제36회 한국정보처리학회 추계학술발표대회 논문집 제18권 제2호 (2011.11)

루팅의 증가로 인한 위험과 대책에 관한 연구

정우준*, 김호연**, 박민우**, 정태명***
*성균관대학교 전자전기컴퓨터공학과
**성균관대학교 정보통신공학부
e-mail :*sarradin@skku.edu, **{hykim, mwpark}@imtl.skku.ac.kr,
***tmchung@ece.skku.ac.kr

A Study on Risks and Safety Measures in Increased Rooting of Mobile OS

Woo-Joon Joung*, Ho-Yeon Kim**, Min-Woo Park**, Tai-Myoung Chung***
*Dept. of Electrical and Computer Engineering, Sungkyunkwan University
**School of Information Communication Engineering, Sungkyunkwan University

요 약

최근의 스마트폰 제조업체는 유저들의 루트 권한을 제한하고 있다. 하지만 많은 유저들은 루팅 프로그램을 통해 루트 권한을 획득해 스마트폰을 사용하고 있다. 또한 임의의 공격자가 배포한 악의적 강제루팅 프로그램에 의해 루트 권한이 탈취되는 경우도 증가하고 있다. 따라서 루팅 프로그램 자체의 문제, 루트 권한을 획득해 생기는 문제 등 많은 보안상의 문제가 발생하고 있다. 본 논문에서는 데스크톱의 Linux에서 사용되고 있는 SELinux를 통해 이런 문제를 보완하는 방법을 제안하고자 한다.

1. 서론

가트너의 보고서에 따르면 2010년 스마트폰 OS의 보급 대수는 약 3억대로 전년대비 72.1%의 판매상승을 보였다[5]. 이러한 스마트폰의 판매현황과 더불어 스마트폰의 보안위협 또한 증가하고 있다. 유통되고 있는 대부분의 스마트폰OS의 경우 UNIX를 기반으로 하여 모바일 OS로 사용하고 있다. UNIX의 경우 루트권한을 획득하지 못한다면 악성코드가 사용할 수 있는 파일의 권한이 없어 일반적으로 모바일OS의 경우 악성코드로부터의 안전지대라고 불렸다. 하지만 사용자가 마음대로 커스터마이징을 할 수 없는 불편함이 있었다. 루트 권한이 없어 생기는 문제를 해결하기 위해 사용자가 몰래 루트 권한을 획득하도록 해주는 경우가 늘어나고 있다. 최근 미국에서는 이런 루팅이 “합법적으로 입수한 프로그램의 사용”의 목적이라면 합법이라고 판결을 내리고 있다. 또한 사용자의 의도와는 상관없이 단말을 원격에서 루팅해 악성코드를 설치하거나 시스템 영역을 변조하는 악성 애플리케이션이 만들어져 배포되고 있다. 이렇게 루트 권한을 획득한 경우 모바일 OS가 악성코드 안전지대라고 할 수 없다. 기존에는 이런 루팅을 막기 위해 스마트폰의 취약점을 이용한 루팅 프로그램이 발견되면 그 취약점을 막아 루팅 프로그램을 무력화하는 방법을 사용하고 있다. 하지만 이렇게 루팅하는 방법을 막기만 하는 것으론 한계가 있을 뿐만 아니라 이미 루팅을 한 사용자를 위한 대책으로는 부족하다. 또한 루팅을 원하는 사용자가 새로운 취약점을 찾아 루팅을 하는 것을 막을 수 없는 문제점이 발생하게 되었다. 따라서 본 논문에서는 루팅을 원하는 사용자에게 기존 데스크톱에서 사용되고 있는 SELinux를 사용해 안드로이드의 보안 기능을 강화해서 루트 권한을 사용 할 수 있도록 해 루팅의 문제점을 해결하는 방법을 제안하고자 한다.

본 논문의 구성은 다음과 같다. 2장에서는 관련연구로써 루팅의 특성과 악의적 루팅의 문제, 그리고 기존의 SELinux에 대해 알아보고, 3장에서는 SELinux 사용 시의 장단점과 SELinux의 적용 방법에 대해 알아본다. 마지막으로 4장에서는 본 논문의 결론과 향후 연구 계획을 소개한다.

2. 관련연구

2.1 스마트폰 루팅의 특성

스마트폰의 루팅이란 루트 권한을 가지지 못하게 출시되고 있는 안드로이드 환경의 스마트폰에서 루트 권한을 획득하는 것이다. 데스크톱 환경에서는 사용자가 입력한 비밀번호로 슈퍼유저 권한을 획득할 수 있지만 안드로이드 환경에서는 일반적인 방법으론 슈퍼유저권한을 획득할 수 없다. 하지만 루팅은 유저 권한만 가진 사용자가 슈퍼유저 권한을 획득할 수 있도록 해준다. 루팅은 주로 부트로더를 해제하지 않고 우회하는 방법을 통해 이루어진다.

루팅을 하기 위한 방법으로 <표 1>의 세 가지 방법이 주로 사용되고 있으며, 직접 쉘 프로그래밍을 통해 하는 방법이 아닌 프로그램을 통해 하는 방법은 단순한 클릭 한번으로 쉽게 루팅이 가능하다.

루팅은 편의상 <표 2>과 같이 3단계로 구분할 수 있는데 이 구분법은 SuperOneClick의 개발자인 CLShortFuse에 의해 알려져 있다[6].

<표 1> 루팅의 방법

방 법	세부 방법
Z4root	안드로이드 어플리케이션을 이용한 방법
SuperOneClick	데스크톱 프로그램을 이용한 방법
안드로이드 Debug Bridge	직접 루팅(안드로이드 SDK의 tools/ platform-tools 패키지를 설치하여 활용)

<표 2> 루팅의 구분

구 분	세부 내용
쉘 루트 (shell root)	슈퍼유저 권한을 얻었지만 /system 디렉토리의 수정이 불가능한 상태
임시 루트 (temporary root)	슈퍼유저 권한을 얻고 /system/bin/ 디렉토리에 su가 설치되었지만 재부팅 시 권한이 상실되는 상태
풀 루트 (full root)	슈퍼유저 권한을 얻고 /system/bin/ 디렉토리에 su가 설치되었으며 지속적으로 권한이 유지되는 상태

2.2 악의적 루팅의 증가

일반적으로 안드로이드 사용자가 슈퍼유저 권한을 가지고 있지 않는다면 보안상 큰 문제점이 발생하지 않는다. 하지만 사용자가 직접 루팅을 한 경우 루트 권한을 가짐으로 인해 보안상 문제점이 생길 뿐만 아니라 루팅을 하는데 사용한 프로그램의 비밀번호를 따로 변경하지 않고 사용하는 경우가 많아 이를 통한 침입이 가능해진다. 그리고 불법 애플리케이션에 의한 루팅을 통해 슈퍼유저 권한을 획득하게 되면 타인이 자신의 스마트폰을 컨트롤할 수 있게 되어 보안성을 보장할 수 없게 된다. 이런 안드로이드의 취약점을 이용해 불법적으로 루트 권한을 획득하는 악성 애플리케이션이 많이 발견되고 있다. 그중 하나로 진저브레드(안드로이드 SDK 2.3버전)의 취약점을 이용한 애플리케이션이 있다. 이 악성 애플리케이션은 정상 애플리케이션에 대한 재패키징 형태로, 안드로이드 마켓에서는 유포되지 않았지만 블랙마켓, 3rd party 마켓 등에서 유포되고 있다. (그림 1)은 루트 권한을 얻기 위해 진저브레드의 취약점을 이용하는 코드이다[7].

```
obj = Android.os.Build.VERSION.RELEASE;
if(!file.exists() || ((string) (obj)).compareTo("2.3.4") > = 0)
                                                    …… (1)
                          :
 process = Runtime.getRuntime().exec(s3);    …… (2)
                          :
 int k3 = Runtime.getRuntime().exec(s11).waitFor();
                                                    …… (3)
                          :
```

(그림 1) 주요 루팅 함수

(1)을 통해 현재 스마트폰의 안드로이드 버전을 체크하고, 안드로이드 SDK 2.3 버전일 경우 (2)를 통해 루트 권한 획득을 시도한다. 이 후 (3)을 통해 root shell을 인스톨한다.

해당 악성 애플리케이션은 내부 패키지에 포함된 gbfm.png 파일을 이용하는데 이 파일은 png파일로 위장된 elf파일이며, 이 파일을 이용한 쉘 명령으로 루트 권한 획득이 가능해진다.

그밖에 DroidKungFu이라 불리며 안드로이드폰을 강제 루팅 후 권한을 탈취하여 좀비화시키는 악성 애플리케이션이 발견되었다. 이 애플리케이션 역시 정상 앱을 리패키징하여 재배포되었고, 중국의 마켓과 포럼에 유포되었다. 이 악성 애플리케이션이 설치되면 doSearchReport()와 updateInfo() 함수를 통해 스마트폰의 다양한 정보들을 수집한다. 수집되는 정보는 IMEI, ostype, osapi, model, SDKVersion, SDcard info, internal Memory Size, Net operator, phone number, running service 등이며, 특정 URL로 유출된 정보를 전송하게 된다. 이렇게 정보를 수집한 후 전송한 뒤, 악성 애플리케이션은 (그림 2)의 getPermission() 함수를 통해 루팅을 시도한다[8].

```
private void getPermission()
{
  if          ((Settings.Secure.getInt(getContentResolver(),
"adb_endable", 0) == 0) && (setUsEnabled() >= 10))
    mPermState = 0;
int I = mPreferences.getInt("P3", 0);
                          :
SharedPreferences.Editor localEditor1 = mPreferences.edit();
SharedPreferences.Editor          localEditor2          =
localEditor1.putInt("P3", j);
                          :
}
```

(그림 2) getPermission()함수

루팅이 성공하게 되면 cplegacyres() 함수를 통해 'assets' 에 저장되어 있던 다른 악성 애플리케이션을 설치하게 되는데, 이 악성 애플리케이션은 구글의 'Google Search'를 위장해 'Google SSearch'이라는 이름을 사용한다. 이 애플리케이션은 서비스 형태로 루트 권한을 받아 동작하며, 설치된 사용자의 스마트폰을 원격통제(좀비화)하는 기능을 수행한다.

이 외에도 Failing Down, Hot sexy video, super ringtone maker, super gitar solo 등 여러 악성 애플리케이션이 안드로이드 폰을 강제로 루팅시키는 것으로 밝혀졌다. <표 3>은 [8]에서 확인된 강제로 루팅시키는 악성 애플리케이션의 개발자와 목록으로, 이외에도 계속 악성 애플리케이션이 증가하고 있다.

<표 3> 악성 애플리케이션의 개발자와 목록들

개 발 자	목 록
Magic Photo Studio	Sexy Girls: Hot Japanese Sexy Legs HOT Girls 4 Beauty Breasts Sex Sound Sex Sound: Japanese Mango Studio
Floating Image Free	System Monitor Super StopWatch and Timer System Info Manager E.T. Tean
Call End Vibrate	BeeGoo

이렇게 악성 애플리케이션이 설치된 스마트폰은 IMEI/IMSI 등의 기기정보, 전화번호, /proc/cpuinfo를 통한 프로세스, CPU타입, 모델, 제조사 등의 정보를 수집하고, 수집한 정보를 외부로 유출한다. 또 루팅을 위한 SDK 버전 정보를 획득해 강제 루팅으로 루트권한을 획득, 추가적인 apk 파일 다운로드 및 설치를 시도한다. 이런 악성 애플리케이션은 보안업체에 의해 발견되기 전까진 구글 안드로이드 마켓을 통해 유포되어 확산되고, 발견되어도 특별한 관심이 없는 사용자의 경우 빠른 대처가 어렵기 때문에 이에 대한 대책이 없다면 쉽게 외부에서 루트 권한을 획득해 좀비화할 수 있다.

2.3 SELinux의 개요

기존의 Linux, Unix 계열의 인증과 권한 시스템은 해커가 루트 권한을 획득하면 해당 시스템의 파일과 디렉토리, 프로세스 및 모든 데몬에 대한 실행권한이 주어지게 된다. 이런 문제를 보완하기 위해 SELinux가 개발되었다.

SELinux 정책은 사용자, 프로그램, 프로세스 그리고 이들의 동작 대상인 파일과 디바이스를 포함한 시스템 전체, 즉, 모든 주체와 객체에 대한 접근 허가(access permissions)를 포함한 패키지를 말하며 페도라에서 사용가능한 정책 패키지는 strict , targeted 두 가지가 있다.

SELinux를 사용하면 프로그램 실행에 대한 세세한 곳까지 제어가 가능하다는 장점이 있지만 일반적인 리눅스식 자유재량 접근 제어 방식에 비해 설정하기가 까다롭고 새로운 개념을 익혀야 한다는 단점이 있다. 다음의 <표 4>는 SELinux의 세 가지 기본 설정으로 사용자가 원하는 설정을 적용해 사용할 수 있다.

안드로이드는 일반적인 데스크톱에서 사용되는 Linux 커널을 플랫폼 구동을 위한 부분만 추가한 채 거의 그대로 사용하고 있다. 하지만 Linux가 방화벽이나 SELinux같은 여러 보안 대책을 사용하는데 반해 안드로이드에서는 루트

<표 4> SELinux의 기본설정

이 름	세부 내용
disabled	◆ SELinux 보안 제어를 사용하지 않는 경우 disabled옵션을 선택 ◆ 보안 제어 기능을 끄고 시스템이 보안 정책을 사용하지 않도록 설정 ◆ 시스템 부팅시에 부트로더의 파라미터로 selinux=0 으로 설정하고 부팅하는 것과 같은 설정
permissive	◆ 서비스 거부 메시지를 통보 ◆ 자료와 프로그램에 이름을 할당한 후 로그를 기록하지만 보안 정책을 사용하지는 않음 ◆ 경고 옵션을 선택 시 가끔씩 보안경고 대상이 아닌 것을 경고 대상으로 탐지하는 오류(false positive)나 경고 대상인 것을 탐지하지 않는 오류(false negative)가 발생할 가능성도 있어 주의가 필요
enforcing	◆ SELinux를 완전히 활성화하는 옵션 ◆ 추가 시스템 보안을 위해 모든 보안 정책 (허가가 없는 사용자가 특정한 파일이나 프로그램에 접근하는 것을 거부)이 사용 ◆ SELinux가 완전히 실행되어도 아무런 지장을 받지 않고 일반적인 시스템 작업을 수행할 수 있다고 확인이 되는 경우 이 옵션을 설정

권한을 허용하지 않는 것 외에는 특별한 보안 대책을 사용하지 않고 있다. 하지만 사용자가 루팅을 하거나 강제로 루팅을 시키는 악성 애플리케이션이 설치되 루트 권한을 획득한 안드로이드 스마트폰은 보안에 취약해지게 된다. 이런 안드로이드의 보안성을 강화하기 위한 방법으로 루트 권한을 가진 상태에서도 시스템에 악영향을 줄 수 없게 해주는 SELinux를 안드로이드에도 적용하는 방법을 생각해보았다.

SELinux는 사용자가 하고자 하는 동작을 네트워크 후킹과 비슷한 동작으로 액세스 권한을 처리한다. 발생하는 오퍼레이션을 SELinux를 통해 후킹을 하여 해당 오퍼레이션이 파일이나 디렉토리에 액세스가 가능한지를 검사한다. 이를 위해 Linux내의 모든 파일과 디렉토리, 프로세스에 보안 설정을 부여하고 이것을 따로 DB에 저장하여 이 DB에 없는 동작을 강제로 차단하게 된다. 그 결과 SELinux는 외부에서의 접속을 허용하지 않는다. 만약 루팅을 통해 루트 권한을 얻었다 하더라도 내부적으로 액세스 권한을 가질 수 없기 때문에 동작을 강제로 차단시킬 수 있게 된다.

3 안드로이드 환경에서의 SELinux

3.1 SELinux사용에 예상되는 장단점

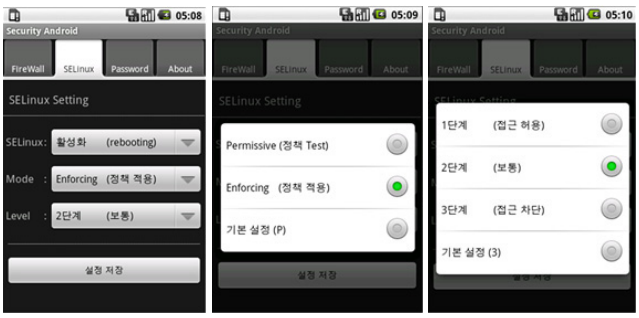
현재 안드로이드를 루팅해 사용하는 사람들은 대부분 루팅하지 않는 경우가 보안상 더욱 안전하다는 것을 알고 있지만, 스마트폰을 좀 더 유연하게 사용하기 위해 위험을 감수하고 있다. 이렇게 루팅을 하게 된 스마트폰은 인터넷 뱅킹 등 몇몇 중요한 애플리케이션에 보안상의 이유로 접속이 불가능하게 된다. 하지만 SELinux가 안드로이드 환경에 적용된다면 이런 취약점을 감수하지 않아도 안전하게 루팅을 할 수 있게 된다. 따라서 루팅을 한 상태에서도 여러 보안이 중요한 애플리케이션에 접근할 수 있게 된다. 보안상의 문제점이나 루팅하면 사용하지 못하는 애플리케이션 때문에 루팅을 하지 않았던 사용자들도 마음 놓고 루팅을 통해 스마트폰을 커스터마이징 할 수 있다. 또한 예전에 발견되던 악성 코드들이 주로 배터리 소모 공격이나 URL을 통한 피싱 프로그램 위주였다면 최근에 발견되는 악성 애플리케이션은 대부분 안드로이드의 취약점을 공략해 강제 루팅을 시도하는 형태인데, SELinux를 사용한다면 이런 악성 애플리케이션에 대비할 수 있게 된다.

SELinux를 사용한다는 것은 추가적인 리소스와 DB를 사용한다는 것이므로 성능상의 저하가 발생하게 된다. 하지만 데스크톱에서 SELinux의 성능 저하를 크게 신경 쓰지 않는 것처럼 스마트폰의 성능도 꾸준히 발전하고 있으므로 지금 당장은 문제가 될지 모르지만, 곧 충분히 감안할 수 있는 성능의 스마트폰이 나오면 해결될 것이다. 또한 SELinux는 중요 파일이나 디렉토리에 추가 권한을 설정해 줘야하기 때문에 일반 사용자가 사용하기에 불편하다는 문제점이 발생한다. 하지만 이런 문제는 애플리케이션 단계에서 자동으로 기존의 루트 권한을 막아둔 부분에 추가 권한을 설정할 수 있도록 해주면 해결할 수 있을 것이다.

3.2 안드로이드에 적용된 SELinux

SELinux를 활성화하기 위해서 안드로이드가 부팅될 때 수행되는 init 프로세스에 스크립트를 추가하고, SELinux 세팅 데몬을 커널 상에서 돌려준다. SELinux를 사용하기 위해 관련 유틸리티들을 필요로 하는 라이브러리와 함께 빌드를 해서 타겟에 올린 후, 커널에서 비활성화 되어있는 SELinux를 활성화시키고, 안드로이드 환경에 맞게 보안 policy DB를 구축한다. 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 추가한다[1].

(그림 3)은 안드로이드 환경에서 SELinux를 적용시켜 어플리케이션 단에서 실행한 그림이다[1]. 맨 좌측의 그림은 SELinux의 활성을 보여주고, 가운데 그림은 정책의 적용을 나타낸다. 맨 오른쪽 그림은 정책 적용시의 접근레벨을 나타낸다. 이렇게 SELinux가 활성화되면 루트 권한으로 로그인 하여도 SELinux의 권한이 없어 어떤 동작도 수행할 수 없다.



(그림 3) 안드로이드에서의 SELinux 사용

4. 결 론

루트 권한을 허용하지 않은 채 추가적인 보안설정을 하지 않는 현재 상황에서는 사용자가 루팅을 통해 루트 권한을 획득하거나 강제적으로 루팅을 시키는 악성 애플리케이션 등을 통해 루트 권한을 획득했을 경우 보안성이 취약해지게 된다. 따라서 루트 권한을 획득했을 경우를 대비한 보안성 강화 대책이 필요하다. 본 논문에서는 이를 위해 데스크톱에서 사용되고 있는 SELinux를 안드로이드에 사용해 이런 문제를 보완하는 방법을 제시하였다. SELinux를 적용함으로써 루팅을 원하는 사용자가 안전하게 스마트폰을 원하는 대로 커스터마이징해 사용할 수 있고, 불법 애플리케이션에 의해 루트 권한을 획득한 경우에도 스마트폰의 보안성을 좀 더 강화할 수 있게 될 것이다.

참고문헌

[1] 정성화 외, "SELinux 기반 안드로이드 보안시스템 구축에 관한 연구," 한국산학기술학회 논문지, Vol.11, No.8, pp.3005-3011, 2010.

[2] 정준희 외, "스마트폰의 이용과 관련한 법적 문제점에 대한 연구", 고려대학교 대학원, 2011

[3] 김익환 외, "안드로이드 플랫폼에서 응용프로그램의 유연한 권한설정 기법 설계 및 구현", 서울시립대학교 대학원, 2011

[4] Steven Salerno 외, "Exploration of Attacks on Current Generation Smartphones", International Conference on Mobile Web Information Systems (MobiWIS), Vol. 5, pp.546-553, 2011

[5] 가트너, http://www.gartner.com

[6] 엔프로텍트, http://erteam.nprotect.com

[7] 안철수연구소, http://core.ahnlab.com