한국산학기술학회논문지
Vol. 11, No. 8 pp. 3005-3011, 2010

# SELinux 기반 안드로이드 보안시스템 구축에 관한 연구

정성화[1], 노태정[1*]
[1]동명대학교 메카트로닉스공학과

# A Study on Implementation of Android Security System Based on SELinux

**Seong-hwa Jeong[1] and Tae-Jung Lho[1*]**
[1]**Dept. of Mechatronics Engineering, Tongmyong University**

**요 약** 최근 고성능의 스마트 폰이 속속 등장하면서 스마트 폰의 보안 문제가 대두되고 있다. 특히 오픈 플랫폼의 경우엔 더욱 바이러스의 타겟이 되기 쉬워졌다. 또한, 시만텍, 안철수연구소 등 많은 보안솔루션 업체들이 모바일 보안 시스템을 개발하고 있지만, 아직 Android 관련 보안 프로그램은 상용화된 제품이 없는 실정이다. 이러한 문제를 사전에 해결하기 위해 오픈 플랫폼 중의 하나인 Android 상에 일반 Linux에서 동작하는 SELinux를 사용할 수 있도록 환경을 구축하여 Android 시스템의 보안 기능을 개발하였으며, S3C6410 상에서 사용자 어플리케이션을 검증하였다.

**Abstract** As soon as high-performanced smart phones is rapidly emerging in recent, its security problems come to the front. Especially in case of an open platform, it is easy to be a target of virus. Many security solution industries such as Symantec and Ahnlab are developing a mobile security system, but they have not yet a commercial product. We developed the effective security function of Android system based on SELinux to solve this problem, and verified its performance by applying the user applications developed to S3C6410 board.

**Key Words :** Android Mobile Platform, Security Enhanced Linux(SELinux), Open Source Software, kernel, Security Policy, User Application

## 1. 서론

요즘 관심을 모으고 있는 스마트폰은 '손안의 PC'라고 불릴 만큼 기능이 막강하다. 특히, 응용 프로그램의 제작 및 배포가 자유롭고 다양한 콘텐츠 생성이 가능하다는 점에서 무척 환영받고 있다. 하지만, 이를 이용해 악의적인 프로그램을 제작 및 유포해서 개인정보를 습득하거나 돈을 벌려는 사람들이 나타나 많은 우려를 사고 있기도 하다. 스마트폰은 기존의 휴대폰과 다르게 PC에서 제작이 가능한 프로그램의 대부분을 만들 수 있어 악의적인 프로그램을 제작하게 되면 그 파괴력은 PC와 맞먹을 수 있기 때문이다. 현재까지 보고된 모바일용 악성코드는 약 600여종으로 대부분 심비안(Symbian) 기반의 트로이목마가 대부분이고, 초기 바이러스 및 웜 의 형태로 개발되었다가 최근에는 단말기 내의 특정 정보를 탈취하거나 금전적인 이득을 목적으로 하는 트로이목마로 빠르게 변화하고 있다.

2007년 11월에 Google이 모바일 관련 여러 개발자가 참여하는 OHA(Open Handset Alliance)라는 모임을 통해 Linux를 기반으로 하고 있는 Android라는 새로운 모바일 플랫폼(mobile platform)을 발표하였다[1-3]. 그리고 오픈 소스인 Linux kernel의 보안 강화를 위해 미국의 NSA(National Security Agency)에서 Security- Enhanced Linux(SELinux)[4-6]라는 연구 프로젝트로 진행되었으며, 2001년 1월 최초로 발표된 Linux 기반 보안 운영체제이다. 현재 Linux 쪽의 접근 제어 프로젝트 중에 가장 활발한 활동과 성능을 보이고 있다.

Android는 모바일뿐만 아니라 임베디드 환경에서도 다양하게 적용될 수 있는 플랫폼이니만큼 MID, 네비게이션, IPT 등 기존의 다른 여러 플랫폼에 얼마나 빨리 적용될 수 있는지, Android의 확장성을 이용하여 얼마만큼

*교신저자 : 노태정(tjlho@tu.ac.kr)
접수일 10년 05월 14일 수정일 (1차 10년 06월 16일, 2차 10년 06월 21일) 게재확정일 10년 08월 10일

다른 제품과 차별화할 수 있는지가 중요한 문제로 부각될 것이다.

최근 고성능의 스마트폰이 속속 등장하면서 스마트폰의 보안 문제가 대두되고 있다. 특히 오픈 플랫폼의 경우엔 더욱 바이러스의 타겟이 되기 쉬워졌다. 특히, 스마트폰은 기존의 휴대폰과 달리 PC에서 제작 가능한 프로그램의 대부분을 만들 수 있어 악의적인 프로그램을 제작하게 되면 그 파괴력은 PC의 사례보다 더 큰 위험을 불러올지도 모른다. 또한, 시만텍(Symantec), 안철수연구소(AhnLab) 등 많은 보안솔루션 업체들이 모바일 보안 시스템을 개발하여 부분적으로 사용되고 있지만, 아직 Android 마켓에는 공개되지 않은 실정이다.

이러한 문제를 해결하기 위해 오픈 플랫폼 중의 하나인 Android 상에 일반 Linux에서 동작하는 SELinux를 사용할 수 있게끔 환경을 구축하여 Android 시스템의 보안 기능을 강화하고자 하였다.

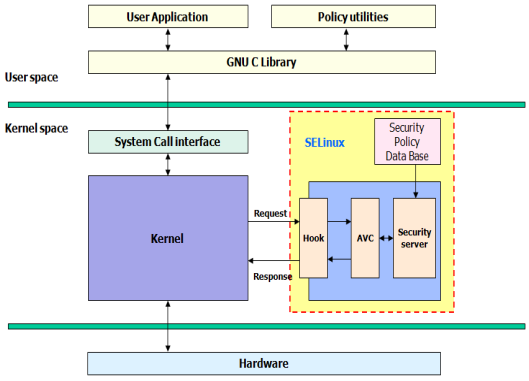
## 2. Android 시스템의 보안성

Android는 일반 Linux kernel을 Android 플랫폼 구동을 위한 부분을 제외하고는 아무런 수정 없이 사용되고 있다. 여러 Linux 배포판들은 방화벽, SELinux등을 활성화시키거나 여러 가지 보안에 관한 어플리케이션을 사용하여 시스템의 보안성을 강화하려고 하였다. 그러나 Android는 일반 Linux kernel을 사용한 만큼 보안 관련 기능이 매우 약하다.

root의 권한이 시스템에 악영향을 미칠 수 있는 가능성을 갖고 있는 만큼 이를 보호하기 위해 root의 권한을 갖더라도 모든 권한을 가질 수 없는 시스템이 필요하다. SELinux는 이렇게 root의 권한을 갖더라고 모든 권한을 가질 수 없게 하기 위해 개발되어졌다.

SElinux는 사용자가 하고자 하는 모든 동작을 network hooking의 방식과 비슷한 동작으로 access 권한을 처리하게 된다. 즉, Linux kernel 내에 발생되는 operation에 대해 SELinux가 hooking을 하여 해당 operation이 파일이나 디렉토리에 access가 가능한 것인지 아닌지 확인하게 된다.

그림 1에서와 같이 SELinux가 access 권한을 강제적으로 제한할 수 있는 가장 큰 원인은 Linux 내의 모든 파일이나 디렉토리 및 모든 process에 security context라는 것을 부여하고 이것을 security policy DB에 따로 저장하여 DB에 없는 동작은 강제로 차단하는데 있다.




[그림 1] SELinux가 적용된 시스템의 구조

SELinux는 사용하기가 매우 어렵지만 환경 설정만 잘해 놓으면 매우 뛰어난 보안 시스템이라고 할 수 있다. 기능적인 면에서의 SELinux는 외부에서의 시스템 접속을 허용하지 않는다. 가령 DDoS (Distribute Denial of Service attack: 분산서비스 거부) 공격과 같은 것을 통하여 root의 권한을 얻어 접속에 성공하였다 하더라도 내부적으로 access 권한이 security context에 따라 다르기 때문에 모든 동작에 대하여 강제로 차단시킬 수 있게 된다. SELinux는 이러한 서비스를 제공하여 사용자는 시스템 내의 중요한 정보를 보호할 수 있다.

## 3. 전체 시스템 구성

Android는 Google에서 개발한 Linux와 오픈 소스 기반의 개발 플랫폼으로서 모바일 전용 오픈소스 소프트웨어 toolkit이며, 어플리케이션은 Java로 작성해야하고 가상머신 Dalvik 위에서 실행된다.

Google은 대부분의 Android 소스 코드를 공개하였고, 이에 따라 제한 없이 Android를 임베디드시스템에 탑재하여 개발할 수 있게 되었다. Android는 완전한 개방형 환경을 지향하고 있고 시스템의 kernel 또한 개방형 OS인 Linux kernel 2.6.x를 채택하고 있다. Android 플랫폼의 가장 아래 부분이 OS를 담당하고 있는 Linux kernel이고, 이 위에 여러 라이브러리와 Android 런타임(run time)이 존재한다. 그 위에 어플리케이션 프레임워크와 중요 어플리케이션들로 이루어진다.

Linux는 이미 수많은 하드웨어에 이식된 바 있는 검증된 OS로, Android는 하드웨어나 주변기기에 대응하기 위하여 Linux kernel을 사용한 메모리 관리, 프로세스 관리 등의 H/W 종속적인 작업을 수행한다.

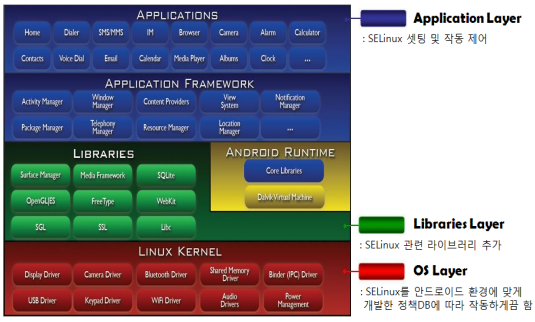
우리나라 모바일 시장에서 표준화된 어플리케이션 개

발을 위해 위피(WIPI)를 이용하고 있는 것처럼 Android 도 이와 유사하게 virtual machine 방식의 런타임 계층을 따로 두고 있다. Android의 가장 큰 특징 중의 다른 하나가 바로 모든 어플리케이션이 자바로 개발된다는 점이다. Android는 모바일 환경에서의 성능 최적화를 위해 자체적으로 Dalvik이라는 JVM (Java virtual machine)을 개발하였다. Android는 Dalvik이라는 런타임 환경을 통해 더 확실하게 HW와 SW를 계층화 하고 있다. 표준 위피가 탑재된 어떠한 단말에서도 동일한 어플리케이션을 실행할 수 있는 것과 마찬가지로 Android도 Dalvik이 동작하는 모든 단말기에서 Android 어플리케이션이 동일하게 실행가능하다.

Android는 오픈 플랫폼답게 여러 오픈 소스 등을 결합하여 편리한 어플리케이션 개발도구를 제공하고 있다 [6,7]. Android 어플리케이션은 자바로 개발되기 때문에 자바 통합툴로 잘 알려져 있는 이클립스(Eclipse) 개발환경을 이용하게 된다. SDK를 설치하면 이클립스 플러그인(Eclipse Plug-in) 형태로 Android 전용 어플리케이션을 개발하여 Android 에뮬레이터 등 여러 개발 도구들과 자동으로 연결할 수 있게 된다.

Android 전체 소스를 받아서 시스템을 빌드하기 위해서는 몇 가지 시스템의 제약으로서 Linux 상에서 개발도구를 갖추고 있어야 하지만, 일반 어플리케이션 개발을 위해서는 이클립스와 JDK만 설치되면 되기 때문에 Windows XP/Vista, Linux, MacOS 등 대부분의 운영체제에서 쉽게 사용할 수 있다.

그림 2와 같이 전체 시스템을 설계하였으며, Android의 OS Layer에서 SELinux를 활성화(enable) 시키고, Android 환경에 맞게 개발한 policy DB에 따라 작동할 수 있도록 한다. 또, 라이브러리 계층에서 SELinux 관련 라이브러리를 추가하고, user application을 이용해 손쉽게 SELinux를 셋팅 및 작동 제어할 수 있는 환경을 개발하였다.




[그림 2] 전체 시스템 구성도

Android 플랫폼에서 Linux kernel 단에 존재하는 SELinux를 사용할 수 있는 환경을 구축하여 시스템 관리자와 보안 관리자의 기능을 분리하고, 슈퍼 유저라도 시스템의 중요한 파일 및 디렉터리, 그리고 프로세스에 함부로 접근할 수 없도록 한다. 또 접근권한이 없는 subject가 object에 access하는 것을 차단할 수 있도록 한다.

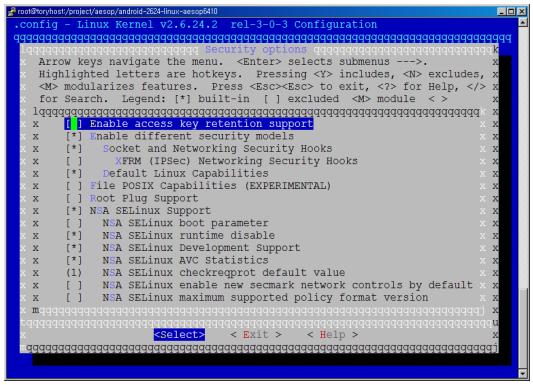
## 4. 시스템 구현

### 가. SELinux 환경 구축

기존의 Linux, Unix 계열 인증과 권한 시스템은 해커가 시스템의 취약한 부분을 공격하여 root 권한을 획득하면, 해당 시스템의 모든 자료는 물론, 모든 데몬(demon)에 대한 실행권한이 주어지면서 해커에 의해 시스템이 파괴될 위험이 있다. 이러한 문제점을 해결하고 오픈 소스인 Linux kernel의 보안을 강화시키기 위해 미국의 NSA에서 SELinux를 개발하게 되었다. 특정 데몬의 버그를 통해 root 권한을 획득하더라도 다른 데몬이나 시스템에는 접근할 수 없도록 하여 시스템의 보안을 강화하였다.

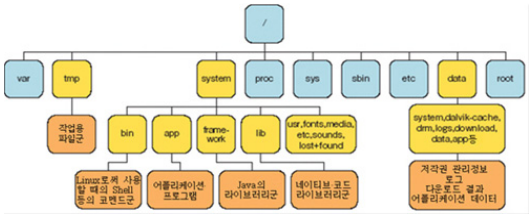
Linux kernel 2.6 이상 버전에는 기본적으로 SELinux가 들어가 있기 때문에, Android kernel 안에도 존재하지만 Android에서는 SELinux를 사용하지 않기 때문에 Android의 Linux kernel에서도 SELinux가 활성화 되어있지 않다. Android에서는 kernel에 존재하는 SELinux를 사용하지 않기 때문에, 어떤 어플리케이션이 시스템 콜을 호출하면 UID, GID와 같은 기본적인 허가만 확인을 하고, 요청을 허용해버린다.

SELinux 환경을 구축하기 위하여 우선 그림 3과 같이 SELinux kernel의 configuration을 설정하였다.




[그림 3] Security option setting

에뮬레이션 환경에서는 PC에서 동작하는 ADB(Android Debug Bridge) 툴을 사용해 ADBD 프로세스와 통신함으로써 Linux의 쉘을 조작할 수 있다. 쉘 조작으로 이용하는 ls나 mv 등의 명령어는 일반 Linux의 경우 busybox를 통해 관리하지만, Android에서는 그림 4와 같이 Google에서 자체적으로 제작한 toolbox를 이용하고 있다.




**[그림 4]** Android toolbox 파일시스템 구조

그림 5와 같이 Linux kernel 안에 있는 SELinux를 사용하기 위해서는 busybox안의 SELinux 관련 유틸리티를 이용할 수 있어야 한다. SELinux를 활용하여 Android 보안환경을 구축하기 위해 busybox를 Android 환경에 맞게 static build하여 Android 시스템에 포함시켰다.


**[그림 5]** 보안 Application용 Root FS reorganization

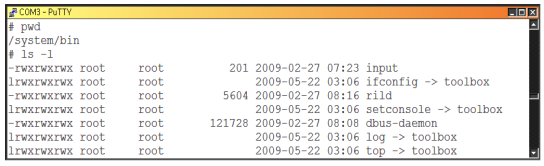
Android에는 GPL 기반 라이브러리가 없기 때문에 Android 상에서 구동되지 않게 되므로 이들 각각은 모두 static build를 해줘야 하고, 이를 위해서는 타겟용으로 build하기 위해 사용되는 크로스컴파일러(cross compiler) 안에 SELinux 및 방화벽 관련 동작을 위한 라이브러리를 모두 build해서 포함시켜주어야 한다. 따라서 busybox configuration은 아래와 같이 설정하였다.

```
Busybox Settings  --->
General Configuration  --->
[*] Support NSA Security Enhanced Linux(이 부분을 체크해야 SELinux Utilities 옵션이 나타남.)
```

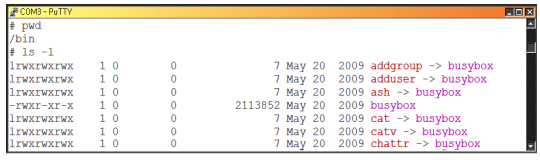
```
SELinux Utilities ->          [*] runcon
[*] chcon                     [*] Enable long options
[*] Enable long options       [*] selinuxenabled
[*] getenforce                [*] setenforce
[*] getsebool                 [*] setfiles
[*] load_policy               [*] Enable check option
[*] matchpathcon              [*] setsebool
[*] restorecon                [*] sestatus
```

```
Busybox Settings  --->
Build Options  --->
[*] Build BusyBox as a static binary
    (no shared libs)
```

그림 6, 그림 7은 각각 toolbox, busybox로 ls -1 실행한 결과이며, 그림 8은 타겟 board에 보안관련 utility 디렉토리 생성 결과이다.




**[그림 6]** toolbox로 ls -1 실행 결과



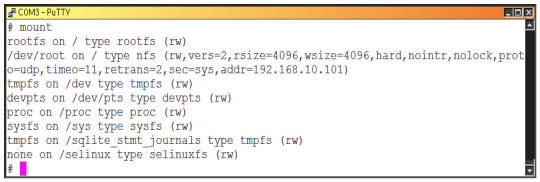

**[그림 7]** busybox로 ls -1 실행 결과


**[그림 8]** 보안관련 utility 디렉토리 생성 결과

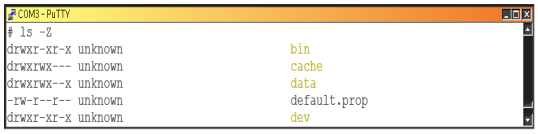
SELinux 관련 라이브러리의 설치는 아래와 같다.

```
libsepol-1.16.14.tar.gz (181KB)
libselinux-1.34.15.tar.gz (130KB)
libsemanage-1.10.9.tar.gz (154KB)
checkpolicy-1.34.7.tar.gz (56KB)
bison-2.3.tar.gz (6,920KB)
flex-2.5.35.tar.gz (1,423KB)
selinux package download :
http://userspace.selinuxproject.org/trac/wiki/Releases
```

/nfsboot/nfsboot.Android/init.rc파일에 SELinux 마운트 시키는 shell명령 추가는 mount selinuxfs none /selinux noauto 이며, 그림 9, 그림 10은 각각 mount 상태와 보안 허가 확인을 나타낸다.



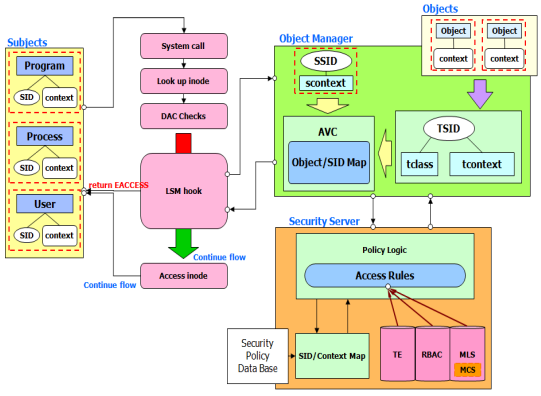

[그림 9] SELinux mount 상태 확인




[그림 10] List security context and permission

## 나. SELinux 작동 테스트

그림 11에서와 같이 subject가 object에 access하기 위해서는 subject가 갖고 있는 security context가 object의 security context에 대하여 access 권한을 갖고 있는지 security policy DB에서 확인하고 권한이 있으면 access할 수 있지만 만일 권한이 없으면 access할 수 없게 된다.




[그림 11] SELinux의 동작 원리

이로써 아무리 모든 권한을 다 가질 수 있는 root라도 root에 부여된 security context가 특정 object의 security context에 대한 권한이 DB에 없다면 root는 모든 동작을 다 할 수 없게 된다.

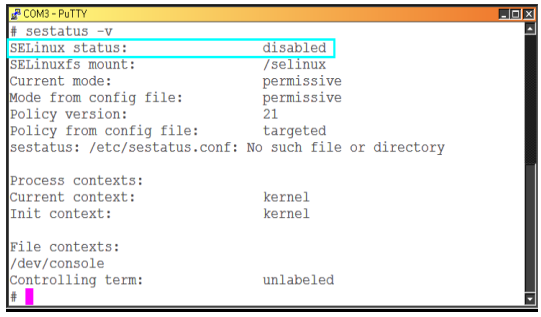
모든 subject와 object(사용자, 프로그램, 프로세스 및 이들의 동작 대상인 파일과 디바이스를 포함한 시스템 전체)에 대한 access 허가를 포함한 패키지로 페도라(Fedora)에서 사용가능한 policy에는 strict, targeted가 있다.

페도라 코어에서 SELinux policy로서 strict policy를 적용함으로 인해 발생하게 된 여러 가지 문제점 때문에 현재에는 보다 완화된 policy 패키지 targeted policy가 설치할 때 기본으로 제공되고 있다.

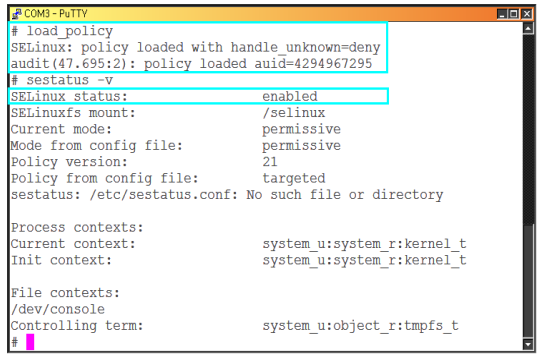
일반 사용자들이 SELinux를 사용하기 위해서는 수준 높은 전문기술이 필요하다. targeted policy는 자주 문제시되는 부분들만 우선적으로 적용시키고, 나머지는 표준 Linux 보안과 동일하게 운영되도록 적용한 policy이다.

현재, targeted policy에서는 dhcpd, httpd (apache.te), named, nscd, ntpd, portmap, snmpd, squid, syslogd 데몬을 관리한다. 이들에 대한 policy 파일은 /etc/selinux/targeted /src/policy/domains/program에서 찾을 수 있다.

그림 12는 SELinux 작동 테스트를 위한 policy 로드 설정이며, 그 결과는 그림 13과 같다.




[그림 12] SELinux disabled

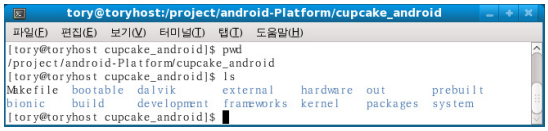



[그림 13] SELinux policy 로드

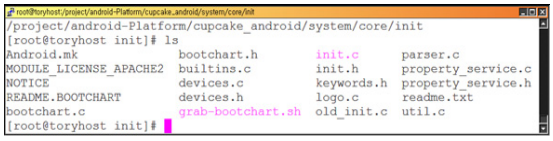
policy DB를 Android 타겟 보드로 이동한 결과는 다음과 같다.

```
[root@toryhost refpolicy]# pwd
  /project/selinux/refpolicy
[root@toryhost refpolicy]# make
[root@toryhost refpolicy]# cp policy.21 /nfsboot/
nfsboot.Android/system/etc/selinux/targeted/policy/
```

그림 14는 Android cupcake 1.5 platform의 full source를 다운로드 후 build한 것이고, 그림 15는 Android 플랫폼 안에 존재하는 init.c source이다.




**[그림 14]** Android cupcake 1.5 platform full source 다운 후 build



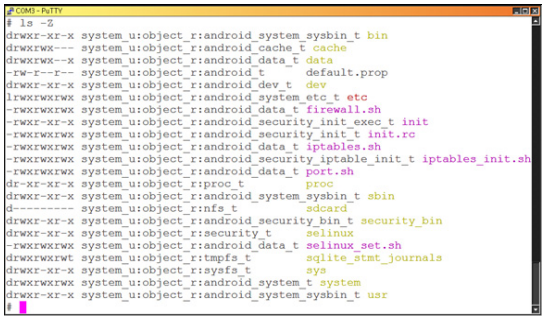

**[그림 15]** Android platform 내에 존재하는 init.c

어플리케이션에서 관리하는 보안 스크립트를 부팅될 때 적용되도록 하기 위하여 Android가 부팅될 때 수행되는 init 프로세스에 스크립트가 수행이 되도록 selinux_setting() 함수를 추가하였다.

user application에서 보안설정을 변경하여 스크립트의 내용이 달라졌을 때, 달라진 내용이 적용되도록 하기위해 SELinux setting demon을 kernel 상에서 돌려준다.

Android에서 SELinux를 사용하기 위해 관련 유틸리티들을 필요로 하는 라이브러리와 함께 빌드를 해서 타겟에 올린 후, kernel에서 비활성화 되어있는 SELinux를 활성화시키고, Android 환경에 맞게 보안 policy DB를 구축하였다.

policy DB에 Android 환경에서 사용될 policy DB를 android.te 라는 이름으로 생성한 후 Android 환경에 맞게 context를 새롭게 정의하고 각각의 context에 대한 접근권한을 "allow [컨텍스트] [대상] {접근권한}"의 형태로 추가하였다.




**[그림 16]** SELinux policy 적용 확인

그림 16과 같이 Android 시스템에 SELinux policy가 적용된 것을 확인할 수 있다. sdcard를 제외한 나머지 디렉터리에 접근권한이 없는 어떠한 subject도 object에 대해 임의로 접근하는 것을 막을 수 있다.

## 다. 보안 어플리케이션 구현

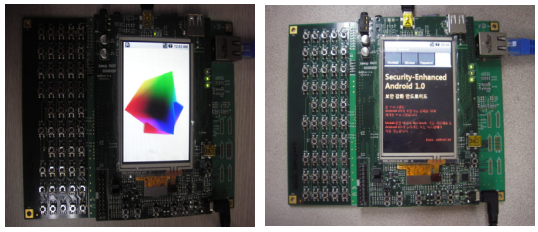
Java로 제작한 SELinux 설정 어플리케이션이다. 기존의 x86계열의 Linux에서 적용하고 있는 보안 policy에 비해 비교적 간편하고 보다 쉽게 SELinux policy를 시스템에 적용할 수 있도록 제작하였다. 그림 17, 18, 19과 같이 Application단에서 SELinux를 활성화 또는 비활성화시킬 수 있고, 접근레벨을 조정할 수 있다.


**[그림 17]** SELinux 활성화


**[그림18]** policy적용


**[그림19]** 접근레벨 조정

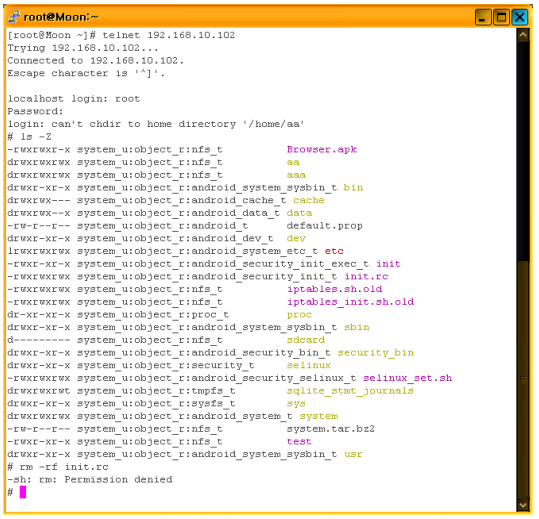
그림 20은 User Application을 S3C6410 Board[8] 상에서 구동한 결과를 나타낸다.




**[그림 20]** S3C6410 Board상에서 User Application의 구동

## 라. 보안 테스트

앞에서 SELinux를 적용하게 되면 root의 권한이라도 강제적으로 동작을 차단 할 수 있다고 하였다. 따라서 그림 21은 telnet으로 Android시스템에 접속하여 root의 권한으로 로그인하였으나 SELinux의 권한 때문에 아무런 동작을 할 수 없다는 것을 보여 준다.




**[그림 21]** SELinux 적용 후 root로 telnet 접속시 파일 삭제 실패 확인

## 5. 결 론

Android 플랫폼은 스마트폰 뿐만 아니라 Net-book PC나 TV 셋톱박스, 디지털 액자, 노래방기기, 유선전화기, 의료기기, 그리고 레스토랑 전자 메뉴판 등 주변의 거의 대부분의 전자제품 속에 들어갈 가능성을 갖고 있고, 이들은 가정의 홈 네트워크로 연결되어 유기적으로 작동될 수도 있기 때문에 Android의 보안은 한 층 더 강화될 필요성이 있다.

이에 대하여 SELinux를 이용한 Android 시스템 보안 체계 구축을 목표로 하였고, 기존 Linux에서 사용하고 있는 보안 기능들을 Android에 적용함으로써 시스템의 활용성 및 효율성, 그리고 호환성을 높일 수 있도록 S3C6410 Board상에서 구동 가능한 시스템을 개발하였다.

## 참고문헌

[1] 김정훈, “구글의 안드로이드 프로그래밍”, 성안당, 09.

[2] 권오철, 김주성, 이창건, 하은용, “안드로이드 모바일 플랫폼에서의 사용자 인터페이스 자동전환 기술의 개발”, Proc. of KIISE2009 Fall Conference, Vol.36, No.2(B), pp.436-440, 2009.

[3] 오용석, 박찬익, “안드로이드 기반 모바일 플랫폼을 위한 새로운 NAND 플래시 메모리 파일 시스템”, Proc. of KIISE2009 Fall Conference, Vol.36, No.2(B), pp.388-393, 2009.

[4] Frank Mayer, Karl Macmillan, David Caplan, “SELinux by Example: Using Security Enhanced Linux”, Peachpit Press, 2007.

[5] James Morris, "File System Labeling in SELinux", Redhat, 2004.

[6] M.Tim Jones, "Anatomy of Security-Enhanced Linux(SELinux)", IBM, 2008.

[7] 문일헌 외, “.NET Compact Framework를 활용한 SOA 환경 기반의 임베디드 RFID 미들웨어 시스템 설계 및 구현”, 한국산학기술학회논문지, Vol.9, No.6, pp.1639-1646, 2009.

[8] 이솝보드 개발자, “Aesop-6410 Board Manual”, 이솝보드, 2009.

**노 태 정**(Tae-Jung Lho) **[정회원]**

- 1984년 2월 : 부산대 기계설계학과 (공학학사)
- 1986년 2월: KAIST 생산공학과 (공학석사)
- 1992년 2월: KAIST 정밀기계공학과 (공학박사)
- 1986년 2월 ~ 1999년 2월 : 삼성중공업 기전연구소(수석연구원)
- 1999년 3월 ~ 현재 : 동명대학교 메카트로닉스공학과 부교수

<관심분야>
IT융합 Mechatronics, Robotics, 제어.자동화, LCD물류반송 자동화, 태양광발전 Module 제조장비 등

**정 성 화**(Sung-hwa Jung) **[준회원]**

- 2008년 7월 ~ 2010년 2월 : 삼성전자 소프트웨어 멤버십 18기
- 2010년 2월 : 동명대학교 메카트로닉스공학과 (공학학사)
- 2010년 2월 ~ 현재 : 삼성전자 반도체 사업부(메모리)

<관심분야>
Robotics, Embedded System, 정보보안, 응용 Software 등