# 안드로이드 어플리케이션 보안 취약점에 관한 연구†

한찬규*, 강성용*, 장학범**, 최형기**
*성균관대학교 휴대폰학과
**성균관대학교 정보통신공학부
e-mail : {hedwig, sykang, hbjang, hkchoi}@ece.skku.ac.kr

# Security Vulnerabilities in Android Applications

Chan-Kyu Han*, Seong-Yong Kang*, Hak-Beom Jang**, Hyoung-Kee Choi**
*Department. of Mobile Systems Engineering
**School of Information & Communication Engineering

**요 약**

최근 안드로이드가 오픈 소스 및 오픈 서비스를 기반으로 출시되었으나, 성장률에 비하여 보안성이 검증된 사례가 미미하여, 보안대책 마련이 시급한 실정이다. 따라서 본 논문에서는 안드로이드보안 아키텍쳐 및 안드로이드 어플리케이션 보안에 관하여 연구하였다. 또한 안드로이드 프레임워크 상에서 개인정보(단말의 위치정보 및 외부 계정정보)가 유출되는 보안 취약점을 실험연구를 통해 발견하였고, 이에 대한 보안대책을 제안하여, 안전한 스마트폰 환경을 제공하고자 한다.

## 1. 서론

최근 풀브라우징 기반의 인터넷, 위치기반서비스, 개인 블로그, 쇼핑 등의 다양한 서비스가 모바일 상에서 가능한 스마트폰이 새로운 휴대폰 산업으로 급부상하고 있다. 이러한 스마트폰은 단말 제조업체와 통신 사업자들의 이해관계를 아울러 충족시키는 가장 적합한 도구로 평가되고 있다. 2007 년 애플의 아이폰(iPhone)이 출시된 이후 일반 소비자들의 관심이 증폭되면서 스마트폰 시장이 확산되고 있다. 한편 구글(Google)이 개발한 자체 운영체제인 안드로이드(Android) [1]를 탑재한 휴대폰이 출시되면서 스마트폰 시장 내 경쟁이 더더욱 치열해지고 있다. 안드로이드는 독특한 아키텍쳐 및 오픈소스로 시장점유율을 확대하고 있으며 따라서 잠재된 성장가능성이 있다고 평가되고 있다.

안드로이드는 악의적인 어플리케이션의 악용을 방지하기 위해 퍼미션 기반 보안구조를 채택하였지만, 오픈소스의 특성에 따라 취약점에 노출될 가능성이 다수 존재한다. 또한 안드로이드가 채택한 리눅스 기반 시스템은 루트권한이 획득되면 상대적으로 다양한 공격에 노출될 가능성이 크다. 안드로이드의 경우 성장률에 비하여 보안성이 검증된 사례가 미미하며, 오픈소스의 특징으로 인해 더욱 기민한 보안대책이 마련이 시급한 실정이다.

하지만 현재까지의 관련 연구들은 스마트폰 보안 취약점의 안드로이드 적용여부에 관한 내용이 다수를 차지한다. 현재 안드로이드 어플리케이션 개발에만 관심이 집중되어 있고, 보안 아키텍쳐에 대한 접근은 미미한 실정이다. 대부분의 보안 관련 선행 기술들은 스마트폰 보안을 위해 연구되었으므로, 안드로이드 아키텍쳐 특성에 따른 보안연구는 미미한 수준이다. 따라서 안드로이드 시스템의 특징을 고려한 보안 위협 및 취약점 분석에 관한 연구가 절실한 시점이다.

본 논문에서는 안드로이드의 어플리케이션 프레임워크 상의 취약점에 대한 가능성을 타진하며, 보안취약점에 관한 대책을 마련하고자 한다. 안드로이드 어플리케이션 보안은 퍼미션 기반에 의존하고 있고, 이때의 퍼미션 허용여부는 사용자의 선택에 따른다. 이러한 사용자 의존적인 보안구조는 악성 어플리케이션의 공격을 허용한다. 본 논문에서는 로그정보 및 어플리케이션 프레임워크를 이용하여 사용자의 개인정보를 유출하는 공격을 제작하였다. 동작과정을 분석하여 안드로이드 보안취약성의 주원인을 도출하고, 이에 따른 안드로이드 보안대책으로 어플리케이션 퍼미션 기법을 수정하고자 한다.

논문의 구성은 다음과 같다. 2 장에서는 안드로이드 특성에 따른 취약점 및 보안 관련연구를 기술하고, 3 장에서는 안드로이드 보안 아키텍쳐의 특성에 대하여 정리한다. 4 장에서는 본 논문에서 발견한 안드로이드 보안 취약점을 정리하고, 5 장에서는 보안대책에 관해 논의한다. 마지막으로 6 장에서 향후 연구와 함께 결론을 기술한다.

† 이 논문은 2011 년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임 (No.2011-0005037)

## 2. 관련연구

안드로이드 보안에 관한 선행 분석 연구 [2]에서는 어플리케이션의 구성요소 및 보안 퍼미션에 관련된 개괄적인 설명을 도시하였다. [3]에서는 안드로이드 상에서의 서비스거부공격, 디바이스 재부팅, 서비스 무한루팅 등의 악성 어플리케이션을 개발하였다. 또한 0에서는 어플리케이션 퍼미션의 보안 취약점을 이용하여, 시스템 내 어플리케이션을 무단 사용하는 공격기법을 제안하였다. 이밖에 해킹기법인 Return-Oriented Programming 기법을 활용한 루팅 [5], 안드로이드 G1 디바이스 상에서의 서비스거부공격 [6] 등이 개발되었다. 안드로이드 상에서 공격탐지에 관한 관련연구는 다음과 같다. 보안정책 및 어플리케이션의 퍼미션 사용을 동적으로 검사하거나 [7], 정보흐름을 추적하기 위해 소프트웨어기법인 taint analysis 를 활용한 추적방법 [8]이 제안되었다. [8]에서는 변수, 네이티브 코드, 컨텐츠, 개인정보 등을 대상으로 어플리케이션 간 메시지를 추적하여, 구글 넥서스원에서 테스트하였다. 그 밖에 Multi level security 를 도입하여 권한제어를 강화한 Security-Enhanced Linux 를 탑재한 안드로이드가 개발되었고 [9], 정적분석을 통한 악성코드 패턴매칭 및 소스코드에 대한 디어셈블링 방법 또한 제안되었다[10].

## 3. 안드로이드 보안 아키텍쳐

안드로이드 보안 아키텍쳐는 (1) 어플리케이션, (2)어플리케이션 프레임워크를 포함한 미들웨어, (3) 리눅스 운영체제 커널이 계층적으로 구성된다. 어플리케이션은 SMS, 이메일 등의 기본 어플리케이션 외, 개발자가 탑재한 어플리케이션이 자바로 구현된다. 미들웨어인 어플리케이션 프레임워크 상의 각 어플리케이션 매니저들은 리소스, 시스템상태, 컨텐츠 등의 접근을 관리한다. 런타임 가상머신에서는 어플리케이션 별로 개별 가상머신을 운용하여 안전한 개체관리를 도모한다. 라이브러리 및 커널에서는 각종 시스템 라이브러리, 하드웨어 종속적인 기능을 지원한다.

각 어플리케이션은 **AndroidManifest.xml** 이라는 설정파일 (manifest 파일)을 유지한다. Manifest 파일에서는 어플리케이션의 구성 컴포넌트를 명시하고, 어플리케이션에서 필요한 퍼미션 및 각 컴포넌트를 보호하는 퍼미션을 기술한다. Manifest 파일에서는 해당 어플리케이션 또는 다른 어플리케이션들이 특정 컴포넌트에 대한 접근제한을 하기 위한 퍼미션을 **<permission>** 메소드를 통해 정의한다 어플리케이션의 특정 동작을 위해 (예. 인터넷 접속) 퍼미션이 필요하다면, **<use-permission>** 메소드를 통해 요청한다. 구글에서는 약 110 개의 퍼미션을 통해 READ_CALENDAR (일정), READ_SMS (단문메시지) 등 개인정보에 접근할 수 있는 퍼미션이나, VIBRATE (진동), CAMERA (카메라) 등 하드웨어 자원을 사용할 수 있는 퍼미션을 정의하고 있다. 또한 INTERNET (네트워크 소켓), BLUETOOTH (블루투스)와 같은 통신정보나, ACCESS_FINE_LOCATION (GPS 정보) 등과 같은 위치정보를 포함하고 있어 더욱 주의 깊은 사용이 요구된다.

각 퍼미션은 위험도에 따라 다음의 4 개 레벨로 분류하고 있다: **Dangerous, Normal, Signature, Signature or System**. 안드로이드 어플리케이션은, 설치 시에 어플리케이션이 사용하고자 하는 퍼미션 정보를 알리게 된다. 전화송신 (Phone calls), SD 카드정보 (Storage) 등과 같이 Dangerous 로 설정된 퍼미션은 사용자에게 경고를 띄우게 되어 있다. 한편 Normal 로 설정된 퍼미션은 사용자에게 경고를 띄우지 않는다.

사용자는 어플리케이션의 특성을 고려하여, 필요하지 않은 퍼미션을 어플리케이션이 과도하게 요청하고 있다고 간주할 시에는 어플리케이션 설치를 취소할 수 있다. 예를 들면, 게임 관련 어플리케이션이 주소록과 관련된 퍼미션을 요구하는 것은 어플리케이션의 목적에 알맞지 않은 경우이다. 이처럼 안드로이드는 사용자의 퍼미션 정보 인지와 함께 어플리케이션 설치 시에 사용자의 분별력 있는 판단을 요하고 있다. 하지만 해당 퍼미션이 필요한 상황에 대한 기술 없이, 설치 시에 퍼미션 판단을 요구하고 있어 정확한 판단은 불가능한 실정이다. 또한 안드로이드에서는 시스템에 정의된 퍼미션 외에, 개발자가 같은 어플리케이션 간의 사용을 위해 개발자가 정의하는 퍼미션도 허용하고 있어 사용자 판단이 더욱 어려운 실정이다. 또한 어플리케이션이 일단 한번 설치되면, 퍼미션에 대한 사용자 확인은 다시 이루어지지 않는다는 문제점이 있다.

## 4. 안드로이드 보안 취약점

본 논문에서는 사용자 판단에 의존하는 안드로이드의 퍼미션 관련 취약점을 이용한 공격을 발견하고, 공격의 구체적인 동작과정을 도시한다.

### (1) 로그파일 유출 공격

각 어플리케이션은 디버깅을 위하여 단말과 관련된 정보를 로그파일에 기록할 수 있다. 따라서 안드로이드 로그파일에는 웹 방문기록, 문자, 주소록, 위치정보 등이 저장되어, 로그파일 유출 시 심각한 개인정보 유출의 문제가 있다. 본 논문에서는 로그 관련 정보 유출을 분석하기 위해 안드로이드 SDK 에서 제공하는 Logcat 정보를 이용하였다. Logcat 은 이클립스 개발환경 상에서 안드로이드 운영체제의 로그메시지를 도시한다. 로그는 dalvik 가상머신, 어플리케이션 프레임워크 상의 각종 매니저, 그리고 리눅스 운영체제 상에서의 로그를 포함하여, 어플리케이션 자체 로그 또한 도시하고 있다.

그림 1에서는 통화송신을 시도하는 단말의 로그정보로부터, 통화기록(송신번호)을 유출하는 악성 어플리케이션을 도시한다. 안드로이드 상에서는 모든 어플리케이션이 로그파일을 공유함으로써, READ_LOGS 퍼미션을 명시하는 어플리케이션이라면 다른 어플리케이션이 기록한 통화기록, 문자정보, 계정정보를 취득할 수 있다는 문제점이 있다. READ_LOGS 퍼미션의

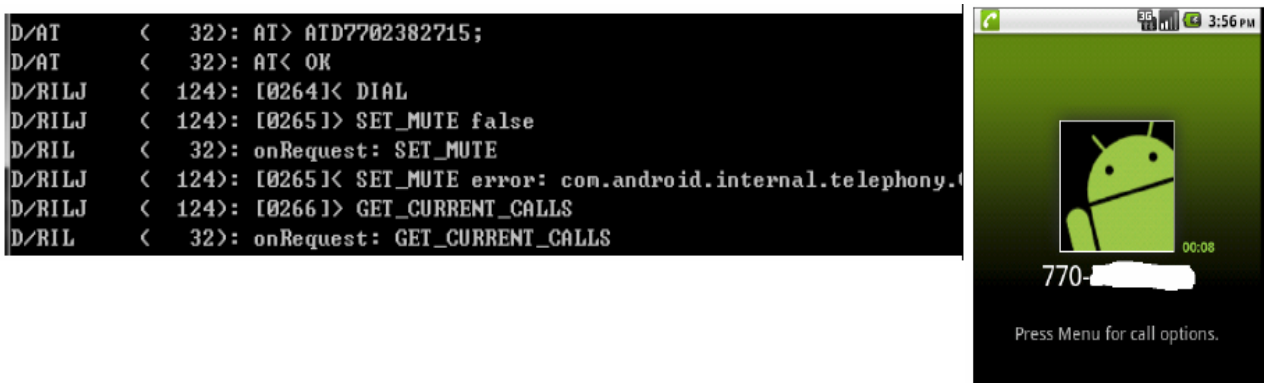


그림 1. 단말의 로그정보 유출

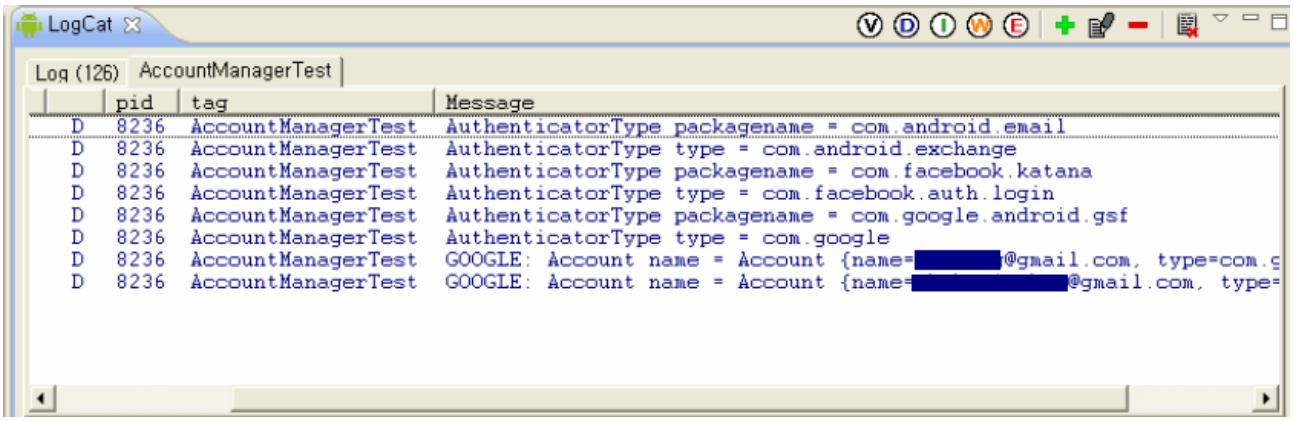


그림 2. 단말에 등록된 account 유출

경우 Dangerous 로 정의되어 설치 시에 경고창을 띄우게 되지만, 과금이나 주소록 등에 대한 정보접근 관련 퍼미션보다는 거부장벽이 낮아서 사용자들은 쉽게 설치를 허용한다. 따라서 그림 1과 같이 전화 어플리케이션의 로그정보를 이용하여, 사용자의 통화기록을 도청하는 것이 가능하다. 실제로 안드로이드 개발자 사이트에서는, 어플리케이션 개발 시 중요한 사용자 정보 및 디바이스 정보를 로그에 기록하지 않도록 권고하고 있다.

### (2) 어카운트 정보 유출 공격

안드로이드 디바이스는 초기부팅 시 사용자의 구글계정입력을 요구한다. 안드로이드 디바이스에 설치된 구글서비스 (Google Mobile Service: GMS) 들 (예. 구글 지메일, 구글토크, 유투브 등)은 사용자의 구글계정에 대한 **Authenticator** 토큰을 공유한다. 또한 3 자 어플리케이션 서비스 어플리케이션 설치 시 (예. 페이스북, 트위터 등) 계정정보를 입력하게 된다. 어카운트 매니저 (Account Manager)는 구글, 페이스북 (Facebook) 등 사용자의 계정을 관리하고, 계정접근 및 권한인증 등을 담당한다.

사용자의 계정에 관련된 정보를 접근하기 위해서는 GET_ACCOUNTS, USE_CREDENTIALS 등이 필요하며, 계정에 접근할 때는 어카운트 매니저가 할당한 Auth token 을 사용한다. 어카운트 매니저는 다양한 외부 계정에 따라 Authenticator 토큰을 운용하며, 해당 Authenticator 에 알맞은 Auth token 을 생성하는 일을 담당한다. 본 연구팀에서는 표 1과 같이 단말에 등록된 모든 계정의 정보를 얻어오는 샘플테스트를 작성하였으며, 그림 2과 같이 단말에 등록된 사용자의 어카운트 정보를 모두 얻어오는 데 성공하였다. 이 때 로그정보를 유출하는 공격을 사용하면, 단말의 어카운트 정보를 유출할 수 있다. 이러한 공격은 사용자의 개인정보를 침해하여, 제 3 의 악성공격으로 발전할 가능성이 있다.

본 논문에서는 추가로 해당 계정의 정보를 이용하여 허위로 로그인을 시도하는 공격코드를 작성하였다. 하지만 취득한 계정정보를 이용하여, 실제 온라인 계정정보에 접근하려 하는 공격코드를 작성하자 안드로이드 미들웨어에서 Security Exception 이 발생하였다. 이는 공격 어플리케이션이 정상 Authenticator 어플리케이션 (=구글 GMS 어플리케이션)과 User Identifier (UID)가 달라서 발생하는 것으로 분석되었다. 하지만 이 때 안드로이드 디바이스는 그림 3처럼 본 논문에서 작성한 AccountManagerTest 에 대하여 구글계정에 대한 퍼미션을 묻게 되는데 사용자가 허용하게 되면 구글계정에 대한 비밀번호 입력을 요구하게 된다. 이 때 공격어플리케이션에서 텍스트입력을 가로채거나, 네트워크 패킷을 캡쳐한다면 사용자 계정에 대한 비밀번호 획득이 가능하다.

## 5. 보안대책에 관한 논의

본 논문에서는 발견된 어플리케이션 취약점에 대한 보안대책으로 다음의 내용을 제안한다. 어플리케이션 취약점은 안드로이드의 퍼미션 시스템에서 기인한 것으로, 다음과 같이 퍼미션 시스템을 수정한다면 어플리케이션 취약점을 방지할 수 있다.

A. **퍼미션의 한시적 허용**: 어플리케이션 퍼미션의 영구 허용을 지양하고, 실행 시 퍼미션 정보를 동적으로 확인하는 기법이 필수적이다. 또한 설치 시 특정 퍼미션에 대한 요청을 선택적으로 거부할 수 있도록 하여, 실제 실행 시 필요한 퍼미션을 일시 허용할 수 있도록 하고자 한다. 이 경우 사용자의 무분별한 퍼미션 요청수락을 방지할 수 있고, 해당 퍼미션이 필요한 시점을 정확하게 인지하게 할 수 있다.

B. **퍼미션의 세분화**: 악성행위를 탐지하기에는 퍼미션의 허용 범위가 과다한 경우가 있다. 예를 들어 INTERNET 과 같은 퍼미션의 경우, 허용범위가 과도하게 넓어 개발자의 의도를 알 수 없다. 따라서 로그, 인터넷 등의 퍼미션을 세분화하여, 개발자의 의도를 분명히 하고 사용자로 하여금 퍼미션에 대한 인식을 환기하고자 한다.

## 6. 결론

본 논문에서는 안드로이드 보안 아키텍쳐 및 안드로이드 어플리케이션 보안에 관하여 분석하고, 어플리케이션 퍼미션 시스템이 취약함을 발견하였다. 실제 개발연구로 안드로이드 프레임워크 상에서 개인정보(단말의 위치정보 및 외부 계정정보)가 유출되는 보안 취약점을 실험연구를 통해 발견하였고, 이에 대한 보안대책을 제안하였다. 향후 연구로는 제안한 취약점 공격을 실제 안드로이드 디바이스 상에서 테스트하고, 안드로이드 운영체제 프레임워크를 수정하여 보안대책을 구현하고자 한다. 이를 위해 퍼미션을 검사하는 어플리케이션 매니저 등의 수정이 요구된다.

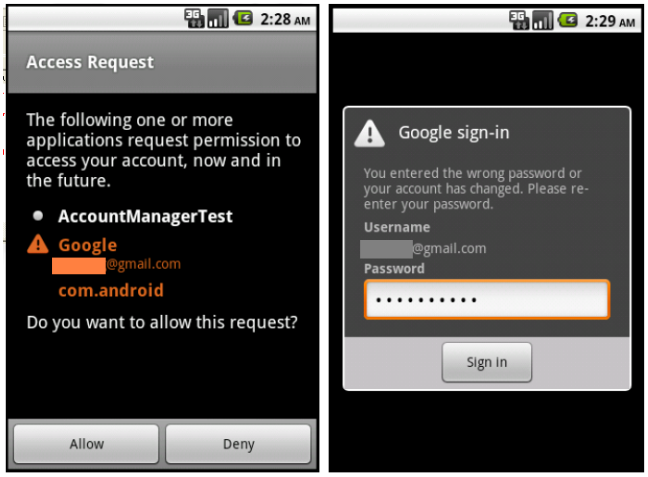


그림 3. AccountManager Test 에서 요청하는 사용자 계정에 대한 비밀번호 유출


### 참고문헌

[1] Android developers, *at available* http://developer.android.com/index.html

[2] W. Enck, *et al*., "Understanding Android Security," *IEEE Security and Privacy*, Jan. 2009.

[3] A. Lineberry, *et al*., "These Aren't The Permissions You're Looking For," *Blackhat* Jul. 2010.

[4] W. Shin, *et al*., "A Small but Non-negligible Flaw in the Android Permission Scheme," *IEEE POLICY*, Jul. 2010.

[5] S. Checkoway, *et al*., "Return-Oriented Programming without Returns," *ACM CCS*, Oct. 2010.

[6] A.-D. Schmidt, *et al*., "Smartphone Malware Evolution Revisited: Android Next Target?," *IEEE MALWARE*, Oct. 2009.

[7] D. Barrera, *et al*., "A Methodology for Empirical Analysis of Permission-based Security models and its Application to Android," *ACM CCS*, Oct. 2010.

[8] W. Enck, *et al*., "Taintdroid: An Information-flow Tracking System for Realtime Privacy Monitoring on Smartphones," *USENIX OSDI*, Oct. 2010.

[9] A. Shabtai, *et al*., "Securing Android-Powered Mobile Devices using SELinux," accepted for publication in *IEEE Security and Privacy*

[10] A.-D. Schmidt, *et al*., "Static Analysis of Executables for Collaborative Malware Detection on Android," *IEEE ICC*, 2009.


표 1. 어카운트 매니저를 이용한 어카운트 정보 획득 (AccountManagerTest)

```
1    AccountManager am = AccountManager.get(this);
2    AuthenticatorDescription[] authTypes= am.getAuthenticatorTypes();
3    for(int q=0; q<authTypes.length; q++){
4       Log.d(TAG,"AuthenticatorTypepackagename= " + authTypes[q].packageName);
5       Log.d(TAG,"AuthenticatorTypetype = " + authTypes[q].type );
6       Account [] accounts = am.getAccountsByType("authTypes[q].type");
7     Log.d(TAG, "Account name = " + gacounts[j]); }
```