openFileInput("music.cfg");
        //讀取兩個檔的內容
        properties.load(stream);
        properties.load(stream1);


    }catch(FileNotFoundException e)
    {
    return;
    }catch(IOException e){
    return;
    }
    return String.valueOf(properties.get("password"));
}
}
```

改善方式：

建議避免使用重定向和轉發與不要將 url 作為目的地的用戶輸入，應該具有驗證 URL 的方法，如果無法避免用戶輸入，請確保提供值的有效性適用於應用程式並授權給用戶。

檢測編號：C.2.1
測試步驟： 使用資料讀取工具對受測軟體在非作業系統保護區內所存放的檔案進行資料讀取。檢查受測軟體是否將明文帳號 密碼存放於非作業系統保護區內。
範例截圖：    進入設定，選擇密碼和表單，點擊管理系統儲存的密碼或者直接在 Chrome 上輸入 Chrome://settings/passwords。
改善方式： 程式開發者應確保所有敏感性資料有採用加密方式進行傳輸，傳輸媒介可包含網路連線、Wifi 連線，甚 至是近場通訊(Near Field Communication, NFC)連線等，若只採用明文方式傳遞，攻擊者可輕易透過網路監聽方式(Sniffer) 竊取機敏性資料。

檢測編號：C.2.2
測試步驟： 執行受測軟體，並儲存敏感性資料。以管理者權限身分使用資料讀取工具，檢查受測軟體是否以明文方式儲存敏感性資料。受測軟體未以明文方式儲存敏感性資料。
範例截圖： 1.選擇加密敏感性資料。 2.有權限使用者才能看到明碼資料。 3.加密敏感資料可使用不同金鑰加密。 4.存取資料即時加解密。 5.強大的加密機制：DES，3-DES，AES。
改善方式： 開發應用程式建議使用強健的加密演算法，並不斷進行反覆測試，直到應用程式開發完成時，仍需執行嚴格的挑戰測試(Battle-Tested)。

參考來源：OWASP Mobile Security Project Mobile Test cases

檢測編號：C.2.3
測試步驟： 使用資料讀取工具對受測軟體執行檔進行資料讀取受測軟體執行檔中是否以明文方式儲存與遠端伺服器溝通之帳號密碼。
範例截圖： Android：此範例為設定如何停用 HttpURLConnection 的 cache[51]。 URL url = new URL(urlString); HttpURLConnection connection = (HttpURLConnection) url.openConnection(); connection.setDefaultUseCaches(false); connection.setUseCaches(false);
改善方式： 建議執行中的敏感資訊如有明文形式應對此資訊提供更完善的保護機制。

檢測編號：C.2.4
簡介： 盡量少用自行撰寫的加密算法，建議使用經過批准的公共算法，如 AES，RSA 公鑰密碼術和 SHA-256 或更好的散列。禁止使用有安全疑慮的加密算法，如 MD5 / SHA1。應選擇更安全如 SHA-256 或更安全的加密演算法。 採用離線生成密鑰，需小心存儲私鑰。禁止在不安全的渠道上傳送私鑰確保基礎架構憑據（如數據庫憑據或 MQ 隊列訪問詳細信息）得到妥善保護（通過嚴格的文件系統權限和控制），或安全加密，並且本機端或遠端用戶不容易解密確保加密資料存儲在硬碟中不容易解密。但資料庫連接如未提供加密的訪問，則資料庫加密是無效的。 根據 PCI 數據安全標準要求 3，對於商戶和處理信用卡的任何人，必須保護持卡人的資料，於 2008 年 PCI DSS 屬於強制性規範。最好方法是禁止存儲不必要的數據，例如硬碟資訊或隱私帳密（PAN，否則稱為信用卡號）。如需存儲 PAN，DSS 規範要求是非常重要的。例如，絕對不能在任何情況下都能存儲 CVV 號碼（卡片後面的三位數字），有關更多信息，請參閱 PCI DSS 指南並根據需要實施控制。
測試步驟： 所有 Web 應用框架都容易發生不安全的加密存儲，防止加密不足。 常見的問題如下列： 1. 敏感訊息未加密。 2. 使用自製的加密算法。 3. 使用不安全的加密算法。 4. 使用有安全疑慮的加密演算法（MD5，SHA-1，RC3，RC4 等）。 5. 使用虛擬碼密鑰，且密鑰存儲在未受保護的存儲中。
範例截圖： Ref: https://source.android.com/security/encryption/full-disk
改善方式： 建議使用經過批准的公共加密算法，並遵循 PCI 數據安全標準要求。

4.3 篩選彙整分析結果

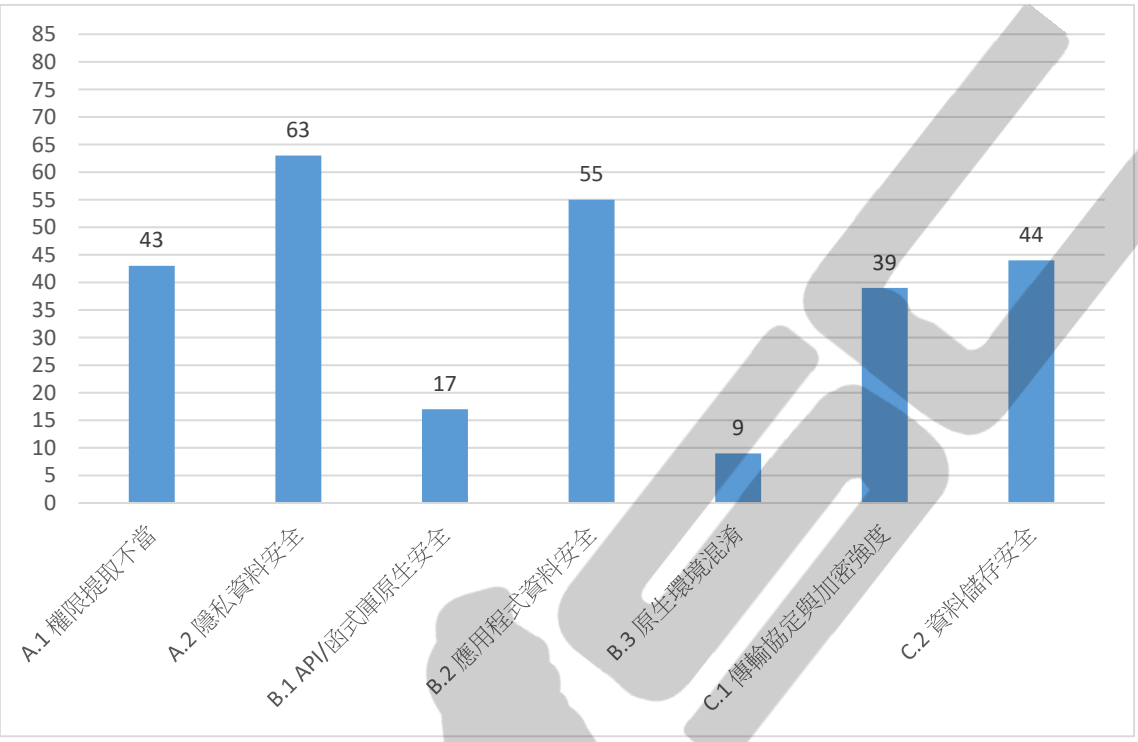


圖 4 彙整分析圖

如附錄二如圖 7 所示，由篩選彙整結果共 85 篇國內論文符合 OWASP 安全議題與 CSA 安全議題，並對應本研究行動裝置應用程式規範檢測項目占的比例圖中隱私資料安全上的相關研究占了 63 篇、應用程式資料安全的相關研究占了 55 篇、資料儲存安全的相關研究占了 44 篇、權限提取不當的相關研究占了 43 篇、傳輸協定與加密強度的相關研究占了 39 篇，結果發現國內學術研究以隱私資料安全、應用程式資料安全為主，而 API/函式庫原生、原生環境混淆相關研究部分相對之下較為薄弱。

4.4 風險評估矩陣

如表 13 所示，依據上續風險評估框架，以 Mobile Top 10 2016-Top 10 作為風險評鑑威脅的分析，將可能造成 CIA 衝擊程度等級分別為風險等級分為低、中、高三級。

表 13 行動裝置風險評估

OWASP Mobile TOP 10	檢測編號	檢測細項	風險等級
M1 平台使用不當	A.1.1	行動應用程式在發布時應完整說明存取敏感性資料、行動裝置資源及宣告權限用途。	低
M4 不安全的認證	A.1.2	行動應用程式存取與個人資料相關敏感性資料應檢查行動應用程式是否提供相關身分授權機制。	低
M6 不安全的授權	A.1.3	行動應用程式應於存取敏感性資料前，取得使用者同意。	低
M6 不安全的授權	A.1.4	行動應用程式未經使用者授權准許使用非相關授權的函數。	中
M6 不安全的授權	A.1.5	行動應用程式經使用者設定拒絕存取敏感性資料後，應用程式不得使用其他方式存取相關敏感資料。	中
M10 多餘的功能	A.1.6	行動應用程式經使用者使用聯絡人或發送接收刪除訊息等功能伺服器必需紀錄，測試者應該嘗試存取另一個用戶的功能，以便驗證是否可以存取不應該被用戶的角色/特權所允許（但可能被允許作為另一個用戶）的功能。	中

OWASP Mobile TOP 10	檢測編號	檢測細項	風險等級
M7 客戶端代碼質量	A.2.1	行動應用程式開發過程中修改密碼執行不當動作所造成的漏洞。	中
M4 不安全的認證	A.2.2	行動應用程式開發過程中是否有註銷功能藉此減少會話劫持攻擊的可能性。	中
M2 不安全的數據存儲	A.2.3	行動應用程式中程式碼或其他封裝之檔案內容，是否存放敏感訊息。	低
M5 加密不足	A.2.4	行動應用程式是否使用密碼強度策略防止隱私資料遭竊取。	中
M10 多餘的功能	A.2.5	行動應用程式有可能造成安全性漏洞功能是否設定為開/關。	中
M2 不安全的數據存儲	A.2.6	行動應用程式是否可在 Rooted / Dev Unlocked / Jailbroken….等環境下存取。	低
M7 客戶端代碼質量	B.1.1	行動應用程式是否存在 SQL 攻擊的風險。	中
M7 客戶端代碼質量	B.1.2	行動應用程式是否具延伸標記語言攻擊字串的風險。	中
M9 逆向工程	B.1.3	行動應用程式是否具逆向工程攻擊的風險。	高
M10 多餘的功能	B.1.4	行動應用程式代碼是否以常量的方式將敏感資訊書寫在原始碼。	高
M4 不安全的認證	B.1.5	行動應用程式是否具 Session Fixation 攻擊的風險。	中
M10 多餘的功能	B.1.6	除錯設定設定值是否安全。	低
M7 客戶端代碼質量	B.1.7	是否具 FTP 協定注入式攻擊。	中

OWASP Mobile TOP 10	檢測編號	檢測細項	風險等級
M4 不安全的認證	B.2.1	行動應用程式在設定時間內不活動時是否具自動關閉應用程式或鎖定功能。	低
M7 客戶端代碼質量	B.2.2	行動應用程式是否具 LDAP 注入攻擊的風險。	中
M7 客戶端代碼質量	B.2.3	行動應用程式是否具 OS 命令注入攻擊的風險。	中
M2 不安全的數據存儲	B.2.4	行動應用程式所產生的敏感訊息是否存在於記憶體傾印中。	低
M8 代碼篡改	B.2.5	行動應用程式透過惡意方式所產生漏洞導致資料安全暴露在危險的風險。	中
M7 客戶端代碼質量	B.3.1	行動應用程式是否具 XSS 攻擊的風險。	中
M8 代碼篡改	B.3.2	行動應用程式是否具惡意檔案上傳的風險。	中
M4 不安全的認證	B.3.3	行動應用程式是否存在 IOS 快照漏洞。	低
M9 逆向工程	B.3.4	行動應用程式透過惡意再包裝與混淆技術造成安全上的風險。	高
M3 不安全的通信	C.1.1	行動應用程式透過網路傳輸敏感性資料時，是否使用加密傳輸以確保敏感性資料安全。	中
M3 不安全的通信	C.1.2	行動應用程式是否具交談識別碼遭重送攻擊的風險。	中

OWASP Mobile TOP 10	檢測編號	檢測細項	風險等級
M3 不安全的通信	C.1.3	行動應用程式與付費功能伺服器間之資料加密傳輸，是否使用安全之加密演算法。	低
M3 不安全的通信	C.1.4	行動應用程式與伺服器間之資料加密傳輸，是否使用安全之加密演算法。	低
M4 不安全的認證	C.1.5	行動應用程式是否具離線狀態繞過認證的風險。	中
M3 不安全的通信	C.1.6	行動應用程式目前狀態是否驗證 MSISDN 號碼。	中
M4 不安全的認證	C.1.7	行動應用程式是否具被繞過二級身份驗證的風險。	低
M2 不安全的數據存儲	C.1.8	行動應用程式內是否清除 SSL Tunnel 中的文件資訊。	低
M7 客戶端代碼質量	C.1.9	行動應用程式是否具被繞過客戶端驗證風險。	中
M3 不安全的通信	C.1.10	行動應用程式中 SSL 憑證/ SSL / TLS 密碼/協議/密鑰無效風險。	中
M2 不安全的數據存儲	C.1.11	行動應用程式中是否具敏感訊息以明文形式暴露的風險。	中
M1 平台使用不當	C.1.12	CAPTCHA 沒有使用至行動應用程式的公共頁面/登錄頁面上。	中
M7 客戶端代碼質量	C.1.13	是否具敏感訊息作為查詢字串參數繞過安全機制風險。	中
M8 代碼篡改	C.1.14	檢查是否具網址參數修改攻擊風險。	中

OWASP Mobile TOP 10	檢測編號	檢測細項	風險等級
M2 不安全的數據存儲	C.2.1	行動應用程式應將帳號密碼儲存於作業系統保護區內或以加密方式儲存。	低
M2 不安全的數據存儲	C.2.2	行動應用程式於儲存敏感性資料時應提供資料加密功能，以避免遭不正當方式取得敏感性資料。	中
M2 不安全的數據存儲	C.2.3	行動應用程式與遠端伺服器溝通之帳號密碼不應以明文方式存在於執行檔中，以避免遭不正當的方式存取。	中
M5 加密不足	C.2.4	應用程式使用加密強度不足。	中

4.5 國內組織規範比較

如附錄三所示，本研究行動裝置應用程式規範項目與目前國內 CISA-行動應用 App 安全開發指引、NCC-手機系統內建軟體資安評估、工業局-行動應用 APP 基本資安規範做比較，從中可發現國內相關規範針對行動裝置應用程式宣告權限用途、使用者同意與拒絕機制、注入式攻擊、敏感資訊儲存、敏感資訊傳輸加密、傳輸加密演算法安全性、敏感訊息明文方式暴露、帳密應儲存於保護區內加密儲存、敏感資料加密、應用程式加密強度不足等檢測項目規範，結果顯示目前國內行動裝置規範都有針對以上項目進行相關規範，但對於特殊情況下存取、逆向工程、相關認證、客戶端驗證等檢測項目規範並不夠完善。

五、結論與未來展望

5.1 研究結論

行動裝置的普及人們漸漸產生依賴性，行動應用程式開發人員對於安全性考慮不足造成使用者個資洩漏的風險，國內相關行動裝置規範仍有不足之處。本研究收集相關行動裝置檢測規範文獻進行分析，依照資料授權儲存安全、傳輸協定安全、應用程式執行安全、系統執行安全歸類文件規範方向，以 OWASP、NIST 為規範基礎結合 CSA 行動裝置規範白皮書，改善行動應用程式檢測規範項目為權限提取不當、隱私資料安全、API/函式庫原生、應用程式資料、原生環境混淆、傳輸協定與加密強度、資料儲存安全檢測項目規範共 46 項檢測細項，依照檢測項目中各個細項提出檢測步驟、改善方法、風險評估給予建議，讓國內行動裝置規範更為完善。

5.2 未來展望

研究內容針對國內相關行動應用程式安全規範進行改善，未來可納入其他國家與研發技術之文獻進行分析，可使行動裝置應用規範檢測規範更具效度，提升國內行動裝置規範整體的安全性。

提出規範後下個階段希望能夠將檢測細項內的各個規範項目實作完成，提供給予開發者與使用者使用，依據檢測項目進行檢測藉此瞭解目前行動裝置應用程式是否存在安全風險給予開發者與使用者提醒與建議。

參考文獻

[1] Matusik,S.F.,Mickel,A.E.,2011,Embracing or embattled by converged mobile devices? User'experiences with a contemporary connectivity technology,Human Relations(64:8),pp. 1001-1030.

[2] 中國漏洞通報平臺烏雲公告，取自 http://www.ithome.com.tw/news/104363

[3] 嚴重安全漏洞，影響的 Android 用戶人數可能高達九億，取自 https://theinitium.com/article/20160810-dailynews-Android-bug/

[4] 行動 App 資訊安全研討會，取自 http://www.mas.org.tw/news_detail.php?id=25

[5]「行動惡意程式數量將成長至 2,000 萬」, PC 花了 21 年才累計達到這個數字，取自 http://blog.trendmicro.com.tw/?p=15589

[6] OWASP Top 10 Proactive Controls 2016，Available from：https://www.owasp.org/images/5/57/OWASP_Proactive_Controls_2.pdf

[7] OWASP 台灣分會介紹，取自 https://www.owasp.org/index.php/Taiwan

[8] OWASP Mobile Security Project，Available from：https://www.owasp.org/index.php/OWASP_Mobile_Security_Project

[9] OWASP Mobile Top Ten 2015 Data Synthesis and Key Trends，Available from：https://www.dropbox.com/sh/d143o6tbkdx4w4l/AAAQlpmnCpHCgiBqZkgXPSTKa?dl=0

[10] OWASP Mobile Security Project - Top Ten Mobile Risks(2014)，Available from：https://www.owasp.org/index.php/Projects/OWASP_Mobile_Security_Project_-_Top_Ten_Mobile_Risks

[11] Mobile Top 10 2016-Top 10，Available from：https://www.owasp.org/index.php/Mobile_Top_10_2016-Top_10

[12] NIST 簡介，取自 https://www.nist.gov/about-nist/our-organization

[13] SP 800-163,Vetting the Security of Mobile Applications，Available from：http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-163.pdf

[14] SP 800-164, DRAFT Guidelines on Hardware-Rooted Security in Mobile Devices，Available from：http://csrc.nist.gov/publications/drafts/800-164/sp800_164_draft.pdf

[15] ITU-T 簡介，取自 http://www.ituchina.cn/itut/about/201506/t20150611_2112509.html

[16] 經濟部工業局簡介，取自 https://www.moeaidb.gov.tw/external/vie w/rwd_tw/intro/index01.html

[17] 中華民國國家通訊傳播委員會簡介，取自 http://www.ncc.gov.tw/chinese/content.aspx?site_content_sn=2255&is_history=0

[18] 雲端安全聯盟簡介，取自 http://www.twcsa.org

[19] ISO / IEC 27001:2013，Available from：https://www.iso.org/obp/ui/#iso:std:iso-iec:27001:ed-2:v1:en

[20] ISO / IEC 27002:2013，Available from：https://www.iso.org/obp/ui/#iso:std:iso-iec:27002:ed-2:v1:en

[21] Government Mobile and Wireless Security Baseline(2013)，Available from：https://cio.gov/wp-content/uploads/downloads/2013/05/Federal-Mobile-Security-Baseline.pdf

[22] SP800-53 Rev 4, Security and Privacy Controls for Federal Information Systems and Organizations(2013)，Available from：http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf

[23] SP-800-124 Rev 1, Guidelines for Managing the Security of Mobile Devices in the Enterprise(2013)，Available from：http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-124r1.pdf

[24] SP 800-142, Practical Combinatorial Testing(2010)，Available from：http://csrc.nist.gov/groups/SNS/acts/documents/SP800-142-101006.pdf

[25] SP 800-164, DRAFT Guidelines on Hardware-Rooted Security in Mobile Devices(2012)，Available from：http://csrc.nist.gov/publications/drafts/800-164/sp800_164_draft.pdf

[26] SP 800-163,Vetting the Security of Mobile Applications (2015)，Available from：http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-163.pdf

[27] Cyber Threats to Mobile Phones(2011)，Available from：https://www.us-cert.gov/sites/default/files/publications/cyber_threats-to_mobile_phones.pdf

[28] Commercial Mobile Device Implementation Plan(2013)，Available from：https://www.defense.gov/news/dodcMdimplementationplan.pdf

[29] Mobile Device Strategy(2012)，Available from：https://www.defense.gov/news/dodmobilitystrategy.pdf

[30] Mobile Operating System Security Requirements Guide Version 1, Release 3 (2013)，Available from：http://iase.disa.mil/stigs/Documents/u_mobile_operating_system_v1r3_srg.zip

[31] Ten Steps to Smartphone Security(2012)，Available from：https://www.fcc.gov/sites/default/files/smartphone_master_document.pdf
[32] Ten Steps to Smartphone Security (Android)(2012)，Available from：https://www.fcc.gov/sites/default/files/12.14%20Mobile%20Security%20Tips%20(Android%20-%20Links)_0.pdf
[33] Ten Steps to Smartphone Security (iOS)(2012)，Available from：https://www.fcc.gov/smartphone-security/Apple%20iOS
[34] Ten Steps to Smartphone Security (Windows Phone)(2012)，Available from：https://www.fcc.gov/smartphone-security/Windows%2BPhone
[35] Ten Steps to Smartphone Security (BlackBerry)(2012)，Available from：https://www.fcc.gov/smartphone-security/BlackBerry
[36] Mobility Security Guide v2.3(2013)，Available from：https://www.nsa.gov/ia/_files/Mobility_Security_Guide.pdf
[37] Security Requirements for Mobile Operating Systems v1.0 (2013)，Available from：https://www.niap-ccevs.org/pp/pp_mobility_os_v1.0.pdf
[38] Smartphone secure development guidelines for app developers(2011)，Available from：http://www.enisa.europa.eu/activities/Resilience-and-CIIP/critical-applications/smartphone-security-1/smartphone-secure-development-guidelines/at_download/fullReport
[39] Top Ten Smartphone Risks(2014)，Available from：https://www.enisa.europa.eu/activities/Resilience-and-CIIP/critical-applications/smartphone-security-1/top-ten-risks
[40] Smartphones：Information security risks, opportunities and recommendations for users(2010)，Available from：https://www.enisa.europa.eu/activities/identity-and-trust/risks-and-data-breaches/smartphones-information-security-risks-opportunities-and-recommendations-for-users/at_download/fullReport
[41] Opinion 02/2013 on apps on smart devices(2013)，Available from：http://ec.europa.eu/justice/data-protection/article-29/documentation/opinion-recommendation/files/2013/wp202_en.pdf

[42] YD/T 2407-2013 移動智慧終端機安全能力技術要求(2013)，取自 http://www.tenaa.com.cn/html/2407-2013%20%E7%A7%BB%E5%8A%A8%E6%99%BA%E8%83%BD%E7%BB%88%E7%AB%AF%E5%AE%89%E5%85%A8%E8%83%BD%E5%8A%9B%E6%8A%80%E6%9C%AF%E8%A6%81%E6%B1%82.pdf

[43] YD/T2408-2013 移動智慧終端機安 全能力測試方法(2013)，取自 http://www.tenaa.com.cn/html/2408-2013%20%E7%A7%BB%E5%8A%A8%E6%99%BA%E8%83%BD%E7%BB%88%E7%AB%AF%E5%AE%89%E5%85%A8%E8%83%BD%E5%8A%9B%E6%B5%8B%E8%AF%95%E6%96%B9%E6%B3%95.pdf

[44] CNS 27000 資訊安全管理系統－概觀及詞彙，取自 http://www.cnsonline.com.tw/?node=detail&generalno=27000&locale=zh_TW

[45] CNS 27001 資訊安全管理系統－要求事項，取自 http://www.cnsonline.com.tw/?node=detail&generalno=27001&locale=zh_TW

[46] CNS 27002 資訊安全管理之作業規範，取自 http://www.cnsonline.com.tw/?node=detail&generalno=27002&locale=zh_TW

[47] CNS 27003 資訊安全管理系統實作指引，取自 http://www.cnsonline.com.tw/?node=detail&generalno=27003&locale=zh_TW

[48] CNS 27004 資訊安全管理－量測，取自 http://www.cnsonline.com.tw/?node=detail&generalno=27004&locale=zh_TW

[49] CNS 27005 資訊安全風險管理，取自 http://www.cnsonline.com.tw/?node=detail&generalno=27005&locale=zh_TW

[50] CNS 27006 資訊安全管理系統稽核與驗證機構要求，取自 http://www.cnsonline.com.tw/?node=detail&generalno=27006&locale=zh_TW

[51] 行動應用 App 安全開發指引草案，取自 http://www.cisanet.org.tw/Services/infoDownload

[52] 經濟部工業局(2016)。行動應用 App 基本資安檢測基準 V2.0

[53] 雲端安全聯盟(2015)。行動 App 安全項目

[54] 國家通訊傳播委員會(2016)。104 年手機系統內建軟體資安檢測報告書

[55] End User Device Strategy：Security Framework & Controls(2013)，Available from：https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/261980/EUD_Security.pdf

[56] Personal information online small business checklist，Available from：https://ico.org.uk/media/for-organisations/documents/1586/personal_information_online_small_business_checklist.pdf
[57] Safer smartphones–keeping your device secure(2013)，Available from：http://consumers.ofcom.org.uk/files/2013/10/mobile-guideV8.pdf
[58] Security Guideline for using Smartphones and Tablets(2011)，Available from：https://www.jssec.org/dl/guidelines2012Enew_v1.0.pdf
[59] 10 Major Security Threats (2013)，Available from：http://www.ipa.go.jp/files/000033747.pdf
[60] Technology Risk Management Guidelines(2013)，Available from：http://www.mas.gov.sg/~/media/MAS/Regulations%20and%20Financial%20Stability/Regulatory%20and%20Supervisory%20Framework/Risk%20Management/TRM%20Guidelines%20%2021%20June%202013.pdf
[61] KCC announces 'Comprehensive Plans for Smart Mobile Security'(2010)，Available from：http://eng.kcc.go.kr/download.do?fileSeq=30248
[62] William Enck,Ongtang, M. and Mcdaniel, P.(2009). Understanding Android Security
[63] 許博學(2013)。探討 Android 系統安全機制
[64] Smith, M. (1989), Computer Security-Threats, Vulnerabilities and Countermeasures,Information Age, October, PP.205-210
[65] ISO 31010:2009 風險管理—風險評鑑技術，取自 https://www.iso.org/standard/51073.html
[66] 行政院國家資通安全會報技術服務中心，取自 http://ir.csu.edu.tw/bitstream/987654321/1677/1/334.pdf
[67] 資訊系統風險評鑑介紹（上），取自 https://icstwebstorage.blob.core.windows.net/attachfileold/20110627.pdf
[68] 資訊系統風險評鑑介紹，取自 http://download.nccst.nat.gov.tw/attachfilehandout/2010091002.pdf
[69] Android 會話劫持安全測試工具，取自 https://tieba.baidu.com/p/1794188718
[70] 表單輸入自動完成，取自 http://polinwei.blogspot.tw/2009/07/form.html
[71] Web 安全注入攻擊，取自 http://www.kangddos.com/4380.html
[72] FTP 注入攻擊，取自 https://read01.com/Jk8PaG.html

[73] TREND LABS 趨勢科技全球技術支援與研發中心，取自 https://blog.trendmicro.com.tw/?p=5528

[74] Trendlabs-SECURITY-INTELLIGENCE Blog，取自 http://blog.trendmicro.com/trendlabs-security-intelligence/new-ghost-push-variants-sport-guard-code-malware-creator-published-over-600-bad-android-apps/

[75] tcpdump 用於擷取網路介面封包，取自 http://blog.xuite.net/jyoutw/xtech/23669726-tcpdump+的用法

[76] 憑證簽章，取自 http://www.netadmin.com.tw/index.aspx

[77] SQL injection 基本介紹，取自 http://newaurora.pixnet.net/blog

[78] 自簽名的 SSL 證書，取自 https://read01.com/oeNzaa.html

[79] 基於 CAPTCHA 授權，取自 http://www.blogjava.net/baoyaer/articles/102593.html

[80] 盧艷銘(2011)。Android 系統上的滲透測試，國立交通大學資訊科學與工程研究所碩士論文，新竹市。

[81] 孫建興(2012)。行動裝置上惡意軟體行為偵測之研究-以 Android 為例，大同大學資訊經營學系碩士論文，台北市。

[82] 陳健宏(2012)。利用行為相似性偵測 Android 平台惡意應用程式，國立交通大學網路工程研究所碩士論文，新竹市。

[83] 曾子建(2012)。基於動態資訊流動追蹤之預防權限欺騙之使用，國立交通大學網路工程研究所碩士論文，新竹市。

[84] 陶嘉仁(2012)。Android 程式權限分析，國立交通大學資訊科學與工程研究所碩士論文，新竹市。

[85] 陳彥宇(2012)。Android 上的隱私資料洩漏偵測，國立交通大學資訊科學與工程研究所碩士論文，新竹市。

[86] 姜家安(2012)。利用權限過濾器及靜態分析之 Android 惡意軟體偵測，國立交通大學電信工程研究所碩士論文，新竹市。

[87] 黃俊龍(2013)。Android 惡意應用程式之檢測使用樹突狀細胞演算法，輔仁大學資訊工程學系碩士班碩士論文，新北市。

[88] 謝維揚(2013)。MalCatcher:以存取以及網路洩漏隱私資料行為為基礎的 Android 惡意程式行為偵測，國立交通大學網路工程研究所碩士論文，新竹市。

[89] 林志剛(2013)。Android 應用程式安全與權限分析，國立宜蘭大學多媒體網路通訊數位學習碩士在職專班碩士論文，宜蘭縣。

[90] 吳柏欣(2013)。密碼管理應用程式之設計與實作，國立臺灣科技大學電機工程系碩士論文，台北市。

[91] 謝文彬(2013)。通訊網路環境中安全身份認證協定之研究，國立臺灣科技大學電子工程系博士論文，台北市。

[92] 詹智宇(2013)。防止越權存取之加密 NFC 標籤機制研究，國立臺灣科技大學電機工程系碩士論文，台北市。

[93] 蔡沅錡(2013)。Android 行動裝置解鎖與實體鑑識系統設計與實現，國立高雄師範大學資訊教育研究所碩士論文，高雄市。

[94] 宋哲光(2013)。利用 Android 系統隱通道攻擊之惡意軟體分析與偵測，國立成功大學電腦與通信工程研究所碩士論文，台南市。

[95] 江玟璟(2013)。基於資訊外洩的行動惡意軟體行為分析，國立中山大學資訊管理學系研究所碩士論文，高雄市。

[96] 高宗永(2013)。預防第三方軟體對使用者過度請求授權之提醒機制，國立臺灣大學電機工程學研究所碩士論文，臺北市。

[97] 蕭舜文(2013)。PasDroid: 在 Android 系統上即時防堵惡意軟體的保護方案，國立臺灣大學資訊工程學研究所碩士論文，臺北市。

[98] 張文銓(2013)。使用機器學習檢測 Android 手機上的惡意程式，銘傳大學資訊工程學系碩士班碩士論文，台北市。

[99] 蘇宏麒(2013)。針對手機特有資訊為輸入點的 Android 應用程式測試平台，國立臺灣大學資訊工程學研究所碩士論文，臺北市。

[100] 張世杰(2013)。Ape: Android 系統惡意程式之自動化測試環境，國立臺灣大學資訊工程學研究所碩士論文，臺北市。

[101] 沈穎志(2013)。強化 Andriod 惡意程式之自動化動態分析機制之有效性，國立臺灣大學資訊工程學研究所碩士論文，臺北市。

[102] 陳韋廷(2014)。Android 應用程式安全性分析，國立臺北科技大學資訊工程系研究所碩士論文，台北市。

[103] 蘇育暄(2014)。Android 應用程式能力分析與潛在權限機制洩漏隱憂之偵測，國立交通大學資訊科學與工程研究所碩士論文，新竹市。

[104] 鄭容沛(2014)。基於使用者意圖追蹤與事件探勘之 Android SMS 木馬惡意程式偵測，國立臺灣科技大學資訊工程系碩士論文，台北市。

[105] 謝宜蓁(2014)。Android 應用程式安全性分析─以雲端資料櫃 App 為例，大葉大學資訊管理學系碩士論文，彰化縣。

[106] 林慶瑋(2014)。以動態分析來分析 Android APP 之 API 特徵，國立雲林科技大學資訊管理系碩士論文，雲林縣。

[107] 陳儀烜(2014)。智慧型 Android 手機 App 之有效的惡意軟體偵測，大葉大學資訊管理學系碩士論文，彰化縣。

[108] 潘佑宣(2014)。Android App 權限管理檢測分析，元智大學資訊管理學系碩士論文，桃園市。

[109] 楊中皇(2014)。Android 裝置上基於 SCAP 的安全系統設計與實現，國立高雄師範大學資訊教育研究所碩士論文，高雄市。

[110] 廖皓翔(2014)。基於 SQLite 的 Android 應用程式之隱私洩漏分析，國立東華大學資訊管理碩士論文，花蓮縣。

[111] 張紘綸(2014)。適用於 Android 應用程式的隱私風險評估機制，國立東華大學資訊管理碩士論文，花蓮縣。

[112] 喬峯(2014)。多因子 Android 惡意程式偵測系統，國立中山大學資訊工程學系研究所碩士論文，高雄市。

[113] 莊欣瑜(2014)。基於機器學習之 Android 惡意程式複合偵測方法，國立臺灣大學電機工程學研究所碩士論文，臺北市。

[114] 潘彥謙(2014)。Android 系統上程序自修改的偵測與保護，國立臺灣大學資訊工程學研究所碩士論文，臺北市。

[115] 邱毓軒(2014)。基於動態載入之 Android App 防複製攻擊機制，國立臺灣科技大學資訊管理系碩士論文，台北市。

[116] 范雋彥(2014)。一套適用於 Android 平台的應用程式風險評估機制，國立臺灣科技大學資訊管理系碩士論文，台北市。

[117] 柯儒(2014)。基於檢查意圖權限之 Android 多層次共謀型惡意程式漏洞偵測，國立臺灣科技大學資訊工程系碩士論文，台北市。

[118] 黃士杰(2014)。在無法確保點對點安全時鑑別他人持有行動裝置之機制，國立臺灣科技大學資訊管理系碩士論文，台北市。

[119] 林廷翰(2014)。基於動態分析之加殼惡意程式偵測系統，國立臺灣科技大學資訊工程系碩士論文，台北市。

[120] 鄭忻忻(2014)。行動應用程式的個人資料保護── 公開透明原則的強化與本國實證研究，國立交通大學科技法律研究所碩士論文，新竹市。

[121] 董才業(2014)。行動資料安全管理之設計原則與實作，國立臺灣大學資訊工程學研究所博士論文，臺北市。

[122] 游繼凱(2014)。為行動裝置設計之智慧型自動鎖定機制，國立臺灣科技大學資訊管理系碩士論文，台北市。

[123] 陳信仁(2014)。利用視覺密碼以預防小額付款詐騙之技術，逢甲大學資訊工程學系碩士論文，台中市。

[124] 邱柏銓(2014)。鏈狀蜂巢模糊區域保護連續查詢位置隱私，中原大學資訊工程研究所碩士論文，桃園市。

[125] 王振達(2014)。動態資料攪亂機制於智慧型行動裝置之研究，國防大學資訊管理學系碩士論文，桃園市。

[126] 林宜盈(2014)。利用基於身份的加密系統預防手機小額付款詐騙，國立政治大學資訊科學學系碩士論文，台北市。

[127] 吳旻訓(2014)。瞭解權限要求對於採用手機 APP 之影響，國立中山大學資訊管理學系研究所碩士論文，高雄市。

[128] 周彥竹(2014)。整體學習之非侵入式手機使用者識別機制，國立中央大學軟體工程研究所碩士論文，桃園市。

[129] 林家瑋(2015)。基於互信息及類神經網路之 Android 惡意程式檢測系統，國立雲林科技大學資訊工程系碩士論文，雲林縣。

[130] 侯凱中(2015)。探討使用者對 APP 應用程式權限認知風險及對使用意願之影響—以 Google Android APP 為例，嶺東科技大學資訊管理系碩士論文，台中市。

[131] 溫立欣(2015)。一項 Android 手機上詐騙簡訊的偵測與防禦機制，國立中央大學資訊工程學系在職專班碩士論文，桃園市。

[132] 許明恩(2015)。Android 錯誤特徵抽取查詢庫，國立臺灣大學電子工程學研究所碩士論文，臺北市。

[133] 張崴(2015)。混合式 Android 應用程式安全機制之研究，國立臺灣大學電機工程學研究所碩士論文，臺北市。

[134] 張宇丞(2015)。攻擊情境之概念及其在 Android 惡意程式偵測之應用，國立臺灣大學電機工程學研究所碩士論文，臺北市。

[135] 呂紹綱(2015)。一個針對第三方 Android 市集所設計的行動裝置應用程式完整性驗證機制，國立臺灣科技大學資訊管理系碩士論文，台北市。

[136] 蔡育軒(2015)。基於資料外洩路徑的 Android 應用程式隱私風險分析方法，國立臺灣科技大學資訊管理系碩士論文，台北市。

[137] 鐘珮珊(2015)。於 Android 應用程式置入風險資訊之方法，國立臺灣科技大學資訊管理系碩士論文，台北市。

[138] 楊芳捷(2015)。基於 Android Pay 的行動支付機制，國立東華大學資訊管理系碩士論文，花蓮縣。

[139] 張瑋玲(2015)。基於行為分析與機器學習的 Android 惡意軟體檢測方法，國立清華大學資訊工程學系碩士論文，新竹市。

[140] 劉冠緯(2015)。Android 惡意程式靜態分析機制之研究，國立高雄第一科技大學資訊管理研究所碩士論文，高雄市。

[141] 歐佳綾(2015)。採用不可否認簽章抵禦 App 複製攻擊，國立臺灣科技大學資訊管理系碩士論文，台北市。

[142] 陳彥伶(2015)。DroidCIA: 基於 HTML5 語法行動應用程式的惡意程式碼注入攻擊偵測，國立臺灣科技大學資訊工程系碩士論文，台北市。

[143] 李昱璇(2015)。Droidivision：基於惡意意圖模擬之多層次共謀攻擊漏洞分析，國立臺灣科技大學資訊工程系碩士論文，台北市。

[144] 蔡卓倫(2015)。行動裝置防肩窺攻擊之身分認證機制，銘傳大學電腦與通訊工程學系碩士論文，台北市。

[145] 陳冠霖(2016)。Android 行動裝置隱私衝擊評估系統設計與實現，國立高雄師範大學軟體工程與管理學系碩士論文，高雄市。

[146] 朱英糧(2016)。Android 手持裝置之 Root 管理工具評測，大葉大學資訊管理學系碩士論文，彰化縣。

[147] 張詠竣(2016)。因應反鑑識攻擊之記憶體萃取分析證據研究，中央警察大學資訊管理研究所碩士論文，桃園市。

[148] 劉子慶(2016)。一個讓使用者協助指定敏感資料的手機應用程式安全分析方法，國立臺灣科技大學資訊管理系碩士論文，台北市。

[149] 陳鴻源(2016)。Android 惡意程式偵測系統，國立臺中科技大學資訊工程系碩士論文，台中市。

[150] 張哲源(2016)。基於許可權與模擬器分析的 Android 惡意軟體偵測系統之設計，銘傳大學資訊工程學系碩士論文，台北市。

[151] 周宏曄(2016)。Android 軟體錯誤線上回報系統，輔仁大學資訊工程學系碩士論文，新北市。

[152] 曾瀚緯(2016)。行動軟體自動化安全評估機制之設計與實作，國立臺灣海洋大學資訊工程學系碩士論文，基隆市。

[153] 林敬軒(2016)。不需伺服器公鑰之電子病歷匿名認證協定，長庚大學電機工程學系碩士論文，桃園市。

[154] 方彥霏(2016)。建構行動裝置數位證據鑑識標準作業程序之研究-從智慧型手機萃取數位證據分析，國立宜蘭大學多媒體網路通訊數位學習碩士論文，宜蘭縣。
[155] 修敏傑(2016)。在行動隱私中評估隱私性和可用性之研究，國立高雄大學資訊管理學系碩士論文，高雄市。
[156] 林修妤(2016)。基於自動化權限分析之 BYOD 安全政策制定研究，國立中央大學資訊管理學系碩士論文，桃園市。
[157] 王俊傑(2016)。DroidDAPA: 偵測廣告潛在攻擊，國立臺灣科技大學資訊工程系碩士論文，台北市。
[158] 張翎翎(2016)。基於使用者行為之防肩窺攻擊行動裝置認證機制，銘傳大學電腦與通訊工程學系碩士論文，台北市。
[159] 陳冠霖(2016)。Android 行動裝置隱私衝擊評估系統設計與實現，國立高雄師範大學軟體工程與管理學系碩士論文，高雄市。
[160] 尤永隆(2017)。具 Root 權限之 Android 惡意軟體的有效檢測技術，大葉大學資訊管理學系碩士論文，彰化縣。
[161] 王正喆(2017)。Android 應用程式安全性分析之研究-以即時通 APP 為例，國防大學資訊管理學系碩士論文，桃園市。
[162] 洪婉娟(2017)。應用多重 GPS 資訊於對稱式加密機制之研究-以 Android 智慧型行動裝置為例，國防大學資訊管理學系碩士論文，桃園市。
[163] 蔡薤竟(2050)。iOS 越獄鑑識與證據分析研究，中央警察大學資訊管理研究所碩士論文，桃園市。
[164] 葉書廷(2050)。iOS 鑑識調查以行動手機於即時通訊之探索研究，中央警察大學資訊管理研究所碩士論文，桃園市。

附錄一 國際組織與各國規範公告之相關技術細節彙整對應表

OWASP Mobile 2014 Top 10 Risk[54]

資料授權 儲存安全	➢ M2：應避免使用 NSUserDefaults 的儲存敏感資訊，例如:避免使用 NSManagedObject 所有資料將儲存在未加密的資料庫文件中。 ➢ M2：機密資訊的儲存或暫存有必要考慮使用一個標準的加密函式庫，如 CommonCrypto。但對於特殊敏感的資訊應用，則可考慮使用白箱加密，以避免數位簽名在通用加密函式庫中被找到而造成洩漏。 ➢ M2：Android 可用“setStorageEncryption”以強制加密本地端文件的儲存，而對於 SD 卡儲存某些安全函式可以透過 javax.crypto 中的函式庫實現。 ➢ M2：儲存敏感訊息時，完全依賴於編碼加密或解密密鑰，應考慮提供操作系統預設加密機制外的附加加密。 ➢ M5：當行動裝置軟體之資料需要從行動裝置匯出時，則該資料需要用使用者金鑰進行加密，以確保僅持正確金鑰者方可存取該資料。
傳輸協定安全	➢ M2：強制用戶使用標準的 Web API 或登錄方案（通過 HTTPS）進行身分驗證。 ➢ M2：應用程式可應用 SSL/TLS 傳輸機敏資訊，網路會談層或其他敏感數據傳輸到後端 API 或 Web 服務通道之加密保護。 ➢ M3：考慮未來安全弱點發生在 SSL 執行發現的情況下，額外提供數據加密將提供資料保密有效的二次防禦。較新的威脅允許攻擊者在資料通過行動裝置 SSL 庫加密之前截取行動裝置內傳輸中的敏感資訊，並竊聽網路流量後發送到目標伺服器。 ➢ M3：使用 SSL/TLS 傳輸時，應避免網路會談層的混雜，進而造成使用者 session ID 的洩露。 ➢ M6：不應使用被證明有明顯弱點的演算法與通訊協定，例如:RC2,MD4,MD5,SHA1。
應用程式 執行安全	➢ M5：確保所有身分驗證請求都需要在伺服器端進行。一旦驗證成功，資料將被載入到行動裝置。以確保資料只在認證成功後才可使用。 ➢ M5：如果需要的數據的客戶端儲存時，資料將需要被安全地從使用者之登錄憑證所得之加密密鑰進行加密。 ➢ M2：資料庫應考慮使用 SQLcipher 進行資料加密。 ➢ M2：對於存儲在 keychain 的項目應利用安全的 API 加以保護。 ➢ M3：使用受信任的 CA 提供商簽署的證書及要求 SSL 驗證程序的完整，避免使用自簽憑證，作為安全應用程序的一部分。 ➢ M3：唯有在確認金鑰使用受信任且有效憑證，並且與伺服器的確認身

	分後才可建立安全連線。 ➢ M3：在 Android 系統上，移除開發期間所有可能允許應用程序接受所有憑證，如 org.apache.http.conn.ssl. AllowAllHostnameVerifier 或 SSLSocketFactory. ALLOW_ALL_HOSTNAME_VERIFIER。如果使用延伸 SSLSocketFactory 的類別，則應確保 checkServer Trusted 方法正確的被實作，使伺服器能有效核對憑證之正確性。 ➢ M3：在 iOS 系統使用 CFNetwork 時，可以考慮使用安全傳輸 API 來指定受信任的客戶端憑證。幾乎在所有情況下，使用 NSStream 通訊連結 SECURITYLEVEL 的 TLSv1 應使用具備更高標準的密碼強度。以確保（NSURL 或包裝）NSURL 通訊不允許自簽或無效憑證，如 NSURL 類方法 setAllowsAnyHTTPSCertificate。 ➢ M10：行動裝置應用程式在編譯時應針對其完整性檢測，須能夠發現程式碼被新增或修改。應用程式必須能在執行時對完整性衝突做出反應。 ➢ M7：避免行動裝置應用程式遭注入威脅，伺服器應該對下列可能的注入威脅，做適當的處理，包含: SQLite Injection, JavaScript Injection, Local File Inclusion, XML Injection, Format String Injection, Classic C Attacks, Intent Injection/Fuzzing。
系統執行安全	➢ M9:行動裝置應該具備適當的超時保護,以避免未經授權的惡意存取。良好的超時週期根據應用程式的安全敏感性、既有的風險狀況等有很大的不同，建議的方針是： ・15 分鐘的高安全性應用。 ・30 分鐘中等安全應用。 ・1 小時低安全應用。 ➢ M5：行動裝置應用程式中之持久性認證功能，用戶密碼不應該在裝置上儲存。 ➢ M5：理想情況下，行動裝置應用程式應使用能夠讓用戶自行撤銷已安裝特定於裝置之憑證，以確保可避免裝置被盜或遺失時，遭到未經授權的存取。 ➢ M8：避免使用不合時宜的 handleOpenURL 方法處理 URL 的呼叫；應採用 openURL: sourceApplication: annotation 這類的方式與透過安全參數驗證受信任的應用程式白名單。 ➢ M6：避免使用行動裝置應用程式內建加密程序，以防止攻擊者透過逆向工程或繞道的方式，取得機敏資料。

參考來源：NCC-手機系統內建軟體資安評估

NIST SP 800-163[54]

資料授權 儲存安全	➢ 啟用已獲得使用者授權功能：應用程式必須描述所有工作、按鈕，選項和其他介面連接所必需的作業並得到使用者確認。 ➢ 避免啟用未授權功能：權限可能透過隱式授予，使應用程式能在未經使用者同意的前提下，擅自使用相關功能。 ➢ 限制權限：應用程式應該只有最小的必要權限，且應避免與使用者授權聲明不一致。需要被加以考量應用程式所獲取權限的如下： • File input/output (I/O) and removable storage • Privileged commands • APIs ➢ 測試應用程式更新：應用程式的新版本都必須經過測試，以確定任何可能產生的新弱點。 ➢ 避免應用程式共謀：避免應用程式可能透過彼此之間的資訊交換，而獲得原本未被使用者授權資料之取得。 保護敏感資料：應導入被證明足夠強健的加密機制(參考經 FIPS 140-2 驗證核可之加密機制)，以保護所蒐集/儲存/傳輸敏感資料的機密性和完整性。在隱私處理時也需要被考慮如位置數據、車載攝影照片，及設備身分的應用程式等，也需要被加以保護避免洩露。
傳輸協定安全	➢ 未加密通訊：當應用程式之間進行內部通訊，則將可能讓應用程式收集額外資訊，並注入惡意資訊。而當使用在對外通訊時(Datanetwork, Wi-Fi, Bluetooth, NFC, etc.)則可能招致中間人攻擊之威脅。
應用程式 執行安全	無
系統執行安全	無

參考來源：NCC-手機系統內建軟體資安評估

NIST SP 800-164[54]

資料授權 儲存安全	➢ 受保護的儲存裝置：保護儲存在設備上資料的機密性和完整性，並當使用中的資料發生未授權的應用程式 試圖存取被保護的項目時，應撤銷其存取權限。 ➢ 避免當行動裝置遭竊而導致機敏資料洩露，需要有足夠強健的 Data at Rest (DAR)機制，可參考下列標準： ・NIST SP 800-111 敘述有關環境風險和注意事項兩大類設備的資訊、儲存加密技術為終端使用者設備參考指南。 ・FIPS-140 敘述有關 DAR 加密保護機制之標準，並保證使用標準化和普遍接受的機制進行加密操作，以及完成獨立的驗證。 ・ FIPS-199 為了有效的應用基於策略的受保護的儲存媒體，資訊的所有者需要明確定義出資料的保護需求和政策。其中資訊將被進行機密等級分類，並依據使用分類的標準作為處理的依據。 ・NIST SP800-60 用以協助對應資訊安全類別，關於醫療資訊、財務或個人資訊之保護，則可進一步參考 NIST SP800-122。 ➢ 加密機制強健性由加密演算法與密鑰長度所共構，參考下列標準： ・ NIST SP800-133 密鑰的生成的準則。 ・ NIST SP800-90A 建議透過使用確定性隨機變數產生器，來產生具有足夠熵值的隨機變數以提升安全強度。 ・ NIST SP800-132 建議提供密鑰導出函數，並可用於推導從密碼密鑰對資料加密之密鑰或密鑰的保護的準則。 ・ ISO / IEC11889 提供重點服務保護加載密鑰用於確保機密性和完整性。此服務可以實現為一個純粹基於軟體服務或與硬體支援的可信平台模組。 ・ NIST SP800-88rev1，提供媒體清除原則，確立了消除數據的三種方法，以提供不同級別的保護。
傳輸協定安全	➢ 透過建立身分，產生連線、維護、共享、蒐集、宣告及處理六個方面已完成整個管理的生命週期。 ➢ 發出服務請求的一方，應對遠端的裝置進行評估，並參與設備上的軟體所使用的通訊協定，以保證機密性、完整性及簽章的正確性，並完成設備驗證。
應用程式 執行安全	無
系統執行安全	隔離並防止在同一設備上的應用程式和資訊意外交互存取，參考 NIST SP800-88rev1 媒體清除原則，以提供不同級別的保護。

參考來源：NCC-手機系統內建軟體資安評估

資料授權 儲存安全	➢ 4.6.3 用戶資料的加密儲存：未經授權的軟體無法存取加密儲存區之電話簿資料。 ➢ 4.6.5.1 用戶資料遠端鎖定：受測之行動裝置經鎖定後，在尚未解鎖前，行動裝置持有者無法進行操作。 ➢ 4.6.5.2 用戶資料遠端刪除：受測之行動裝置資料經刪除後，用戶資料無法再被讀取。 ➢ 4.6.6.1 用戶資料的本地備份：透過備份功能進行使用者電話簿、簡訊、多媒體簡訊、郵件與多媒體資料之備份，並產生相關提示。
傳輸協定安全	➢ 4.4.1.1.1 藍牙接口開啟/關閉之開關。 ➢ 4.4.1.1.2 藍牙接口開啟之受控機制。 ➢ 4.4.1.1.3 NFC 接口開啟/關閉之開關。 ➢ 4.4.1.1.4 NFC 接口開啟之受控機制。 ➢ 4.4.1.2 藍牙配對連接的受控機制。 ➢ 4.4.1.3.1 藍牙接口連接狀態顯示。 ➢ 4.4.1.3.2 NFC 接口連接提示。 ➢ 4.4.1.4.1 藍牙接口數據傳輸受控機制。 ➢ 4.4.1.4.2 NFC 接口數據傳輸受控機制。
應用程式 執行安全	無
系統執行安全	➢ 4.6.1.1 開機密碼保護：行動裝置開啟身分認證功能，以確保唯有持正確密碼方可令行動裝置開機。 ➢ 4.6.1.2 開機後鎖定狀態的密碼保護：行動裝置開啟身分認證功能，以確保唯有持正確密碼方可使用行動裝置。

參考來源：NCC-手機系統內建軟體資安評估

附錄二 回顧文獻樣本描述與對應本研究規範項目

編碼	題目	研究發現	研究方法	對應本研究規範項目
1	Android 系統上的滲透測試[80]	有心人士散佈惡意軟體，竊取使用者個人隱私資料，或是擅自使用付費服務，造成使用者金錢損失。Android Market 並沒有對上載的應用程式提供嚴格的審查機制，加上 Android 手機允許安裝非 Market 上的應用程式，這些都會讓使用者容易下載到惡意軟體仍渾然不覺。	透過滲透測試系統蒐集各種 Android 系統上的漏洞，並提供檢測和相應的建議，並透過 Wi-Fi 攻擊途徑來實現這些檢測。	A.2 隱私資料安全 B.2 應用程式資料安全
2	行動裝置上惡意軟體行為偵測之研究以 Android 為例[81]	智慧型手機上儲存著豐富的個人隱私資訊，各家資安公司所發現的手機惡意軟體中，竊取個人隱私資料的木馬程式所占比率最高，伴隨而來的智慧型行動裝置資通訊安全議題。	針對惡意軟體必須對外連線，將竊取資料傳送至特定主機的異常網路行為特徵，提出一套行動裝置惡意軟體行為檢測的作法，並實際架設一個無線網路監測環境，透過實驗方式，以 Android 手機為例，比對惡意軟體與正常軟體網路行為異常特徵，驗證其可偵測出手機感染惡意軟體之異常網路行為。	A.2 隱私資料安全 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
3	利用行為相似性偵測Android平台惡意應用程式[82]	Android系統平台，攻擊者透過再包裝與混淆的技術，將惡意程式碼隱藏到多個看似一般的應用程式來進行散佈。然而，被打包惡意程式碼的應用程式即使有了不同的外表，但同樣的惡意程式碼仍然會產生出同樣的行為。	利用系統呼叫序列來進行應用程式的行為偵測方法，能夠從多執行緒的惡意程式所產生的系統呼叫序列中找出共同子序列，且利用貝氏機率模型來過濾出有較高機率出現在惡意應用程式，但較低機率在正常應用程式執行時出現的系統呼叫序列。	B.1 API/函式庫原生安全 B.3 原生環境混淆
4	基於動態資訊流動追蹤之預防權限欺騙之使用[83]	由於Android程式間頻繁的合作因素，導致程式間元件交流的安全性變得重要。權限欺騙之使用可以間接取得它原本所沒擁有的權限為了去從事一些惡意的行為。弱點分析、基於Manifest的存取控制以及程式間通訊的監控等之前的研究並沒有辦法完全地解決權限欺騙之使用的問題，因為他們皆缺乏辨別呼叫串之源頭的能力。	藉著動態資訊流動追蹤能夠辨識intent的來源，可以執行精細的存取控制。具備此辨識能力，就可以干涉程式元件通訊的過程並確保每一個通訊的過程不會違反強制存取控制的機制去完全解決權限欺騙之使用的問題。	A.1 權限提取不當
5	Android程式權限分析[84]	Google Play在app的安全檢查機制並沒有相當完善，所以使用者很容易就會暴露在資訊安全的危機當中。如果使用者沒有細心注意app的所有內容，很可能就會被存取到個人隱私資	實作出一個系統，從蒐集的資料建立資料庫，並且提供一個可以方便使用的app來幫助使用者檢測app所用到的手機資源，提供給使用者建議，幫助使用者判斷app的安全性。	A.1 權限提取不當 A.2 隱私資料安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
		料，洩漏給惡意的第三方。		
6	Android 上的隱私資料洩漏偵測[85]	智慧型手機、平板電腦科技蓬勃發展，擁有強大的功能，也讓這些移動裝置成為新的攻擊目標最常見的攻擊是資料竊取與外流。	提出 LeakDet，一個可安裝在 Android 行動裝置上，可偵測隱私資料洩漏的系統。LeakDet 能夠偵測封包內是否含有隱私資料，並且阻擋該封包流向的 IP。將最新版的入侵偵測系統(IDS)軟體 Snort 移植到 Android 手機上，並搭配 Snortsam 以及 Linux Kernel 內建的防火牆 Iptables 來達到入侵防禦系統(IPS)的功能。	A.2 隱私資料安全 B.2 應用程式資料安全
7	利用權限過濾器及靜態分析之 Android 惡意軟體偵測[86]	現在智慧型手機的功能越來越全面，例如:網路銀行、NFC、GPS 等，都使得攻擊者有更多攻擊的模式。因為在 Google Market 上傳應用程式沒有限制所以惡意軟體的數量也隨之增加。	提出使用 Android 應用程式的權限先判斷應用程式，以減少使用靜態分析的頻率，若判斷結果是可疑的應用程式，再以靜態分析來偵測。	A.1 權限提取不當
8	Android 惡意應用程式之檢測使用樹突狀細胞演算法[87]	Android 系統允許用戶在系統下安裝其他的應用程式。這些軟體中，可能存在惡意應用程式可以執行有害的行為，例如電話費用詐騙、竊取簡訊內容及其他惡意行為。	每個應用程式執行時，記錄其系統呼叫(system call)使用樹突狀細胞演算法對收集資料進行良性或惡意應用程式的分類，將收集數據輸入到 DCA 做異常檢測，透過 DCA 方法與決策樹、貝氏分類及支持向量機器比較。	A.2 隱私資料安全 B.2 應用程式資料安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
9	MalCatcher: 以存取以及網路洩漏隱私資料行為為基礎的 Android 惡意程式行為偵測[88]	Android 上的惡意程式數量長大速度非常地快速，在現今防毒軟體所使用的 signature-based static analysis 方法已經無法跟上惡意程式的演化速度。	提出動態分析惡意程式行為方法，主要修改 Android 系統原始碼並重新編譯，運行在模擬器上建立一獨立且受控制的環境供 APP，並將一網路封包監控軟體 snort 重新交叉編譯為 Android 系統可執行之版本運行在模擬器上來監控網路封包是否有洩漏使用者隱私資料。	A.2 隱私資料安全 B.1 API/函式庫原生安全
10	Android 應用程式安全與權限分析[89]	由於 Android 開放源始碼的特性，使得投入應用程式的開發者眾多，上傳到應用程式市集 Google Play 的應用程式又沒有一套很嚴謹的審機制，導致 Android 應用程式的素質是良莠不齊，一般使用者只能被動的選擇評分高或風評好的開發者下載。但除了在 Google Play 下載之外，很多人也會到第三方市集或不知名網站下載應用程式，在這種情況下很容易就下載到劣質，甚至是危害系統，洩露個人資料的惡意軟體。	提出一套篩選機制對應用程式在安裝於裝置之前做權限及安全性的分析，接著透過事先在 Android 模擬器上運行並觀察是否有異常行為，之後再安裝到實體裝置上，來降低額外的風險以及避免浪費不必要的金錢與時間。實際驗證 Google 雲端硬碟的安全性，確實也在 FTP 協議中發現明碼傳送的帳號密碼。	A.1 權限提取不當 A.2 隱私資料安全 C.1 傳輸協定與加密強度
11	密碼管理應用程式之設	目前市面上的密碼管理軟體大多沒有對暴力破解法做防範且系	在 iOS 平台上開發一套應用於 iPhone 的密碼管理系統來輔助使用者管	A.1 權限提取不當 C.1 傳輸協定與加密強度

編碼	題目	研究發現	研究方法	對應本研究規範項目
	計與實作[90]	統欄位格式皆為固定，導致系統可能有被破解密碼的疑慮和使用者受到系統格式限制而不能任意輸入資料的缺點。	理紀錄其密碼資訊，日後使用者將只需記得一組密碼即可。透過系統的仿真虛擬資料也可防止他人暴力破解出真實密碼。	
12	通訊網路環境中安全身份認證協定之研究[91]	相互認證確保網路上訊息發送者與接受者的合法性，允許被授權者存取有限的資源，也保護服務提供者不被非法者攻擊。而根據安全強度、通訊成本及計算量的高低，不同的終端設備採用的密碼元件也有所不同。譬如在行動通訊上，在相同安全強度下，基於橢圓曲線密碼系統的相互認證機制更有效率、更節省資源。	針對兩種網路環境無線存取網路及無線感測網路，分別提出適用之安全認證協定。無線存取網路中，利用橢圓曲線密碼系統快速加解密的優點提出了個適用於行動裝置上的使用者匿名認證機制，改善先前相關文獻的計算效率，並藉由修改運算的步驟，避免先前參考文獻可能遭遇到的惡意攻擊，如阻斷服務攻擊及使用者身份追蹤等。在感測器網路環境中，提出基於單向雜湊函數上之認證協議。由於所提出的方法利用到單向雜湊函數及互斥或的運算，因此與其他感測器網路上認證協議比較，更加地實用且有效率。	C.1 傳輸協定與加密強度 C.2 資料儲存安全
13	防止越權存取之加密NFC標籤機制研究[92]	NFC 技術被廣泛的應用在門禁管理、票券服務、電子付款、裝置配對等應用上。隨著NFC 應用的普及和使用上的需求，安全性的議題便成為了重要	提出防止越權存取的加密機制，透過對稱式加密演算法來保護 NFC Tag 中的個人資料，並藉由非對稱式加密演算法的特性達到金鑰交換的目的，藉此擴大 NFC	A.1 權限提取不當 C.1 傳輸協定與加密強度

編碼	題目	研究發現	研究方法	對應本研究規範項目
		的考量。尤其是針對於沒有處理器能夠執行加密機制來建立安全通道的 Passive Tag 而言，目前都沒有很完善的機制保護使用者的資料不被竊取。	Tag 應用的範疇。由具有 NFC 功能的 Android 智慧型手機實作加解密功能，將 NFC Tag 上所載有的明文資料透過加密演算法處理後，再將加密後的密文寫回 NFC Tag 中。若有惡意的攻擊者想使用非法的讀取裝置對 NFC 進行越權存取時，將會得到無法辨識的資訊，藉以保護使用者的個人資料安全。	
14	Android 行動裝置解鎖與實體鑑識系統設計與實現[93]	為確保用戶個人的隱私安全性，Android 搭載了 Screen Lock 的基本安全保護措施(包含 Pattern Lock、PIN Lock、Password Lock)，但這些功能也有可能被當作一種犯罪的保護媒介使用，因此產生了如何在保護裝置鎖定的狀態下完成數位證據採集並解除保護以完成其他的鑑識作業的需求。	依據美國國家標準技術局手機鑑識流程，針對 Android 智慧型手機在 Screen Lock 保護的狀態下進行實體資料採集及磁碟映像製作並解除保護狀態，蒐證出的資料及映像檔可透過 PC 端進行分析及刪除資料的還原，最後解除 Android 的螢幕保護鎖定讓鑑識人員進行邏輯資料的採集或其他鑑識作業。	A.2 隱私資料安全 B.2 應用程式資料安全
15	利用 Android 系統隱通道攻擊之惡意軟體分析與偵測[94]	Covert Channel 的存在，為 Android 不同 APP 間資料訊息傳遞，提供隱匿不受監控的方式。惡意軟體可以利用螢幕亮度、手機音量，以及 External Storage 等等	以竊聽智慧型手機使用者隱私為主的實驗測試軟體，其結合 VoIP 與 Covert Channel 技術並做為實驗分析的對象。並提出了一項避免智慧型手機電子感應元件遭到惡意軟體利用後，透	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
		媒介來達成 Covert Channel 資料傳遞的手段。若沒有機制對此種威脅做適當的處理，則 Malware 可以利用合作的方式降低單獨個體所需的資料權限，隱匿的傳送資料，對使用者的隱私及資料安全。	過 Android 本身的隱通道洩密而對使用者的隱私造成傷害的解決方案。	
16	基於資訊外洩的行動惡意軟體行為分析[95]	越來越多人使用行動裝置進行各項活動，其中儲存的大量使用者資料，駭客們針對其開發並製作惡意軟體，這些惡意軟體以行動裝置上使用者資料為目標，造成使用者的資料外洩，被用來謀取利益，對使用者造成極大的損失。	提出用敏感性資料洩露為特徵來偵測惡意程式的方法，利用封閉環境的模擬裝置來隔離並執行應用程式，透過行為分析的方式，監控並記錄應用程式執行過程中的行為，並對其相關行為進行深入分析，從檔案與網路連線等方面，來分析並追縱敏感性資料是否被送出行動裝置外面，同時也針對應用程式發送短訊息的行為是否異常等方式，來偵測惡意軟體。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全
17	預防第三方軟體對使用者過度請求授權之提醒機制[96]	惡意軟體會破解系統權限、濫用加值服務、大量發送廣告訊息、竊取使用者的各類資訊,對於注重個人隱私。在 android 系統中,應用程式要執行特定功能都必須要在安裝的時 候先取得相對應的授權。因此,透過	整合 android 應用程式和網路服務的系統架構,利用推薦演算法-協同過濾產生的推薦指數來達成提醒機制。當 使用者覺得推薦程度低的時候可以檢視列出的權限表,重新考慮是否 要安裝。最後,以使用者的決定與推薦分數相近的	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
		確認應用程式請求的權限可以避免許多的惡意軟體。	程度來驗證我們系 統的功能性。	
18	PasDroid: 在Android 系統上即時防堵惡意軟體的保護方案[97]	目前的 Android 系統無法讓使用者知道應用程式何時存取用戶的私密資料再來，由於 Android 缺少審核機制，使得惡意軟體正爆炸姓成長，而這些惡意軟體可能竊取使用者的私密資料。	利用 PasDroid 可以降低使用者私密資料被竊取的風險，並且可以讓使用者自行定義哪些檔案是私密資料並且持續追蹤這些資料。PasDroid 提供白名單機制讓使用者控制應用程式允許送出的私密資料類型。當有未經授權的私密資料被傳送出去前，PasDroid 會阻止這筆資料的傳送並且跳出警告視窗通知使用者。	B.2 應用程式資料安全 C.1 傳輸協定與加密強度
19	使用機器學習檢測Android 手機上的惡意程式[98]	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，再進行如惡意消費、手機資源消耗、協助犯罪，或者竊取資料。	研究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。	A.1 權限提取不當 A.2 隱私資料安全 B.1 API/函式庫原生安全 B.2 應用程式資料安全
20	針對手機特有資訊為輸入點的Android 應用	大量資訊儲存於手機上,私密資訊的管理與存取的安全性 也逐漸浮現。為了讓應用	針對此件作為測試輸入點,嘗試輸入非預期性的資料針對應用程式進行攻擊。為了加速測試	A.1 權限提取不當 B.2 應用程式資料安全 B.3 原生環境混淆

編碼	題目	研究發現	研究方法	對應本研究規範項目
	程式測試平台[99]	程式間能分享手機中的各式資訊,Google 定義了一種 Android 元件來管理資訊,並透過權限來控管其他應用程式能否存取此資訊,但 Google 卻未驗證其寫入的資訊合法性與正確性。	流程,開發 Android 應用程式分析工具,透過分析獲得的攻擊路徑以及讀取資料後程式執行流程,從中取出具有淺在危險的應用程式,對其製作可能的攻擊模式來檢測其應用程式。	
21	Ape: Android 系統惡意程式之自動化測試環境[100]	Google 開發的 Android 作業系統提供了一套基於權限的安全機制來限制應用程式無法隨意存取使用者的私密資料，然而這套機制卻不夠縝密導致許多惡意程式仍然可以逃過此機制。	結合了聰明的事件產生器與動態分析工具的自動化偵測環境 Ape，是一套能夠自動偵測 Android 應用程式是否會洩漏敏感資料的服務。利用 Ape，使用者事先檢查任何一個從第三方網站下載的應用程式，並得到一份分析報告包含資料是否洩漏以及一個特定的 Activity Call Graph， 供使用者做進一步的分析。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全
22	強化 Andriod 惡意程式之自動化動態分析機制之有效性[101]	惡意程式具有隱匿行蹤，在無使用者點擊觸發或是等待特定條件的情況下啟動惡意行為，例如透過等待監聽系統廣播事件的方式。研究提出自動化動態分析惡意程式架構，可偵測機敏性資料外洩行為，運用智慧型事件觸發機制，結合使用者介面事件觸發器來探索並	此架構建置於 TaintDroid 基礎上，透過監控應用程式對機敏性資訊相關應用程式介面之使用及是否有發送簡訊或撥打電話的行為，來偵測機敏性資料外洩或金錢不當得利之惡意行為。運用反偵測模擬器的技術來防止惡意程式透過偵測是否執行於虛擬化環境的技術來隱匿其惡意行為。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度

編碼	題目	研究發現	研究方法	對應本研究規範項目
		啟動圖形介面活動元件，系統事件觸發器模擬系統廣播事件來自動化地揭露。		
23	Android 應用程式安全性分析[102]	智慧型手機裝置與生活越來越密切，使得駭客針對惡意程式來竊取使用者的隱私資料，造成使用者的資料外洩或是用來謀取利益，對使用者造成極大的損失。	透過靜態分析偵測應用程式中的權限、暴露的風險與惡意程式進行相似度比對，再利用 Sandbox 來模擬 Android 行動裝置並執行應用程式，同時對執行的所有行為進行監控及記錄，將兩者得到的資料進行深入分析，並輸出成報表提供給使用者觀看，幫助使用者判斷應用程式的安全性。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全
24	Android 應用程式能力分析與潛在權限機制洩漏隱憂之偵測[103]	Android 提供以權限為基礎的機制限制應用程式之間與系統資源之間的存取。應用程式必需請求特定的權限來取得對受保護資源的存取。但現存的 Android 權限機制中仍然存在著某些潛在的漏洞及隱憂。	提出了一個以 client-server 為架構的系統來進行應用程式的能力分析,並利用圖形資料庫中的 cypher query language 來支援潛在漏洞的偵測。	A.1 權限提取不當 B.2 應用程式資料安全
25	基於使用者意圖追蹤與事件探勘之 Android SMS 木馬惡意程式偵測[104]	Android 系統相關 App 提供人們在行動通訊上的便利性，但也因為結合傳統手機功能，惡意程式可以在使用者不知道的情況下，擅自寄送簡訊訂閱服務，造成使用者財務損失。	SMSDroidCare，利用反向工程取出程式 API Call 使用資訊，追蹤 API Call 使用狀況，來進行分析，再配合事前分析 SMS 木馬惡意程式擅自發送 SMS 簡訊收費服務訂閱的行為所制定的事件序列類型，符合	A.1 權限提取不當 B.1 API/函式庫原生安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
			產生的事件序列透過頻繁情節探勘出最頻繁的情節法則模式。	
26	Android 應用程式安全性分析—以雲端資料櫃 App 為例[105]	Android 應用程式發展迅速，人們得以透過行動裝置及雲端相關 App 經由網際網路達到雲端儲存服務功能。則智慧型手機應用雲端服務 App 已漸漸成為使用者在任何時間、地點進入雲端的入口，而人們使用雲端儲存服務也已成為一種新趨勢，但也伴隨著資訊安全議題。	封包擷取方法蒐集雲端儲存服務伺服器與使用者端資料傳送的封包，藉由封包分析嘗試瞭解 CloudBox App 安全性的弱點。	C.1 傳輸協定與加密強度
27	以動態分析來分析 Android APP 之 API 特徵[106]	越來越多人把隱私資料存入手機中，隱私資料不只限於個人資料，也包含手機的感應器收集的資訊。大多數人對於智慧型手機的惡意程式敏感度較低。	動態分析 Android 惡意 App 的 API 呼叫方法，利用 APIMonitor 調出被隱藏 API，在模擬器上建立獨立且受控制的環境供 App 導入執行。	A.1 權限提取不當 A.2 隱私資料安全 B.1 API/函式庫原生安全 B.2 應用程式資料安全
28	智慧型 Android 手機 App 之有效的惡意軟體偵測[107]	Android 軟體開發及 Google Play Market 的開放式平台，沒有嚴謹的核准制度。雖然 Android Market 有了一種全新的安全性機制 Bouncer 保護，但駭客卻仍然有辦法躲過偵測，盜取受感染的智慧型行動裝置使用者資料。	針對網路傳輸通訊的過程進行惡意軟體偵測，採用動靜態混合偵測法偵測出未知的 Android App 惡意軟體，以有效防範因智慧型行動裝置網路通訊傳輸而洩漏個資或機密資料的威脅。	A.2 隱私資料安全 C.1 傳輸協定與加密強度

編碼	題目	研究發現	研究方法	對應本研究規範項目
29	Android App 權限管理檢測分析[108]	Android 對於程式的管控僅安裝時供程式相關權限表列，讓使用者決定是否安裝，達成對程式的權限管制。對於不完全了解權限表列意義的使用者，容易跳過檢視權限表列的動作而直接安裝應用程式。Android SDK 對於 Permission 的說明文件不夠，使得許多非惡意的應用程式要求過多的權限的現象。	應用程式權限管理系統協助使用者追蹤應用程式存取私人資訊的情形，同時也可讓使用者設定應用程式的存取權限以及使用權。主要包括 APK 重新封裝伺服器，讓使用者可以上傳/下載 APK,以及相對應的客戶端程式 AMCG，可以與重新封裝後的 APK 溝通,做管制 APK 的存取行為。	A.1 權限提取不當 B.3 原生環境混淆
30	Android 裝置上基於 SCAP 的安全系統設計與實現[109]	大部份行動裝置攻擊都是透過系統的弱點跟漏洞，而手機內的個人資料安全和隱私保護更是刻不容緩。	以 Android SDK 與 Java 語言設計 Android APP 程式掃瞄裝置,利用第三方軟體套件 jOVAL 的離線掃描功能，在網頁上以報表方式分析結果及評分，並且提供使用者設備配置建議，以確保手機資訊的安全。	A.2 隱私資料安全 B.2 應用程式資料安全
31	基於 SQLite 的 Android 應用程式之隱私洩漏分析[110]	越來越多個人敏感性資料皆被有意無意儲存在 Android 系統之中，因此使得駭客有較高的機會來拼湊出個人(或組織)的隱私資料。	Android 系統上的隱私分析框架,為 AppLeak,用以對 Android 系統上的應用程式進行資訊損失評估、隱私洩漏檢測和隱私風險評估。透過 SQL 指令和內容檢索機制針對特定應用程式所對應之 SQLite 資料庫進行敏感資訊的挖掘，此外，AppLeak 亦採用	A.2 隱私資料安全 B.1 API/函式庫原生安全 B.2 應用程式資料安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
			了新的分析觀點：使用者感知和客觀攻擊意識，來針對行動應用程式執行的資料流進行隱私分析。	
32	適用於Android應用程式的隱私風險評估機制[111]	開放式的 Android 系統平台上，各種新型態的隱私詐騙手法與預測潛在隱私機制層出不窮，對於使用者資料的保護已造成重大的威脅。	使用一種新提出的資訊與隱私洩漏操作定義，藉此有效率的分析出應用程式的隱私風險，此外針對使用者對本身敏感資訊的主觀認知與惡意攻擊者端的客觀認知進行敏感資訊的制定。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.2 資料儲存安全
33	多因子Android惡意程式偵測系統[112]	Android 系統的開放性以及使用者會將原本需要付費的 App 重新封裝後供他人下載，使用者可輕易的在自己的智慧型手機上安裝第三方市集上所下載App,但由於第三方市集上 App 毋須經過官方認證，因此出現惡意 App 機率較高。	側錄了 App 執行時、閒置時所使用到的 System Call 以及 App 所要求權限，接著使用資料探勘的技術來比較官方市集與已知惡意 App 之紀錄差異，再使用機器學習之技術來建造偵測模型。	A.1 權限提取不當 A.2 隱私資料安全 B.1 API/函式庫原生安全 B.3 原生環境混淆 C.1 傳輸協定與加密強度 C.2 資料儲存安全
34	基於機器學習之Android惡意程式複合偵測方法[113]	Android 平台上的惡意程式偵測為當前重要且熱門的研究議題。	使用靜態分析的方式，逆向工程取得應用程式的 Android API 使用情形，歸納出惡意行為以及正常行為的特性，並結合機器學習的方法-支持向量機，從現有的資料中分別學習獲得在惡意行為及正常行為上的分類模型。	B.1 API/函式庫原生安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
35	Android 系統上程序自修改的偵測與保護[114]	惡意程式嘗試使用各種包裝來隱藏自己以迴避偵測，Android 系統上的程序自修改正是一個用來隱藏自己代碼的新技術。	提出偵測方法來協助偵測這種類型的惡意程式,並根據偵測的結果發展了一套保護機制來避免執行到修改後代碼的風險。	B.2 應用程式資料安全
36	基於動態載入之 Android App 防複製攻擊機制[115]	現行的 App 保護機制無法有效保護 App 開發者的權益。	動態載搭配數位版權管理的概念，設計 App 防複製攻擊機制。機制中，使用者執行 App 時，需通過身分鑑別獲得分離程式區段，方可執行 App 的所有功能。本機制亦可達到匿名性、不可偽造性、雙向鑑別、抵抗中間人攻及抵抗重送攻擊等安全需求。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密加強 C.2 資料儲存安全
37	一套適用於 Android 平台的應用程式風險評估機制[116]	使用者常將隱私敏感資訊常駐於手持裝置中，為確保個人隱私的保障與安全性，並在使用社群服務、線上交易的場合中能夠在隱私洩漏風險及便利間取得平衡，因此需要一個有效且能夠量化隱私洩漏程度的模型。	量測資訊洩漏的方法並引入使用者感知作為評估隱私洩漏的參數，實際以一個雛形應用程式來驗證此方法風險計算模型。同時提出評估隱私風險的框架，在使用者運行應用程式時的流程中所蒐集到的使用者輸入資訊，來評估風險層級。	A.1 權限體取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密加強 C.2 資料儲存安全
38	基於檢查意圖權限之 Android 多層次共謀型惡意程式漏洞偵測[117]	Android 系統允許安裝第三方手機應用程式，在 Android 中,主要是透過內部元件通訊當作通訊機制，手機應用程式若是不當的使用 ICC,則可能遭受到越權攻擊。	提出一個工具叫做 MLC Tracker 檢查意圖物件特權，識別功能洩漏與代理洩漏以預防多層次共謀型攻擊。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
39	在無法確保點對點安全時鑑別他人持有行動裝置之機制[118]	採用較強的認證方式，透過具有非對稱式金鑰，做到點對點的安全，以便達到交互認證。但系統資源的限制與便利性的考量，無法完全做到點對點的安全，使用者需要識別到府服務的人員所攜帶裝置，是否真的為合格的裝置。	鑑別他人的行動裝置的機制，讓使用者在使用前，可以自行決定是否鑑別這些行動應用服務，是真正有經過服務提供者的授權，藉以避免惡意的行動應用服務竊取使用者的機敏資訊。	A.1 權限提取不當 A.2 隱私資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
40	基於動態分析之加殼惡意程式偵測系統[119]	加殼是一種軟體壓縮加密技術，在軟體工程中很普遍被使用。軟體工程師用此技術保護其開發之軟體與壓縮檔案大小，避免軟體被逆向工程的方法破解或修改。然而，近幾年來加殼卻常常成為在駭客撰寫惡意程式時，用來混淆防毒軟體偵測的趨勢。	藉由逆向工程及動態分析，辨識惡意軟體檔案是否加殼。目的是想要補足靜態分析方法其可能分析失誤之處。特點在於擷取組合語言指令順序的獨特性，擷取出檔案反組譯後將 EntryPoint 程式執行點開始起的第 1 行至第 15 行之組合語言碼，做為訓練集特徵，再用相同方式擷取出測試檔案。	B.1 API/函式庫原生安全
41	行動應用程式的個人資料保護—公開透明原則的強化與本國實證研究[120]	行動應用程式的功能日趨多元，上下游供應鏈多重的個資蒐集處理業者，使得可能存在的個人隱私侵犯愈發複雜與嚴重。以行動應用程式的產業運作實況切入，從個人資料保護的目的與種類出發，分析行動	藉由比較法分析途徑，針對歐盟與美國主管機關、以及國際組織在 2013 年先後發表的行動應用程式隱私保護意見書與規範建議，進行政策的比較研究。針對現今各國通用的「通知和同意機制」在實際運作上所面臨的難題，以及我國個人資料保護法	A.1 權限提取不當 A.2 隱私資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
		應用程式對個資影響的深度和廣度。	在行動應用程式議題上所面臨的困境，逐一剖析，並研擬因應對策與解決方案。	
42	行動資料安全管理之設計原則與實作[121]	定義並探討了嶄新的資料存在問題（Data Presence Problem），涵蓋資料聚合、資料散布，與資料長時間存在的子問題，並提出了使用者控制、使用者端加密，與短暫性等三項顯著功能作為系統設計原則，分別解決上述三項子問題。	同步與非同步潘朵拉安全訊息協定；非同步潘朵拉訊息協定是密碼學前向安全性質的非同步訊息加密協定，整合每條消息可更新加解密臨時金鑰的機制，用來將訊息解密的臨時金鑰，會在訊息傳送者所設定的過期條件滿足時被安全地刪除，這機制讓訊息無法再取得避免資料外洩的訊息傳送者而言，尤其重要。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
43	為行動裝置設計之智慧型自動鎖定機制[122]	行動裝置作業系統設置自動鎖定機制，在手機休眠後會自動鎖定螢幕，必須輸入圖形鎖、PIN 碼或密碼才能解鎖並兼顧安全性與便利性，增設自動鎖定倒數設定。	行動裝置設計智慧型自動鎖定機制，根據使用者的過去習慣來動態調整自動鎖定倒數時間，並同時在安全性與便利性中找出最佳的平衡點。	A.2 隱私資料安全 B.2 應用程式資料安全
44	利用視覺密碼以預防小額付款詐騙之技術[123]	智慧型手機內下載並安裝被惡意者植入隱含有簡訊攔截程式碼的惡意 APP，在程式分析簡訊文字內容後攔截 OTP 回傳至惡意者的接收端，進而完成交易驗證，造成使用者莫名的金錢損失。	解決攔截簡訊 OTP 的問題，以視覺密碼 VSS 的方式來取代傳統的純文字 OTP，達到 OTP 傳送時的安全性。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
45	鏈狀蜂巢模糊區域保護連續查詢位置隱私[124]	適地性服務內建定位功能智慧型手機讓使用者於移動中使用該服務找尋鄰近所需興趣點，但使用 LBS 服務時卻須將自己的所在位置與 POI 傳送給 LBS 伺服器，使的 LBS 服務商藉由收集、分析資料而探知 LBS 使用者的個人習慣與位置隱私。	完全相容於使用者的移動連續 LBS 查詢保護方法結合實際地圖的道路產生涵蓋使用者真實位置的鏈狀模糊區域，以避免 LBS 伺服器過濾掉不合理的移動範圍以滿足使用者所提出的隱私強度。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
46	動態資料攪亂機制於智慧型行動裝置之研究[125]	使用者將個人文件儲存於智慧型行動裝置內，但使用者的智慧型行動裝置遺失、遭竊或被駭，會造成個人資訊外洩問題。為了解決上述問題，資料加密是保護個人文件或防止資訊外洩最好的保護措施。若能運用某些特定因子加強文件的加密程序，可強化智慧型行動裝置內文件的機密性。	以虛擬亂數及 GPS 定位資料運用於對稱式加密之機制。此機制係基於藉由使用者輸入各項密碼參數與智慧型行動裝置 GPS 定位資料來執行資料加解密程序。加密文件只有當使用者以正確的密碼參數與他(她)位於正確的位置才能夠被正確地解密出來。	A.2 隱私資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
47	利用基於身份的加密系統預防手機小額付款詐騙[126]	小額付款為電信業者預設開通功能，小額付款的美意主要是為了取代小額的現金交易，增加民眾的便利性，有網路罪犯利用惡意 app 誘導用戶下載開始擷取用戶的手機資訊，攔截小額付	利用基於身份的加密系統將簡訊加密，使用者收到簡訊時，必須使用向電信公司所申請的「簡訊解密」APP 才能解密讀取簡訊，即使攻擊者攔截到簡訊，也會因為沒有解密金鑰而無法順利讀取簡訊。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
		款的通知簡訊，造成財務上的損失。		
48	瞭解權限要求對於採用手機 APP 之影響[127]	Android 平台中 App 為了使用者各式各樣的功能必須各種取得權限，但 App 也有可能要求為了提供功能外的權限，使用者必須先同意 App 的權限要求才能下載 App。	了解權限對於使用者採用 App 的影響，提出了權限-功能適配度的概念，並以科技接受模型為基礎結合認知隱私風險及社會交換理論來做深入的探討。	A.1 權限提取不當
49	整體學習之非侵入式手機使用者識別機制[128]	手機上儲存了許多重要的個人隱私，使手機安全機制越來越被重視。傳統的手機鎖上，都是一次驗證的機制，透過讀取手機使用者的行為作不中斷的識別使用者。	透過手機應用程式能持續收集使用者操作資訊，以觸碰螢幕與方位感測器當作識別使用者特徵資訊，依現有的相關文獻所遭遇到的問題提出改善並強化其識別效果。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
50	基於互信息及類神經網路之 Android 惡意程式檢測系統[129]	Android 作業系統在行動裝置上的市占率最高，但其安全機制卻不甚完善，導致作業系統的安全問題比其他作業系統嚴重。	使用 Apktool 對運行在 Android 上執行檔(.apk 檔)作反編譯，完成特徵萃取，再以 API call 作為 apk 檔特徵，並結合類神經網路的分類功能，進而對運行於 Android 作業系統上的惡意程式進行檢測。	B.1 API/函式庫原生安全
51	探討使用者對 APP 應用程式權限認知風險及對使用意願之影響 — 以 Google Android APP 為例[130]	使用者在安裝 APP 前，對 APP 所要求的使用權限宣告畫面，是否有足夠知識判斷其權限合理性及相關的知覺風險，進而影響後續的安裝使用意願。	以 Android 系統的智慧行動裝置使用者為對象，針對產品權限知識、權限知覺信任及權限知覺風險等理論為基礎進行探討，然後設計建構本研究模型。	A.1 權限提取不當

編碼	題目	研究發現	研究方法	對應本研究規範項目
52	一項 Android 手機上詐騙簡訊的偵測與防禦機制[131]	密集的惡意程式入侵 Android 智慧型手機，綁架受害者手機的簡訊服務，攔截和自動傳送簡訊認證，進行小額付款購買遊戲點數的詐騙事件。	RSDroid 偵測與防禦機制，修改自 Android 的系統框架(Framework)層級的核心概念，簡訊傳入手機時會偵測簡訊被 Abort 和 Delete 事件，以及防禦沒有經由使用者輸入號碼與內容的簡訊被送出，偵測與防禦惡意程式攔截與自動發送小額付款的認證簡訊。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
53	Android 錯誤特徵抽取查詢庫[132]	Android Apps 經常在未經過完整測試的情況下就先公開上架,供使用者下載。造成這個情況的主要原因有兩個:第一,測試工程師普遍使用手動測試;第二,因為產品更新的週期縮短,使得測試工程師沒有最新的待測程式規格書。	Android 錯誤類型抽取查詢庫,提升測試案例中真正導致待測物故障的比例,依照不同類型的錯誤進行分析搭配資料探勘、數據分析方法,找出待測物在執行時的錯誤特徵,最後提供使用者可導致應用程式故障的測試案例。	A.2 隱私資料安全 B.1 API/函式庫原生安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
54	混合式 Android 應用程式安全機制之研究[133]	程式設計師利用 WebView 這一個元件載入 HTML5 的網頁並且利用 addJavavscriptInterface 這一個 API 註冊 WebView 與原生語言的溝通管道，然而這些溝通管道有可能發生安全性的危害。惡意網頁可能會被 WebView 載入並且利	提出 framework 來保護溝通管道。framework 包含兩個部分，第一部分利用 fined-grained access control 防止惡意網頁存取這個溝通管道，第二部分利用機器學習偵測溝通管道使用是否正常。	A.2 隱私資料安全 B.1 API/函式庫原生安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
		用這些溝通管道攻擊手機。		
55	攻擊情境之概念及其在 Android 惡意程式偵測之應用[134]	攻擊情境自惡意程式中學習及選擇並且以 AndroidAPI 來描述，藉此表示 Android 惡意程式特性。	由於攻擊情境幾乎不產生偽陽性的特徵，使其適合作為機器學習方法的前過濾器，來提升在偽陽性率低情況下的惡意程式偵測率，搭配不同的機器學習方法提升偵測率上的效果。	B.1 API/函式庫原生安全
56	一個針對第三方 Android 市集所設計的行動裝置應用程式完整性驗證機制[135]	Google Play Store 之外，存在不少的第三方 app 市集,這些第三方市集缺乏官方的基本檢測機制，當第三方市集使用者在使用非經由官方認證的 app 時,無意間下載到重新包裝過的 app,可能會出現金錢上的損失等等資安問題。	app 指紋產生機制並蒐集 Google Play 與第三方市集等來源的 app 樣本資料庫，以此資料庫之白/黑名單為基礎,建立具備在上傳 app 時會立即對 app 進行安全性檢測機制的第三方市集，在檢測後會依據白/黑名單判定結果決定此 app 所屬的類別判斷該 app 是否可以上架。	B.3 原生環境混淆
57	基於資料外洩路徑的 Android 應用程式隱私風險分析方法[136]	Android 平台提供權限控管機制，當要進行特定操作的應用程式，安裝時必須得到使用者進行這些操作的許可。Android 的許可模型雖然可讓使用者決定應用程式是否可存取通訊錄、簡訊等資料，卻無法讓使用者知道應用程式將敏感資料透過網路或其他方式傳送到使用者不信賴的地點。	使用動態分析的方式找出手機應用程式之資料流向，以視覺化介面呈現其風險程度，以便 Android 智慧型手機應用程式的分析人員，能夠更方便的掌握到應用程式的風險，而能採取相對應的措施去因應這些風險。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
58	於Android應用程式置入風險資訊之方法[137]	Android 平台要求應用程式必須取得使用者的許可才可存取機敏資料或是特定操作。大部分市集站沒有強制要求應用程式開發者一定要提供隱私政策，即便有提供隱私政策，使用者有時也不能確認隱私政策的正確性。	智慧型手機應用程式中嵌入個資使用方式。該政策經過驗證單位的背書,確保符合實際狀況。使用者在安裝檢測客戶端程式後，在應用程式安裝與更新時，去取得存於應用程式中個資使用方法，並通報使用者取得同意。	A.1 權限提取不當
59	基於 Android Pay 的行動支付機制[138]	行動支付的應用服務普及在人們的日常生活中，這種方便又快速的支付模式下，存在惡意使用者盜刷及盜用風險，潛藏著許多安全的疑慮。	Android Pay 系統上的安全性進行探討與研究,提出可強化 Android Pay 的行動支付機制，採用了橢圓曲線加密系統來保障交易之安全性及效率性。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
60	基於行為分析與機器學習的Android惡意軟體檢測方法[139]	Android 系統的普及吸引有興趣的開發者，自行撰寫不同功能且有創意的應用程式，也可以設計出一些惡意軟體, 偽裝成一般應用程式，但卻在背後執行惡意行為。	改良Droidbox 的不足，加入自訂可辨識應用程式當下執行 UI 介面自動觸擊程式，紀錄背後產生的行為並結合網路行為,讀寫順序等等。透過宣告的權限作為判斷輔助利用 machine learning 去判斷是否為惡意軟體。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
61	Android 惡意程式靜態分析機制之研究[140]	Android 普及與開放式的生態環境帶來惡意程式的大幅成長。成千上萬種的應用程式中，使用者該如何確定所使用的應用程式是否為惡意程式，儼然已成為每個人所	整合式的靜態分析機制,使用 Androguard 反編譯 APK 中各種檔案，比較正常與惡意程式，並依據當中的差異來尋找惡意或可疑的屬性。並進一步運用支援向量機來協助辨識，分	A.1 權限提取不當 B.1 API/函式庫原生安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
		面臨的重大問題之一。	別針對大量且類型一致的應用程式進行分析。	
62	採用不可否認簽章抵禦App複製攻擊[141]	基於動態載入之Android App防複製攻擊機制無法有效鑑別使用者身分的正確性，即無法抵禦授權使用者分享其個人相關鑑別資訊，以供非授權使用者通過身分鑑別，進而取得執行App所需的資源檔。	基於橢圓曲線密碼系統、不可否認簽章及簽密法，設計出抵禦App複製攻擊之機制，透過不可否認簽章於驗證簽章有效性時須透過簽署者合作的特性，確保僅有授權使用者得以通過身分鑑別取得資源檔執行完整功能。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
63	DroidCIA: 基於HTML5語法行動應用程式的惡意程式碼注入攻擊偵測[142]	HTML5開發出的app同時繼承了跨網站指令碼的攻擊，像在一般網站中的攻擊一樣。攻擊者可以從各種不同的通道來注入惡意的指令碼，例如：QR Code, Wi-Fi儲存點...等。	此研究方法可以檢測出過去已發現的注入口，且可以檢測到text box ”document.getElementById(”TagID”).value”新的注入管道。這個新的text box注入管道在過去的研究當中並沒有被發現過，因為過去的研究當中只有分析JavaScript APIs，但卻忽略了包含text box管道資訊的HTML檔案的資訊。	A.2 隱私資料安全 B.1 API/函式庫原生安全 B.2 應用程式資料安全 B.3 原生環境混淆 C.1 傳輸協定與加密強度 C.2 資料儲存安全
64	Droidivision：基於惡意意圖模擬之多層次共謀攻擊漏洞分析[143]	Android防禦為permission機制，手機程式之間不當的使用ICC溝通，造成越權的問題，這種未授權某些敏感權限的手機程式透過觸發其他已授權該敏感權限的手機元件進行溝通與傳遞	提供多層次共謀攻擊漏洞檢測方法 - Droidivision，分析存在使用者手機中的軟體是否存在可能的共謀行為的弱點。不同於檢測手機的溝通和權限，藉由建構出惡意程式中的API關係架構以模擬出	A.1 權限提取不當 A.2 隱私資料安全 B.1 API/函式庫原生安全 B.2 應用程式資料安全 B.3 原生環境混淆 C.1 傳輸協定與加密強度 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
		資料的行為。近年來，共謀攻擊更是越權問題中的一部分而現在的防禦軟體很難檢測出單一共謀的軟體的惡意行為。	可能的惡意共謀行為，並檢測使用者手機中存在多層次共謀應用程式組合與惡意目的。	
65	行動裝置防肩窺攻擊之身分認證機制[144]	智慧型手機中，系統常用的身分認證存有肩窺攻擊，像是個人身分識別碼、安卓系統螢幕圖形鎖，皆是攻擊者直接觀看便能取得使用者之密碼。	智慧型手機觸控螢幕之壓力感測器擷取使用者之壓力值，融入現有常用之身分認證機制中，讓攻擊者觀察或記錄驗證過程無法得知使用者之密碼。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 B.3 原生環境混淆 C.1 傳輸協定與加密強度 C.2 資料儲存安全
66	Android 行動裝置隱私衝擊評估系統設計與實現[145]	行動裝置存放了許多的個人資料，例如：簡訊、照片、通訊錄、銀行憑證、帳號密碼等等。因此，個人資料的安全隱憂格外重要。	評估 Android 作業系統和應用程式對個人資料的影響以及降低行動裝置所帶來的隱私風險，參照隱私衝擊評估的概念，分為三個部份研究發現許多應用程式要求過多且可疑的權限，且應用程式首次安裝時，確認對話框上顯示的權限清單並不完整。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.2 資料儲存安全
67	Android 手持裝置之 Root 管理工具評測[146]	Android 平台有多種安全機制保護系統運作，但駭客卻仍然有辦法躲過偵測，盜取受感染的智慧型行動裝置系統資源。因此，Android Root 權限控管就顯得格外重要。	設計多個惡意 App 程式，其內含各種新型之盜取 Android Root 權限的惡意軟體，用以評測與比較各種 Android 手機 Root 權限管理工具之安全性，以找出較佳之 Root 權限管理工具，協助強化 Android 作業系統核心安全，防止惡意的 User Space 應用程	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
			式企圖存取或破壞系統重要資源。	
68	因應反鑑識攻擊之記憶體萃取分析證據研究[147]	數位證據遭受反鑑識後，相關資料難以復原，因傳統的數位鑑識著重在硬碟的取證分析，對於可揮發性記憶體的即時取證容易忽視，導致關鍵的數位證據因不慎而流失。	探討資料竊取、檔案加密及網頁隱私瀏覽之反鑑識手法，透過記憶體鑑識工具，萃取電腦與手機中的可揮發性記憶體，分析記憶體中的片段資訊，找出經反鑑識後的關鍵數位證據，透過情境案例研究進行分析與討論，提供鑑識調查人員面對反鑑識攻擊時之因應參考對策。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
69	一個讓使用者協助指定敏感資料的手機應用程式安全分析方法[148]	運用動態與靜態分析方法，發展出惡意程式的偵測工具，去識別惡意程式。目前分析方法中，都是針對既有系統 API 可取得的個人資料進行分析，並沒有能針對應用程式要求使用者從使用者介面輸入的資料。	提出 Android 應用程式安全分析平台－SensiDroid，結合使用者協同分析之方式，讓使用者能夠指定應用程式中會取得使用者敏感資料的資料輸入欄位，並將該欄位資訊提供至 SensiDroid，透過追蹤資料流之方法，針對使用者指定會輸入敏感資料的使用者輸入欄位進行分析，以發現誤用的情況。	A.2 隱私資料安全 B.1 API/函式庫原生安全 B.2 應用程式資料安全 C.2 資料儲存安全
70	Android 惡意程式偵測系統[149]	Android 系統允許使用者自行安裝來自非官方市場的應用程式，且應用程式反編譯和修改並不困難。使用一般的防毒軟體來掃描應用程式，通常僅能偵測到目前病	透過擷取大量的惡意程式與良性程式的樣本檔案，掃描並記錄兩者的要求權限和使用權限清單之特徵，採用機器學習技術的 LibSVM，讓系統分類未知的應用程式。	A.1 權限提取不當

編碼	題目	研究發現	研究方法	對應本研究規範項目
		毒碼已知的種類，對於未知的新型變種，通常便無偵測的能力。		
71	基於許可權與模擬器分析的 Android 惡意軟體偵測系統之設計[150]	科技不停的在進步但使用者的觀念卻沒有跟上，許多人在使用手機時會自行安裝第三方所提供的軟體，而造成各種危害。趨勢科技防毒公司的報告中指出以 Android 系統為目標的惡意程式數量已經遠遠超出其他手機系統。	提供機械學習系統採用手機應用程式的靜態特徵與動態特徵。靜態特徵方面擷取手機應用程式的許可權、原生許可權、函式及優先權來分析特徵。動態特徵方面修改系統檔案，藉以取得更加完整的紀錄檔做分析，並將手機應用程式安裝在架設的沙盒系統內進行行為分析。透過分析操作應用程式後所產生的紀錄檔知道該應用程式是否有不正常的行為。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.2 資料儲存安全
72	Android 軟體錯誤線上回報系統[151]	Android 系統是基於 Linux 核心的開放式軟體平台和作業系統，開發廠商可以在平台上開發多種客製化功能，在測試階段易於衍生出各式各樣的軟體錯誤問題。	建構一套 Android 軟體錯誤線上回報系統，在行動裝置的軟體開發階段將裝置提供給模擬終端使用者直接操作。一旦發生嚴重錯誤問題，藉由裝置端的服務程式擷取並分析所需資料，透過網路自動傳送完整的錯誤報告至系統後端伺服器。研發人員可以透過錯誤報告進一步分析及處理問題，藉此提早解決使用者操作可能發生實際軟體問題，提	A.2 隱私資料安全 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
			升軟體品質及降低成本。	
73	行動軟體自動化安全評估機制之設計與實作[152]	行動裝置儲存大量個人隱私資訊、隨時連接網路、鮮少關機但資訊呈現不易。上述特性使得行動裝置軟體的安全問題不易被使用者察覺，也更突顯其軟體安全的重要性。	針對 Google Android 平台，參考 Android lint 的弱點檢測項目以研發行動裝置軟體品質檢測所需之核心技術。了解行動裝置軟體之設計及行為，並預警軟體可能潛藏的風險漏洞及效能議題。利用靜態的技術進行檢測並嘗試避免軟體內潛藏可能被利用的漏洞，進而提升整體系統的安全。	A.2 隱私資料安全 B.2 應用程式資料安全 C.2 資料儲存安全
74	不需伺服器公鑰之電子病歷匿名認證協定[153]	基於驗證子的三方認證計畫提出一種具高效能、高安全性、低運算與少量傳輸的安全性機制，研究中發現，攻擊者在認證系統傳輸過程中可以輕易盜取用戶身分，且前人提出方法的操作方式在 Server 認證身分時因無法有效的找到要認證的資訊，使該系統亦無法實現於實作中。	提出了一個新的方案，可以成功阻止攻擊者的攻擊，並能搭載手機實作於 Android 系統中。此方法在效能、運算傳輸上，都有比原方案更優異的表現此方案更適合運用在遠端醫療資訊交換系統上，對於其安全性，將有很大的助益。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
75	建構行動裝置數位證據鑑識標準作業程序之研究-從智慧型手機萃取數	智慧型手機存在大量的電磁記錄，這些記錄是具備鑑識價值的數位證據。數位證據的蒐集、分析、萃取過程，必須使用標準的數位鑑識流程，以提	依據國內學者林宜隆教授所提出的數位證據鑑識標準作業程序 DEFSOP，建構行動裝置數位證據鑑識標準作業程序雛型 DEFSOP。透過 ISO 27037：2012	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
	位證據分析[154]	高其公信力及有效性。	與ISO 27041：2015的分析比對，驗證DEFSOP的嚴謹度及可用度，透過實例驗證DEFSOP For Mobile Device 的完整性與有效性。	
76	在行動隱私中評估隱私性和可用性之研究[155]	個人資料大量的被行動裝置所蒐集，分享個人資料也可能會造成使用者隱私性資料洩漏風險的增加。行動語境的隱私保護已逐漸受到使用者的重視。但目前的行動隱私保護技術多強調可為使用者帶來較高的隱私保護效果，但卻缺少針對行動語境資料的隱私性和可用性之評估及權衡。	提出一個隱私性增加和可用性遺失的定義及計算的方式，以幫助使用者在隱私性和可用性之間取得一權衡。在實驗分析中，在幾種不同的行動隱私保護技術上，分析、探討提出隱私權衡特性，並提供一分析架構作為使用者在選擇行動隱私保護技術時之參考。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
77	基於自動化權限分析之BYOD安全政策制定研究[156]	BYOD 為企業節省採購成本的同時也面臨資訊安全風險，因此制訂與實施BYOD安全政策顯得格外重要。值得注意的是能夠帶來危害的並不只有惡意程式，合法的應用程序也可能暴露隱密資訊。	提出一個「自動化BYOD安全政策制定」的平台，當員工安裝了一個新的應用程式時，平台將自動偵測員工新安裝的應用程式是否為惡意程式，對於合法的應用程式也會分析其宣告的權限是否會對企業帶來安全風險，將以上分析的結果以安全政策來表示，提供給企業做為制定安全政策的參考。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
78	DroidDAPA: 偵測廣告潛在攻擊[157]	Android App 存在惡意攻擊的實際案例。Android App 惡意攻擊手法已經滲透到廣告，最常見利用廣告引導使用者到惡意網站造成個人或企業財務損失，像網站勒索攻擊或釣魚網站等,因為 Ads 元件採用 Ad-Network 動態產生方式呈現，在偵測上非常困難。	利用影像處理技術和分析 Android 系統方式,探討 Android App Ads 觸發後與 Browser 之間資料傳遞。最後產生惡意網站分析報表。此外亦可幫助 VirusTotal 偵測 Malicious URL Infect 和 Malicious URL Repackage 惡意攻擊手法。	A.2 隱私資料安全 B.2 應用程式資料安全 C.2 資料儲存安全
79	基於使用者行為之防肩窺攻擊行動裝置認證機制[158]	越來越多的隱私訊息會存在於裝置中，系統常用的身分認證存有一大難題－肩窺攻擊。許多預防肩窺攻擊的方法被提出，但是有些方法太過複雜造成使用不易，希望認證系統能不造成使用上負擔，並能達到防禦肩窺攻擊，且對於其他的攻擊行為也能達到足夠的安全性。	提出了一個能防禦肩窺攻擊多點觸控手勢認證系統，藉由智慧型裝置觸控螢幕擷取使用者多點觸控之手勢，使用者能在驗證時改變手勢混淆攻擊者，使攻擊者即使觀察也無法得知使用者所使用來通過認證之手勢密碼。我們的方法確實能改善能防禦肩窺攻擊，並能提高系統之安全性。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 B.3 原生環境混淆 C.1 傳輸協定與加密強度 C.2 資料儲存安全
80	Android 行動裝置隱私衝擊評估系統設計與實現[159]	行動裝置通常存放了許多的個人資料，例如：簡訊、照片、通訊錄、銀行憑證、帳號密碼等等。因此，維護個人資料的安全便日趨重要。	評估 Android 作業系統和應用程式對個人資料的影響以及降低行動裝置所帶來的隱私風險，參照隱私衝擊評估的概念。研究發現許多應用程式要求過多且可疑的權限，且應用程式首次安裝時，確認對話框上	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.2 資料儲存安全

編碼	題目	研究發現	研究方法	對應本研究規範項目
			顯示的權限清單並不完整。	
81	具 Root 權限之Android惡意軟體的有效檢測技術[160]	Google Play 的 App 上架審查制度不嚴謹的情況，雖然 Android 平台有多種安全機制保護系統運作，但駭客仍然有辦法躲過偵測，盜取受感染的智慧型行動裝置系統資源。	首先評測現有知名商用防毒軟體及 open source 之 Android Root 權限管理工具的安全性，進一步挑選較佳之 Root 權限防護管理工具，將其整合並強化，提出整合式 Android Root 權限防護評析機制。	A.1 權限提取不當 A.2 隱私資料安全 B.2 應用程式資料安全 C.2 資料儲存安全
82	Android 應用程式安全性分析之研究-以即時通APP 為例[161]	LINE 是台灣最多時使用的App，故將著重在即時通訊軟體 LINE 進行安全性分析，運用封包擷取及分析儲存的資料訊息，來進行 LINE App 資訊安全議題的研究。	將 Android 智慧型手機進行提權，並運用多種軟體工具，如：Wireshark 及 TCP dump，去擷取及分析 LINE 在網際網路上所傳送的封包，結果顯示 LINE 在傳輸訊息上是安全的，但是 LINE 在訊息儲存上使用明文是不安全的。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
83	應用多重 GPS 資訊於對稱式加密機制之研究-以Android智慧型行動裝置為例[162]	企業界興起導入「BYOD」的新潮流，員工透過行動裝置隨時掌握資訊、快速協同合作，不再受到固定裝置束縛，進而提高便利性，使工作生產力大為提升，但如果未能妥善管控行動裝置，將可能導致重要資料洩漏。	運用智慧型行動裝置上內建 GPS 定位做為限制條件，精進對稱式加密機制，以強化加解密程式複雜度及被破解的難度，藉由加密機制，結合 Google Maps 經緯度定址方式提供多重地點設定功能，提升加密資料的便利性及可利用性。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全
84	iOS 越獄鑑識與證據分析研究[163]	Apple 公司對於 iOS 系列裝置的規範，為能夠解套，便發展出	透過 Jailbreak 方式，進行 iOS 7.1.2、8.4、9.0.2 等版本的數位跡證萃取	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度

編碼	題目	研究發現	研究方法	對應本研究規範項目
		Jailbreak(越獄)手段，透過系統之漏洞或其他繞過安全檢查機制的方法,解除 Apple 公司對於 iOS 系列裝置的限制，取得裝置最高控制權限，不僅能夠變更 iOS 系統之功能，還能自行安裝原本不被允許的非官方軟體及擴充套件。	實測，並分析與比較 Jailbreak 前以及 Jailbreak 後的跡證萃取差異，以探討 Jailbreak 程序，對於數位跡證萃取成果的影響，利用 Jailbreak 特性，遠端連線擷取裝置內存資料，運用所萃取出的關鍵性跡證。	C.2 資料儲存安全
85	iOS 鑑識調查以行動手機於即時通訊之探索研究[164]	在 iPhone 或 iPad 裝置中,其 Apple 官方提供之 iTunes 軟體，透過對其執行同步備份所產生備份檔，備份檔包含了大部分儲存在設備上的資料，如聯繫人、短訊、MMS 訊息、設定檔、資料庫檔、照片、日曆、音樂、通話記錄、網路設置、Safari 使用紀錄、cookies 和相關應用程式資料，透過對備份檔進行鑑識分析也可能是有效鑑識分析方法。	利用 iPhone 手機同步備份檔，找出手機所使用的即時通訊軟體可能儲存之通訊紀錄、傳送之多媒體檔案等數位跡證。以備份檔數位鑑識調查方法研究是否能找到所要的數位跡證，同時針對與即時通訊具關聯性應用服務 App 進行鑑識研究調查，以所使用之備份檔鑑識方法，確認是否可以找到所要的相關數位證據。	A.2 隱私資料安全 B.2 應用程式資料安全 C.1 傳輸協定與加密強度 C.2 資料儲存安全

附錄三 本研究與國內檢測規範比較

檢測編號	檢測細項	國內規範		
		1	2	3
A.1.1	行動應用程式在發布時應完整說明存取敏感性資料、行動裝置資源及宣告權限用途。	✓	✓	✓
A.1.2	行動應用程式存取與個人資料相關敏感性資料應檢查行動應用程式是否提供相關身分授權機制。	✓	✓	✓
A.1.3	行動應用程式應於存取敏感性資料前，取得使用者同意。	✓	✓	✓
A.1.4	行動應用程式未經使用者授權准許使用非相關授權的函數。	✓	✓	
A.1.5	行動應用程式經使用者設定拒絕存取敏感性資料後，應用程式不得使用其他方式存取相關敏感資料。	✓	✓	✓
A.1.6	行動應用程式經使用者使用聯絡人或發送接收刪除訊息等功能伺服器必需紀錄，測試者應該嘗試存取另一個用戶的功能，以便驗證是否可以存取不應該被用戶的角色/特權所允許（但可能被允許作為另一個用戶）的功能。	✓		✓
A.2.1	行動應用程式開發過程中修改密碼執行不當動作所造成的漏洞。			
A.2.2	行動應用程式開發過程中是否有註銷功能藉此減少會話劫持攻擊的可能性。	✓	✓	
A.2.3	行動應用程式中程式碼或其他封裝之檔案內容，是否存放敏感訊息。	✓		✓
A.2.4	行動應用程式是否使用密碼強度策略防止隱私資料遭竊取。	✓		✓
A.2.5	行動應用程式有可能造成安全性漏洞功能是否設定為開/關。			
A.2.6	行動應用程式是否可在 Rooted / Dev Unlocked / Jailbroken…. 等環境下存取。			

檢測編號	檢測細項	國內規範		
		1	2	3
B.1.1	行動應用程式是否存在 SQL 攻擊的風險。	✓	✓	✓
B.1.2	行動應用程式是否具延伸標記語言攻擊字串的風險。		✓	✓
B.1.3	行動應用程式是否具逆向工程攻擊的風險。			
B.1.4	行動應用程式代碼是否以常量的方式將敏感資訊書寫在原始碼。	✓		✓
B.1.5	行動應用程式是否具 Session Fixation 攻擊的風險。	✓		
B.1.6	除錯設定設定值是否安全。			
B.1.7	是否具 FTP 協定注入式攻擊。			✓
B.2.1	行動應用程式在設定時間內不活動時是否具自動關閉應用程式或鎖定功能。		✓	
B.2.2	行動應用程式是否具 LDAP 注入攻擊的風險。	✓		✓
B.2.3	行動應用程式是否具 OS 命令注入攻擊的風險。	✓		✓
B.2.4	行動應用程式所產生的敏感訊息是否存在於記憶體傾印中。	✓	✓	✓
B.2.5	行動應用程式透過惡意方式所產生漏洞導致資料安全暴露在危險的風險。	✓		✓
B.3.1	行動應用程式是否具 XSS 攻擊的風險。			✓
B.3.2	行動應用程式是否具惡意檔案上傳的風險。			
B.3.3	行動應用程式是否存在 IOS 快照漏洞。			
B.3.4	行動應用程式透過惡意再包裝與混淆技術造成安全上的風險。	✓		

檢測編號	檢測細項	國內規範		
		1	2	3
C.1.1	行動應用程式透過網路傳輸敏感性資料時，是否使用加密傳輸以確保敏感性資料安全。	✓	✓	✓
C.1.2	行動應用程式是否具交談識別碼遭重送攻擊的風險。	✓	✓	
C.1.3	行動應用程式與付費功能伺服器間之資料加密傳輸，是否使用安全之加密演算法。	✓	✓	✓
C.1.4	行動應用程式與伺服器間之資料加密傳輸，是否使用安全之加密演算法。	✓	✓	✓
C.1.5	行動應用程式是否具離線狀態繞過認證的風險。			
C.1.6	行動應用程式目前狀態是否驗證 MSISDN 號碼。			
C.1.7	行動應用程式是否具被繞過二級身份驗證的風險。	✓		
C.1.8	行動應用程式內是否清除 SSL Tunnel 中的文件資訊。	✓		
C.1.9	行動應用程式是否具被繞過客戶端驗證風險。			
C.1.10	行動應用程式中 SSL 憑證/ SSL / TLS 密碼/協議/密鑰無效風險。	✓	✓	✓
C.1.11	行動應用程式中是否具敏感訊息以明文形式暴露的風險。	✓	✓	✓
C.1.12	CAPTCHA 沒有使用至行動應用程式的公共頁面/登錄頁面上。			
C.1.13	是否具敏感訊息作為查詢字串參數繞過安全機制風險。	✓		
C.1.14	檢查是否具網址參數修改攻擊風險。	✓		
C.2.1	行動應用程式應將帳號密碼儲存於作業系統保護區內或以加密方式儲存。	✓	✓	✓

檢測編號	檢測細項	國內規範		
		1	2	3
C.2.2	行動應用程式於儲存敏感性資料時應提供資料加密功能，以避免遭不正當方式取得敏感性資料。	✓	✓	✓
C.2.3	行動應用程式與遠端伺服器溝通之帳號密碼不應以明文方式存在於執行檔中，以避免遭不正當的方式存取。	✓	✓	✓
C.2.4	應用程式使用加密強度不足。	✓	✓	✓
1. CISA-行動應用 App 安全開發指引。 2. NCC-手機系統內建軟體資安評估。 3. 工業局-行動應用 APP 基本資安規範。				