http://dx.doi.org/10.7236/JIIBC.2013.13.6.123
JIIBC 2013-6-16

개인정보를 위한 안드로이드 저장장치 접근제어

Android Storage Access Control for Personal Information Security

유재만*, 박인규**

Jae-Man You, In-Kyoo Park

요 약 안드로이드 파일시스템은 임의적 접근제어 방식을 사용하여 시스템의 자원에 접근할 수 있는 오픈 시스템이기 때문에 상대적으로 저장장치에 대한 제어가 필요하다. 이러한 특성으로 인해 안드로이드의 VDC를 통해서 접근제어를 하는 경우 안드로이드가 VDC 기능을 제공하는 경우에만 가능하다는 문제점이 있다. 이를 개선하는 방법으로는 VDC의 기능을 시스템 콜(system call)을 통하여 직접 구현함으로써 OS와는 별도로 연동 모듈을 만들어 저장장치 제어기능을 추가하여야 한다. 본 논문에서는 일반적인 저장장치의 마운트 시스템에 대하여 VDC 기능을 이용하여 사용자에 대한 접근을 제어하는 방법을 제안하였다. SD, UMS 와 같은 저장장치에 대한 접근제어를 마운트 방식에 의하여 구현하였고 제안된 기법이 제어가 설정된 저장장치에 파일을 복사/저장하려면 쓰기가 금지되어 제어가 수행됨을 실험을 통하여 검증하였다.

Abstract Android file system is vulnerable to the external access of system resources via its arbitrary access mode and need user's control for SD and UMS medias due to its open architecture. In response to the device control, there is a drawback that its controlability is valid only in the case of embedded linux kernel with VDC function. Hence the solution is to directly implement VDC through system call, with another security module for device storage than system module being added to android system. In this paper the new method of android storage access control for personal information is proposed via VDC for mount system of storage. The access method for SD and UMS were implemented using VDC and mount mechanism. This access control system has been designed to control the granted users in kernel level if files are flowed out by copying. As a result, it was proved through testing that the access control system has exactly detected the write access operation.

Key Words : Android, Security, Storage Media, Embedded, Volume Daemon Command

Ⅰ. 서 론

일반적으로 임베디드(embedded) 시스템은 엘리베이터, TV, MP3 플레이어, 셋톱박스, 디지털 카메라, PDA, 휴대폰, 자동차 엔진 제어, 의료기기 등 일상생활과 매우 밀접한 관계를 가지고 있다. 따라서 그에 대한 응용범위가 점점 다양해지고 임베디드 소프트웨어의 수요가 매우 빠르게 급증하고 있다. 이에 대한 응용 시장의 요구로 인하여 그에 따른 여러 가지 서비스와 관련 기반 기술들이 개발되어왔다. 현재는 데스크톱 PC에서의 인터넷 뱅킹, 홈쇼핑 등의 응용이 임베디드 시스템에서 가능해지고 있으며, 이에 따른 개인의 중요한 정보를 임베디

*정회원, 중부대학교 정보과학과
**정회원, 중부대학교 컴퓨터학과(교신저자)
접수일자 2013년 10월 16일, 수정완료 2013년 11월 19일
게재확정일자 2013년 12월 13일

Received: 16 October, 2013 / Revised: 19 November, 2013
Accepted: 13 December, 2013
*Corresponding Author: fip2441g@gmail.com
Dept. of Computer Science, Joongbu University, korea

드 시스템에서도 가능하기 때문에 임베디드 시스템에서 사용하는 임베디드 운영체제의 보안 기술의 필요성은 시간이 갈수록 중요해 질 전망이다. 따라서 외장 SD, UMS를 통한 자료 공유와 교환이 가장 손쉽기 때문에 외부로부터의 시스템에 관련된 파일이 오염되거나, 유출되었을 경우 심각 보안 위협이 된다. 파일 보안은 여러 가지 관점에서 접근할 수 있다. 예를 들어 시스템 콜에 의한 입출력 제어, 네트워크 입출력 제어, 파일 시스템 단에서의 보안 등 다각적 접근 방법이 있을 수 있다.

안드로이드의 플랫폼에 대한 취약점은 자주 언급되어 왔다. 먼저 시스템 안정화를 위한 시스템의 언락(unlock)을 들 수 있는데 안드로이드는 시스템서비스에 대한 보안정책을 요구하지 않기 때문에 이러한 특권서비스는 디바이스의 자원을 접근하기 위한 보안검사를 경유하지 않기 때문에 보안에 취약성을 가지고 있다. 둘째로 루트권한 획득을 통하여 안드로이드가 가지는 보안 메커니즘이 적용되지 않는 문제점이 존재한다. 따라서 안드로이드는 보안모듈을 유저공간에 배치하기 때문에 안드로이드 프레임워크의 공격에 취약하다. 따라서 응용 프로그램 뿐만 아니라 안드로이드 모듈을 보호할 수 있는 보호프레임워크가 필요하다. 마지막으로 일반 애플리케이션도 루트권한 획득을 통해 시스템 레벨로 접근이 가능하다는 점이다. vold는 커널과의 메시지를 통하여 마운팅 기능을 처리하는데 문제는 전송되는 데이터와 송신자에 대한 인증을 하지 않는 단점이 있다. 본 논문에서는 개인정보의 보호를 위하여 일반적인 데이터 교환 수단인 리무버블(removable) 저장장치와 UMS 를 대상으로 마운트 시스템에 의하여 허가된 사용자만 데이터 접근이 가능하도록 하는 방법을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구를 논하고, 3장에서는 제안한 기법에 대하여 기술한다. 4장에서는 제안한 방법의 실험 및 결과를 논하고 5장에서는 결론을 맺는다.

II. 관련 연구

시스템 관리와 보호를 위해서는 외부 사용자뿐만 아니라, 임의의 절차에 의하여 인증된 내부 사용자에 대한 관리와 접근제어 또한 갖추어야 할 요건이다. 안드로이드 내부에 있는 모든 응용 프로그램은 커널이 제공하는 서비스에 의존하며 커널에 요청하는 시스템 콜을 이용하여 프로세스와 장치간의 인터페이스를 구현한다. 이러한 시스템 콜은 사용자 공간과 커널 공간 사이의 인터페이스를 수행하는 것으로써, 장치 접근, 파일 접근, 프로세스 관리, 네트워킹 등의 서비스를 제공한다. 이러한 시스템 콜을 수정함으로써 안정적으로 보안을 적용할 수 있다.

Encrypt-FS[1]는 시스템 레벨에서 동적 데이터 암호화와 복호화하는 암호화 파일 시스템으로 구현하고 있다. Encrypt-FS는 암호화 엔진을 제공하는 Linux VFS layer에서 작동되며 상위 프로세스는 동적으로 암호화와 복호화를 할 수 있다. Encryp-FS 는 파일을 암호화하기 위해서 두 가지의 암호화 알고리즘인 비대칭 암호화 알고리즘 AES, 대칭 암호화 알고리즘인 공개키 알고리즘 RSA를 사용하고 있다.

SELinux 기반 안드로이드 보안시스템[2]으로 일반 Linux에서 동작하는 SELinux을 안드로이드에서 사용하도록 구축하였다. 안드로이드는 Android 플랫폼 구동을 위한 부분을 제외하면 일반 Linux 커널을 그대로 사용하고 있다. Linux 배포판들은 약한 보안 기능을 보완하기 위해서 방화벽, SELinux 등을 활성화시키거나 여러 가지 보안에 관한 어플리케이션을 사용하여 시스템의 보안성을 강화하려고 하고 있다. SELinux는 root 권한에서도 모든 권한을 가질 수 없도록 되어 있다. SELinux 는 사용하기가 매우 어렵지만 환경 설정만 잘해 놓으면 유용한 보안 시스템이 될 수 있다.

안드로이드에서 강제적 접근제어에 의한 보안커널모듈로 리눅스 서버급의 시스템에서 적용되던 강제적 접근제어를 용용하여 안드로이드용 보안 커널모듈로 보안을 강화하는 방법[3]을 보여주고 있다. 안드로이드의 보안정책의 특성으로 사용자가 다운로드하여 실행하는 모든 응용프로그램을 잠정적인 보안 위협 요소로 취급하여 시스템 자원에 대한 접근을 강제적으로 통제하여 보안을 강화하고 있고 안드로이드 상에서 보안을 강화하기 위해서 여러 가지 접근 방법들이 제안되어 왔다.

III. 제안기법

1. 제어시스템의 구성환경

개인정보 보호를 위한 타겟은 안드로이드 2.2 Froyo

이고 개발 환경은 Ubuntu 9.04 커널 버전 2.xxx-generic과 Android NDK, Revision(5c)를 사용하였다. 장치제어를 위한 마운트 방식의 Native서비스의 흐름도는 그림 1과 같고 안드로이드 프로요의 경우 vdc가 실제 장치의 마운트를 담당하고 있다.

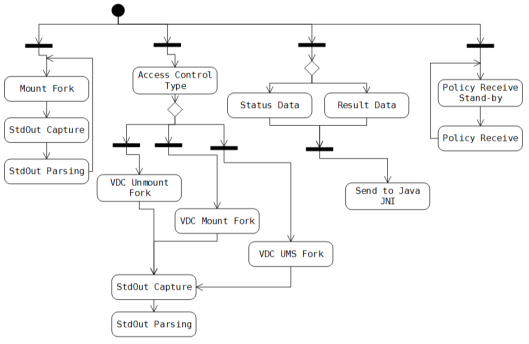

그림 1. 네이티브 서비스의 흐름도
Fig. 1. Flow chart of the native service

2. 접근제어의 설정범위

장치에 대한 접근의 단위는 기본적으로 Mount Point이므로 Mount Point 기반의 장치에 모두 적용이 가능하다고 할 수 있다. 안드로이드의 경우 Internal/External SD - Memory가 Mount Point로 접근이 가능하다고 할 수 있다. 제안된 시스템에서는 환경파일에서 안드로이드에서의 고정된 Mount Point를 등록하여 장치를 제어하도록 하고 있다.

INTERNELSDMOUNTPOINT=/mnt/sdcard
EXTERNELSDMOUNTPOINT=/mnt/sdcard/sdcard/

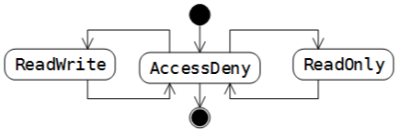

그림 2. 장치 접근의 상태도
Fig. 2. State diagram of device access

그림 2에서 장치의 상태가 세 가지로 구성되어 있으며 최초의 상태는 AccessDeny 로서 사용할 수 없는 상태이다. 사용자가 읽고 쓰기 가능한 상태가 가장 일반적인 ReadWrite 상태이고 ReadOnly 상태를 읽기만 가능하고 쓰기는 불가능한 상태이다. 안드로이드의 mountd 또는 vold 라는 장치 마운트 데몬과 연동하여 명령어 상에서 vdc를 통하여 장치를 마운트할 수 있다.

3. 저장장치의 접근에 대한 제어동작

저장장치 접근제어시스템은 그림 3과 같이 크게 네 가지 모듈로 구성하였다. 장치보안 Native서비스가 그 중 가장 핵심적인 모듈로서 실질적으로 저장 장치에 대한 접근을 제어하는 서비스를 제공한다[4]. 장치보안 JavaJNI 모듈은 타 시스템에 적용시에는 이 모듈이 없이 바로 어플리케이션과 핵심 서비스를 연결할 수 있지만, 안드로이드에서는 Java 어플이 Native 프로세스를 직접 연결할 수 없기 때문에 JavaJNI가 중간에서 중재 역할을 한다.

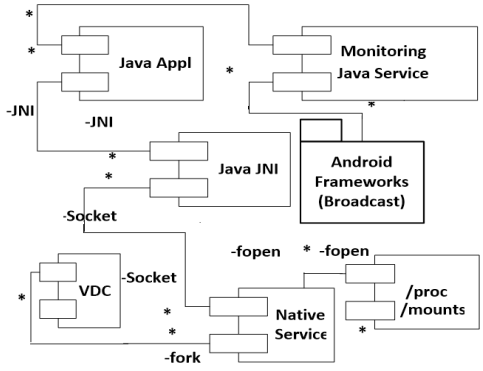

그림 3. 전체적인 모듈 개요도
Fig. 3. The entire module diagram

장치제어 JAVA응용 모듈은 사용자가 직접 정책을 설정하거나, 모니터링 Java서비스로부터 장치의 상태 변경을 수신하는 역할을 담당하고 있다.

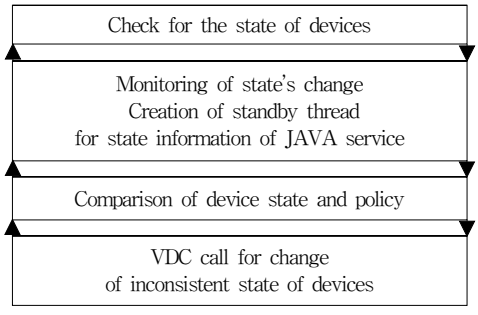

그림 4. 네이티브 서비스의 알고리즘
Fig. 4. Native service algorithm

장치 제어 Native 서비스는 외부 저장 장치를 마운트, 언마운트를 담당하는 모듈로써 안드로이드 서비스 형태로 구동된다. 장치 모니터링 모듈과 연동하여 장치가 연결 상태가 변동시 장치 모니터링으로 부터 이벤트를 수신하여 접근 권한 설정에 따라 실제 제어를 하게 된다. 장치 모니터링 서비스 모듈에서는 장치의 상태가 변경할 때, 안도로이드의 프레임워크는 브로드케스트의 이벤트를 발생시킴으로써 장치를 상태 변화를 모니터링 할 수 있다. 장치 모니터링 서비스 모듈은 Java 서비스 모듈로써 브로드케스트되는 이벤트를 수신한다[5]. 따라서 수신하는 이벤트는 장치의 장착, 제거, UMS Connection 등으로써 이벤트를 수신하면 이 정보를 다시 장치 제어 서비스로 전송하여 장치의 상태가 변경되었음을 알린다.

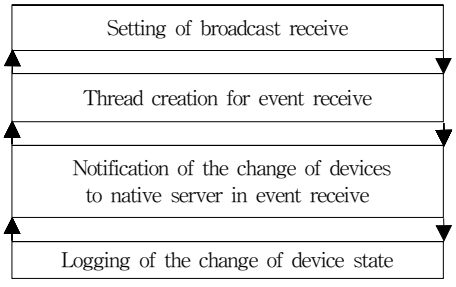

그림 5. 모니터링 서비스의 알고리즘
Fig. 5. Monitoring service algorithm

장치제어시스템의 모듈, 형태, 설치 및 실행 권한과 구현 언어는 Table 1과 같다. Native Service 의 설치와 실행 권한이 루트이기 때문에 루트 권한이 없으면 Native Service 를 작동시킬 수 없다.

표 1. 디바이스 제어의 명세
Table 1. Specification of device control

Module	Type	Install/Run authority	Language
Java App	Activity	Application	Java
Java Service	Service	Application	Java
JNI	JNI	Application	C
Native Service	Service	Root	C

Ⅳ. 실험 및 결과

1. 시험환경

안드로이드 플랫폼 환경에서의 저장장치 접근 제어를 검증하기 위하여 임베디드 타겟 보드 기반의 프로토타입 시험 환경을 구성하였다[6,7]. H-AndroSV210 타겟 보드는 Micro-SD Memory 에 부트로더, 커널, 램디스크, 안드로이드 시스템까지 모두 기록한다. Micro-SD 의 일부 영역을 Internel SD Memory 와 Externel SD Memory 영역으로 할당하고, 타겟보드의 USB 포트와 PC와 USB 포트를 USB 케이블로 연결하고 안드로이드에서 UMS 연결을 지정하면 타겟보드는 UMS 모드로 연결된다[8,9].

2. 시험방법

접근제어 정책을 설정하고 로그를 확인하기 위하여 ADB 를 통하여 그림 6에 나타나 있는 장치제어 서비스를 확인하였다. 접근제어 정책의 설정을 위하여 각각의 값은 Jni 에 할당하고 Jni 는 소켓 통신으로 Native 서버에 전달한다.

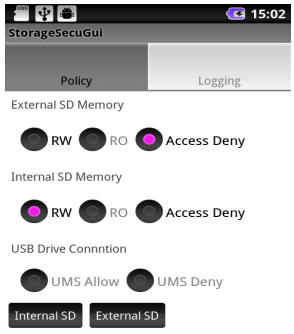

그림 6. 저장장치의 정책제어
Fig. 6. Java policy control for devices

정책 탭에서 리무버블 저장장치와 UMS 로 임베디드 장치를 이동식 디스크로 사용하는 것에 대한 정책을 설정할 수 있고 설정한 값은 Native 장치제어 서비스로 전달되어 변경된 정책이 적용된다. 안드로이드에서 Internal SD 의 경우 마운트 포인트는 /mnt / sdcard 이고 디바이스는 /dev / block / vold / 179:1이고, External SD 의 경우 마운트 포인트는 /mnt / sdcard 이고 디바이스는 /dev / block / vold / 179:1이다. 마운트 제어에 의한 것이기 때문에 마운트에 의해 디바이스는 적용이

가능하다. 보안의 종류로는 Allowance, Protection, ReadOnly 세 가지의 정책을 설정할 수 있으며 화면상에서 현재의 장치의 상태와 Mount Point 를 확인할 수 있다. 일반적으로 Internal SD 의 Mount Point 는 "/mnt / sdcard/" 이며, External SD 의 마운트 포인트는 "/mnt / sdcard / sdcard"이고, UMS 의 마운트 포인트는 임베디드 장치의 디렉토리에 연결되는 것이 아니고 USB 로 연결된 PC에서 이동식 디스크로 연결되므로 "PC Drive" 로 정했다.

3. 저장장치의 접근 제어결과

디바이스의 로깅 화면이 그림 7에 나타나 있고 장치와 그 장치가 연결된 마운트 포인트의 상태가 변경시 그 상태를 Native 서버 또는 Java 서비스로 받아 화면에 출력한다. “2010 : 06 : 16_05/29/19“는 YYYY : MM : DD_HH / MM / SS” 포맷이고, 내장 저장 장치의 경우는 “Internal Memory” 외부 저장 장치의 경우는 “External Memory” 로 표시된다. “mnt/sdcard/” 는 윈도우의 폴더에 해당하는 마운트 포인트이다[10,11].

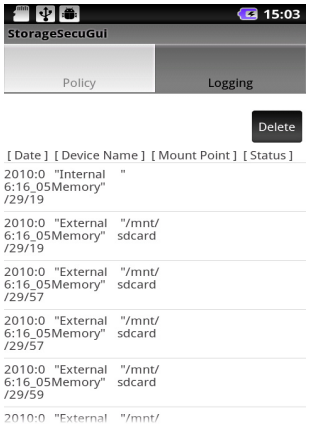

그림 7. 자바 응용프로그램의 로깅
Fig. 7. Logging of JAVA application

ADB shell 로 연결하여 mount 명령어로 console 화면을 그림 8에 출력한 것이다. 안드로이드 시스템의 기본 설정 또는 정책이 모두 허용의 상태일 경우와 동일하고, /dev/block/vold/179:1 이 SD 카드에 해당하는 디바이스 드라이버를 말하며, /mnt/sdcard 에 마운트 되어 있다는 것을 보여주고 있다. 그림 9는 External SD 카드를 차단했을 경우 상태를 보여주며, External SD 카드는 device: "/dev/block/vold/179:1" 마운트 포인트 "/mnt/sdcard" 권한 "읽고 쓰기 가능" 으로 마운트 되어 있는 상태이고, Java 어플에서 차단을 설정하였기 때문에 " 마운트 포인트 "/mnt/sdcard"가 사라졌다.

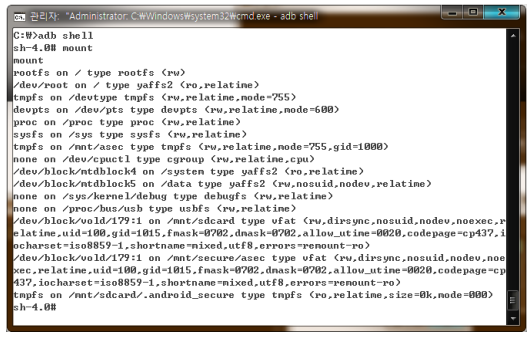

그림 8. ADB에 의한 저장장치의 접근허용
Fig. 8. Access allowance of device card by ADB

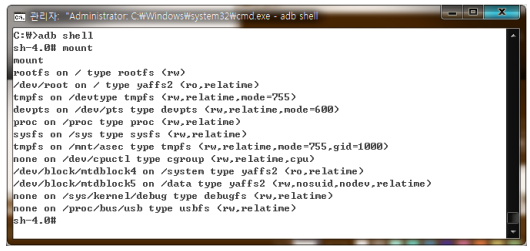

그림 9. ADB에 의한 저장장치의 접근차단
Fig. 9. Access protection of device card by ADB

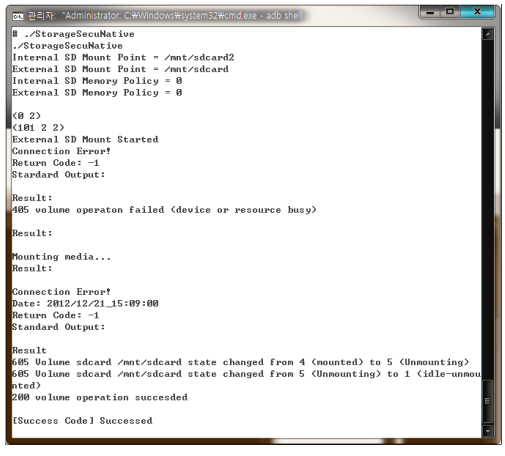

그림 10. ADB에 의한 네이티브 서버 동작
Fig. 10. Operation of native server by ADB

Native 서비스를 구동한 상태와 자바 어플이 차단을 설정한 후 Native 서비스가 마운트 되어 있던 장치를 차단하기 위해서 처리하는 상황을 그림 10에서 보여주고 있다. 또한 장치제어서비스의 구동의 상태를 보여주고 있다.

외부 저장장치가 보안 설정에 따라서 External SD 를 차단하는지 확인하는 방법으로 ADB 쉘 상에서 mount 명령어로 확인할 수 있다. UMS 보안은 PC에서 외장 디스크로 나타나는지 여부로 확인할 수 있다.

V. 결 론

본 논문에서는 안드로이드가 탑재된 임베디드 장치에서의 보안 강화를 위해서 외부 저장 장치에 대한 읽기, 쓰기 및 접근을 제한하는 방법을 제안하였다. 정책 제어 보안 모듈은 통신으로 정책을 설정할 수 있기 때문에 장치 내부에서의 정책 설정은 물론 클라이언트/서버 개념의 원격 제어로도 정책 설정이 가능하다. 따라서 원격에서 장치에 대한 보안 설정과 모니터링을 할 수 있다. 마운트 방식에 의한 제어이기 때문에 각 장치별로 보안 정책에 설정될 수 있다. 또한, 마운트에 의한 장치들은 모두 제어가 되도록 확장이 가능하다. 안드로이드 VDC를 통해서 접근제어를 하는 경우 안드로이드가 VDC 기능을 제공하는 경우에만 가능하다는 문제점이 있다.

이에 대한 해결책으로 첫째 VDC의 기능을 직접 구현한다. 둘째 System Call 단에서 구현하는 것이다. 향후 연구 방향으로는 블루투스 전송 제어, 네트워킹 파일 전송 제어, System Call 단에서의 입출력 제어로 진행하고자 한다.

References

[1] Anagha Kulkarni and Vandana Inamdaer, “CifrarFS-Encrypted File System Using FUSE”. International Journal of Computer Science and Security, Vol. 3, No. 4, pp. 295-302, 2010

[2] Seong-hwa Jeong and Tae-Jung Lho, “A Study on Implementation of Android Security System Based on SELinux”, The Korean Institute of Insustry and Technology, 2010.

[3] Soon-Seok Kwan and Young-Chan Kim, “A Study on the Android Security Kernel Module based on Mandatory Access Control“, The Korean Institute of Communications and Onformation Sciences, 2010.

[4] K. D. Kim, Y. C. Kim, J. H. Kim, A Study on Reliability Evaluation for Embedded Software", The Institute of Internet, Broading and Communication, VOL. 9 No. 3, pp.209-215, June 2009

[5] Jung-Sun Kim, Jung-Min Kang, Hyung-Hyo Lee, Development of Security Problem Definition of Android-based Operating System for Personal Handheld Devices, Journal of Korean Institute of Information Technology Vol. 10 No.11, 2012.11, pp. 1-225

[6] Seong-Jik Choi, MIn-Ji KIm, Jeong Wook Han and Byeong Gu Ahn “Android Based Mobile Student Identity Card”, Journal of The Institute of Internet, Braoding and Communication Vol.13 No.2, 2013, April., pp. 209-215

[7] Youngseok Choi, Sunghoon Kim, Dong Hoon Lee, “Study to detect and block leakage of personal information : Android-platform environment”, Journal of Korean Institute of Information Technology Vol. 23 No.4, 2013.8, pp. 757-766

[8] Seong-Hwa Jeong, Tae-Jung Lho, “A Study on Implementation of Android Security System Based on SELinux”, Journal of the Korea Academia-Industrial cooperation Society, Vol.11, No.8; 2010, pp.3005-3011

[9] Hywong-Joo Song et al., “Inside Android”, wikibooks

[10] http://developer.android.com

[11] http://www.kandroid.org/

저자 소개

유 재 만(정회원)

- 2009년 : 예원예술대학교대학원 공학석사
- 2013년 : 중부대학교대학원 정보과학과 박사

<관심분야 : 패턴인식, 인공지능, 임베디드, 모바일 등>

박 인 규(정회원)

- 1987년 연세대학교대학원 공학석사
- 1997년 원광대학교대학원 공학박사
- 1997년 ~ 현재 : 중부대학교 컴퓨터학과 교수

<관심분야 : 소프트 컴퓨팅, 러프집합>