Journal of The Korea Institute of Information Security & Cryptology(JKIISC) ISSN 1598-3986(Print)
VOL.23, NO.4, August 2013 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2013.23.4.757

개인정보 유출 탐지 및 차단에 관한 연구 : 안드로이드 플랫폼 환경*

최 영 석,† 김 성 훈, 이 동 훈‡
고려대학교 정보보호대학원

Study to detect and block leakage of personal information : Android-platform environment*

Youngseok Choi,† Sunghoon Kim, Dong Hoon Lee‡
Graduate School of Information Security, Korea University

요 약

안드로이드 사용자의 증가와 함께 안드로이드를 타겟으로 하는 악성코드가 급격하게 증가하고 있으며, 대부분의 악성코드는 사용자의 개인정보 유출을 목적으로 하고 있다. 최근 국내에서도 악성코드를 이용해 개인정보를 유출하고, 소액결제를 시도하는 '체스트'라는 악성코드가 출현하여 금전피해를 발생시켰다. 안드로이드 플랫폼에서 개인정보 유출을 탐지하기 위한 다양한 기법들이 제안되었지만, 기존 기법들은 안드로이드 보안모델의 특성상 사용자의 스마트폰에 적용이 어려운 한계를 가지고 있다. 본 논문에서는 커널레벨의 시스템 콜 후킹과 white-list 기반의 접근정책을 이용해 허용되지 않은 개인정보의 접근과 인터넷 연결을 실시간으로 탐지하고, 차단하는 기법을 제안하였다. 또한 구현을 통해 실제 사용자의 스마트폰에 적용이 가능함을 증명하였다.

ABSTRACT

The Malicious code that targets Android is growing dramatically as the number of Android users are increasing. Most of the malicious code have an intention of leaking personal information. Recently in Korea, a malicious code 'chest' has appeared and generated monetary damages by using malicious code to leak personal information and try to make small purchases. A variety of techniques to detect personal information leaks have been proposed on Android platform. However, the existing techniques are hard to apply to the user's smart-phone due to the characteristics of Android security model. This paper proposed a technique that detects and blocks file approaches and internet connections that are not allowed access to personal information by using the system call hooking in the kernel and white-list based approach policy. In addition, this paper proved the possibility of a real application on smart-phone through the implementation.

Keywords: Android, Personal information, System call hooking, Malware detection

I. 서 론

오늘날, 스마트폰의 성능과 통신기술의 발달은 우리에게 많은 편리함과 효율성을 제공해주고 있다. 금융거래, 회사업무, 소셜활동 등 스마트폰 하나면 모든 업무가 가능하다. 하지만, 스마트폰이 데스크탑 PC의 기능을 대신함에 따라 PC에 저장된 개인정보들이 스

접수일(2013년 6월 4일), 수정일(2013년 6월 25일), 게재확정일(2013년 7월 3일)
* 본 연구는 지식경제부 및 한국인터넷진흥원의 "고용계약형 지식정보보안 석사과정 지원사업"의 연구결과로 수행되었음.
† 주저자, sekainoko@korea.ac.kr
‡ 교신저자, donghlee@korea.ac.kr(Corresponding author)

마트폰 속으로 옮겨져 갔고, 그로인해 PC속 개인정보를 노리는 악성코드들도 스마트폰 속으로 함께 옮겨져 왔다. 특히 안드로이드를 타겟으로 하는 악성코드의 수가 눈에 띄게 증가하고 있다. NQ Mobile에서 발표한 리포트[1]에 의하면 2012년에 95%의 모바일 악성코드는 안드로이드를 타겟으로 하고 있다. 대부분의 모바일 악성코드가 안드로이드를 타겟으로 하고 있는 이유는 안드로이드 사용자의 증가와 악성코드 제작의 용이함 때문이다. 실제로 2013년 2월 가트너에서 발표한[2]에 의하면, 안드로이드는 스마트 폰 세계 시장의 69.7%를 차지하고 있다. 안드로이드 어플리케이션은 이식성이 높은 자바언어로 구현되기 때문에 역공학이 쉬워 악성코드의 삽입과 리패키징이 용이하다. 또한 개방형 마켓 구조를 취하기 때문에 배포가 쉬워 악성코드 개발자들의 좋은 타겟이 되고 있다. 오늘날 대부분의 안드로이드 악성코드는 금전적 이득을 취하기 위한 목적으로 개인정보를 유출하고 있다. 국내에도 2012년 11월, 스마트 폰에 저장된 통신사 정보, 전화번호, SMS등의 개인정보를 탈취하여, 소액결제를 시도하는 안드로이드 악성코드 '체스트'가 출현하여 금전 피해를 발생시켰다[3]. 안드로이드 플랫폼 환경에서 개인정보 유출을 탐지하기 위한 다양한 기법들이 제시되었지만 효과적으로 개인정보 유출을 차단하지 못하고 있다. 그 원인은 다음과 같다.

첫째, 악성코드의 진화이다. 초기 안드로이드 악성코드는 잘 알려진 유명 어플리케이션을 선택하여 악성코드를 삽입하고 리패키징하는 비교적 단순한 방법을 사용하였기 때문에 소스코드나 컨텐츠를 검사하는 시그니처 기반의 탐지가 가능하였다. 하지만, 최근 악성코드는 소스코드에 난독화, 암호화를 적용하여 시그니처 기반의 탐지를 우회하고 있다. 또한 정상 어플리케이션을 가장하여 설치된 후에 인터넷을 통해 소스코드를 다운로드 하는 방법을 사용하기 때문에 기존 어플리케이션 레벨의 시그니처 기반 탐지가 불가능하다. 둘째 기존 연구들의 탐지기법들은 사용자 휴대폰에 적용이 어렵다. 안드로이드는 리눅스 커널 2.6을 기반으로 하며 샌드박스라는 보안모델을 적용하고 있다. 기본적으로 어플리케이션은 설치 시 고유한 UID를 할당받고, 독립된 프로세스에서 실행되기 때문에 다른 어플리케이션에 접근 할 수 없다. 또한 정상적인 방법으로 시스템 레벨의 권한을 획득할 수 없기 때문에 시스템의 메모리, 네트워크, 디스크에 접근해야 하는 동적 분석이 불가능하다. 기존 연구들에서 제안된 동적 분석 기법들은 별도의 서버나 가상화된 환경을 구축한 후 실행을 통해 분석해야 하기 때문에 사용자의 스마트폰에 적용이 어렵다.

본 논문에서는 사용자의 스마트폰에서 실시간으로 개인정보 유출을 탐지하고 차단하는 기법을 제안한다. 제안하는 기법은 LKM(Loadable Kernel Module)으로 구현된 시스템 콜 후킹모듈을 커널에 삽입하여, 사용자 어플리케이션의 파일 접근과 인터넷 연결을 감시하기 위해 시스템 콜을 후킹 한다. 후킹을 통해 시스템 콜 함수의 파라미터 값에서 접근하는 파일의 경로와 연결 IP주소를 읽어온다. 사용자 어플리케이션이 white-list에 없는 개인정보에 접근을 시도하거나 IP주소에 연결을 시도할 경우 사용자에게 접근 사실을 알리고, 접근의 허가나 차단을 요청한다. 사용자는 접근내용을 확인하고 차단함으로써 개인정보의 유출을 막을 수 있다. 본 논문의 기여도는 다음과 같다.

- 어플리케이션 레벨과 커널 레벨에서 사용되는 탐지 방법의 특징에 따라 기존 연구를 분류하였다.
- 시그니처 기반의 탐지가 어려운 개인정보 유출 행위를 제안기법을 이용해 탐지해 낼 수 있다.
- 사용자의 스마트폰에서 개인정보의 접근과 인터넷 연결을 실시간으로 탐지하고, 차단하는 기법을 제안하였다.
- 제안기법을 에뮬레이터 환경에 구현함으로써 사용자의 스마트폰에 적용 가능함을 증명하였다.

본 논문의 구성은 다음과 같다. 2장에서 개인정보 유출 행위를 탐지하는 기존연구들의 특징에 대하여 알아본다. 3장에서는 본 논문에서 제안하는 기법을 설명하고 4장에서 제안기법의 특징을 소개한다. 5장에서는 제안기법을 구현하고 결과를 확인한다. 마지막으로 6장에서 결론을 맺고 향후연구에 대하여 설명한다.

II. 관련연구

본 절에서는 개인정보 유출을 탐지하기 위한 기존 연구들의 특징을 알아본다. 본 논문에서는 기존연구들을 탐지 레벨의 특징에 따라 어플리케이션 레벨과 커널 레벨로 〔표 1〕과 같이 분류하였다.

〔표 1〕 탐지레벨에 따른 관련연구 분류

탐지레벨	분석방법	내 용	종 류	특 징
어플리케이션	정적분석	어플리케이션에 포함된 정적인 컨텐츠 분석을 이용한 탐지 〔4〕,〔5〕,〔6〕,〔7〕,〔8〕,〔11〕,〔14〕	- 시그니처 분석 - 안드로이드 퍼미션 분석 - 어플리케이션 구조 분석 - 소스코드 분석	어플리케이션을 실행하지 않고, 안에 포함된 정적인 컨텐츠를 분석하기 때문에 동적분석보다 상대적으로 분석시간이 짧지만, 암호화, 난독화, 동적 실행 등을 적용한 새로운 형태의 공격을 탐지하기 어렵다.
	동적분석	악성코드 실행 후 악성행위를 분석 〔8〕,〔9〕,〔10〕,〔11〕	- 데이터 흐름 분석 - 실행 로그 분석 - 안드로이드 API 분석 - 모니터링	어플리케이션을 직접 실행하여 얻어진 정보를 분석하기 때문에 정적분석보다 정밀한 탐지가 가능하지만, 실행과 분석을 위한 별도의 가상환경이나 서버가 요구된다.
커널	동적분석	시스템 콜 후킹을 이용한 분석 〔12〕,〔13〕,〔14〕	- 시스템 콜 로그분석 - 시스템 콜 파라미터 분석	시스템 자원에 접근이 가능하여 정밀한 탐지가 가능하지만, 시스템의 수정이 요구된다.

2.1 어플리케이션 레벨의 탐지

어플리케이션 레벨에서는 동적 분석 외에도 정적분석이 가능하다. 안드로이드에서 정적분석은 어플리케이션을 실행하지 않고, 그 안에 포함되어 있는 소스코드, 퍼미션 정보, 레이아웃, 리소스 등 컨텐츠의 내용을 기반으로 탐지를 하고 있다. DiCerbo 등〔4〕은 manifest.xml 파일에 포함된 안드로이드 퍼미션을 이용해 개인정보 유출행위를 예측 할 수 있음을 보였다. Kim 등〔5〕은 안드로이드 퍼미션 외에도 DEX 파일을 disassemble 하여, 정보유출을 시도하는 API를 추출해냈고 이를 동시에 활용하여 개인정보 유출을 탐지하는 모델을 제안했다. Zhou 등〔6〕은 안드로이드 마켓에서 알려진 악성코드들을 빠르게 탐지하기 위하여 퍼미션과 API를 기반으로 시그니처를 만들어 냈고, 동적으로 코드를 가져와 실행하는 새로운 형태의 악성코드를 탐지하기 위하여 네이티브 코드와 동적 코드실행을 탐지하는 휴리스틱한 탐지 기법을 제안했다. 또한 Droid Ranger를 구현하여 제안 기법의 효율성을 검증하였다. Dong-jie Wu 등〔7〕은 퍼미션과 안드로이드 컴포넌트, API를 조합한 정보를 이용해 개인정보 유출행위가 포함된 악성코드를 효과적으로 분류 할 수 있도록 여러 가지 클러스터 알고리즘을 시험하여 가장 적절한 알고리즘을 찾아냈다.

Fuchs 등〔8〕은 manifest.xml 파일 안에 명시된 보안정책과 안드로이드의 컨텐츠 프로바이더를 이용하여 개인정보의 흐름을 추적하는 Scandroid를 구현하여 개인정보의 일반적인 흐름을 분석했다. Enck 등〔9〕은 실제 악성코드의 실행을 통해 민감한 개인정보의 흐름을 추적 할 수 있도록 TaintDroid를 구현하고 이를 이용해 개인정보 유출행위를 탐지 할 수 있음을 증명하였다. Zhao 등〔10〕은 개인정보의 접근과 유출 행위의 로그를 기록하고, 로그를 기반으로 행위의 흐름과 특징을 찾아낸 후 SVM 알고리즘을 적용하여 성공적으로 개인정보 유출 행위를 탐지 할 수 있음을 보였다. 국내에서는 심원태 등〔11〕이 Dalvik VM(가상머신)을 수정하여 안드로이드 API를 추출하고, API의 호출순서를 이용해 개인정보 유출과 악성행위를 탐지하는 기법을 제안하였다. 위의 연구들은 어플리케이션 레벨의 정적분석과 동적분석을 실시하여 개인정보 유출행위를 탐지 해냈다. 하지만, 어플리케이션 레벨의 정적분석은 기존 악성코드의 시그니처가 요구되기 때문에 새로운 형태의 공격에 취약하다. 또한 제안된 기존 연구들의 동적분석 기법들은 분석을 위한 가상화된 환경이나 별도의 서버가 요구되어 실제 사용자의 스마트 폰에 적용이 어려운 한계를 갖고 있다.

2.2 커널 레벨의 탐지

커널 레벨에서는 시스템자원의 접근이 가능하여 실시간으로 행위를 분석 할 수 있지만 시스템의 수정이 불가피한 단점이 있다. Buraguera 등〔12〕은 시스템 콜 로그를 기록하여, 개인정보 유출 행위의 특징을 추

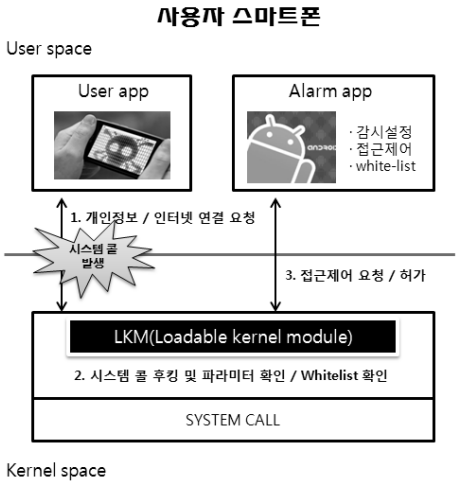
출한 후 vector 로 만들고, K-means 알고리즘을 적용하여 악성코드를 탐지하는 방법을 제안하였다. Takamasa isohara등 [13]은 시스템 콜을 후킹 하는 방법으로 시스템 콜과 파라미터 로그를 기록하여 정보 유출시 사용되는 시그니처를 찾아냈다. Thomas 등[14]은 시스템을 수정하여, 가상화된 환경을 만들고 그 안에서 퍼미션을 분석하고, 시스템 콜 로그를 기록하여 악성코드를 탐지하는 종합적인 분석방법을 제시하였다. 하지만, 기존 커널 레벨에서의 연구들은 로그기반의 탐지를 활용하기 때문에 실시간으로 탐지가 어렵고, 차단이 불가능하다.

III. 제안 기법

3.1 제안하는 기법 개요

제안하는 기법은 [그림 1]과 같이 User_app, Alarm_app, 시스템 콜 후킹 모듈(LKM)로 구성된다. User_app은 사용자가 설치한 어플리케이션으로 개인정보 유출 행위의 감시 대상이 된다. Alarm_app은 LKM으로 구현된 시스템 콜 후킹 모듈과 연동되며 white-list에 존재하지 않는 개인정보의 접근과 인터넷 연결 시도를 사용자에게 알린다.

사용자가 [그림 2]와 같이 Alarm_app에서 개인정보 유출행위를 탐지하고 차단하기 위한 사용자 어플리케이션을 선택하면, 시스템 콜 후킹을 위한 모듈이 커널에 삽입된다. 후킹모듈은 LKM으로 구현하여 시스템 운영중에 언제든 동적으로 모듈의 삽입과 제거



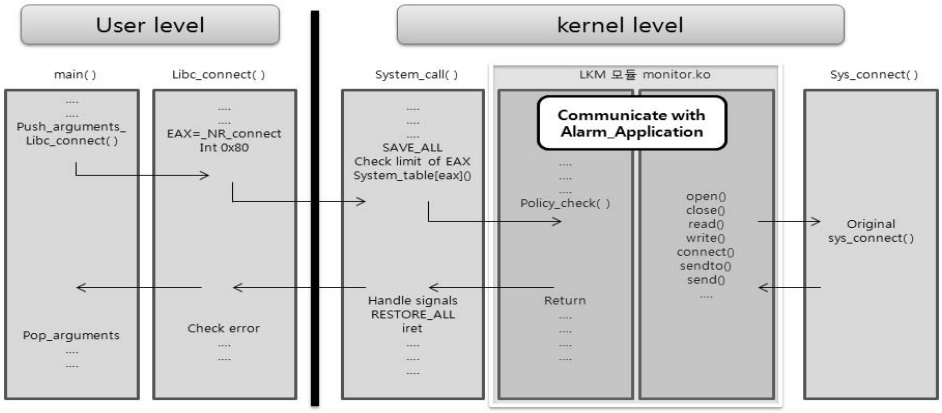
[그림 1] 제안기법 동작 흐름

[그림 2] 접근제어 어플리케이션 설정

가 가능하다. 커널에 삽입된 후킹 모듈은 파일 접근 [표 3]과 인터넷 연결[표 4]에 관련된 시스템 콜을 후킹 하여, 파라미터 값에서 접근하는 파일의 경로와 연결 IP주소를 읽어온다. 읽어온 값이 Alarm_app의 white-list에 존재하지 않는 개인정보이거나 IP주소일 경우 사용자에게 Alarm_app을 통해 접근의 허가나 차단을 요청하게 된다. 사용자는 접근요청 정보를 확인하여 허용되지 않은 개인정보의 접근이나 인터넷

[표 2] 스마트폰 개인정보 파일

개인정보	파 일 경 로
연락처	/com.android.providers.contacts/databases/contacts2.db
통화목록	/com.android.providers.telephony/databases/telephony.db
SMS	/com.android.providers.telephony/databases/mmssms.db
검색기록	/com.android.browser/databases/browser.db
패스워드	/com.android.browser/databases/webview.db
달력	/com.android.providers.calendar/databases/calendar.db
이메일	/com.android.email/databases/EmailProvider.db
카카오톡	/com.kakao.talk/databases/KakaoTalk.db
페이스북	/com.htc.socialnetwork.facebook/databases/facebook.db
메모	/com.sec.android.app.memo/databases/Memo.db
공인 인증서	/mnt/sdcard/NPKI/yessign/USER



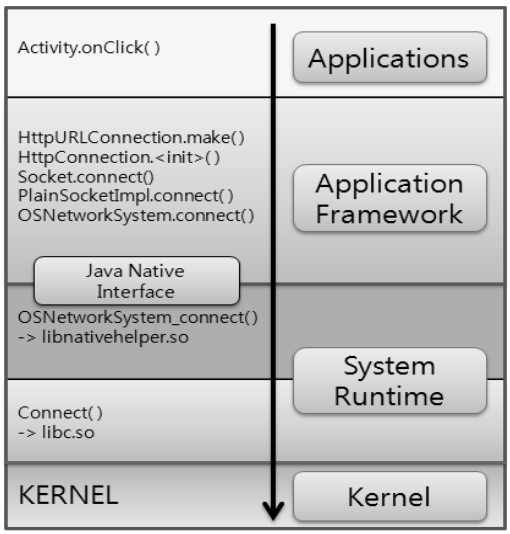
〔그림 3〕 LKM 모듈을 이용한 시스템 콜 후킹과정

연결을 탐지하고 차단함으로써 개인정보의 유출을 차단 할 수 있다.

3.2 시스템 콜 후킹

어플리케이션 레벨에서 실행하는 함수들은 모두 시스템 콜로 변환되어 커널에 전해진다. JAVA로 만들어진 안드로이드 어플리케이션의 경우 JNI(Java Native Interface)를 통해 C 나 C++ 형태의 안드로이드 시스템 라이브러리를 호출하고, 이는 다시 libc 나 libstdc++를 참조하여 시스템 콜을 호출하게 된다. 실제로 인터넷 연결을 위해 화면을 클릭할 경우 〔그림 3〕과 같은 순서로 시스템 콜이 호출된다. Activity.onClick() 함수의 실행과 동시에 안드로이드 API 함수인 connect()가 호출되고 이 함수는 JNI를 거쳐 OSNetworkSystem_connect() 함수를 호출하게 된다. 하지만 이 함수 역시 libc 에 정의된 connect()를 참조하여 시스템 콜을 발생시킨다. 따라서 커널 레벨에서 시스템 콜을 후킹하면 모든 개인정보의 접근과 인터넷 연결 시도를 효과적 제어 할 수 있게 된다〔15〕.

본 논문에서는 시스템 콜을 후킹하기 위하여 동적으로 모듈 삽입이 가능한 LKM으로 후킹 모듈을 구현하였다. 정상적인 시스템의 경우 〔그림 3〕과 같이 인터넷 연결을 위한 connect() 시스템 콜이 호출되면 라이브러리에서 호출된 시스템 콜 번호를 EAX 레지스터에 저장한다. 후에 시스템 콜 인터럽트를 발생 시키고, 시스템 콜 테이블에서 EAX 레지스터에 저장된 번호를 읽어 해당 시스템 콜 함수를 실행한다. 하지만, LKM으로 구현된 후킹 모듈을 커널에 삽입하게



〔그림 4〕 안드로이드 시스템 콜 동작

되면 〔그림 4〕와 같이 시스템 콜〔표 3〕,〔표 4〕을 후킹하여, 시스템 콜 함수의 파라미터 값에서 접근하는 파일의 경로와 연결 IP주소를 확인 할 수 있다.

3.3 개인정보 접근 정책

먼저, 개인정보의 접근을 탐지하기 위하여 스마트폰 안에 포함된 개인정보들을 파악하고 분류하였다. 안드로이드 어플리케이션은 /data/data/"package" 디렉토리에 저장되며, 대부분의 개인정보 관련 데이터는 SQLite 데이터베이스로 저장되어 있다. 분류된 개인정보들은 금융거래에 이용되는 공인인증서와 프라이버시를 침해 할 수 있는 정보들로 〔표 2〕와 같다.

〔표 3〕 파일 접근탐지 시스템 콜

시스템 콜	함수 선언부
open()	int open (const char ***pathname**, int flags)
read()	ssize_t read (int **fd**, void *buf, size_t count)
write()	ssize_t write (int **fd**, const void *buf, size_t count)

악성코드나 사용자 어플리케이션이 개인정보에 접근하는 것을 차단하기 위하여 파일 경로 기반의 white-list를 이용한다. 삽입된 후킹 모듈을 이용해 시스템 콜 파라미터 값에서 접근하는 파일의 경로를 확인한다. 감시 대상으로 설정된 어플리케이션이 개인정보 파일에 접근을 시도하면 white-list를 확인하여 허용된 개인정보 파일에만 접근을 허용한다.

white-list에 존재하지 않는 개인정보 파일에 접근을 시도하면 Alarm_app을 통해 사용자에게 개인정보 접근 사실을 알리고, 접근의 허가나 차단을 요청한다. 파일에 대한 접근을 탐지하기 위해 후킹 하는 시스템 콜은 〔표 3〕과 같다.

3.4 인터넷 연결 접근 정책

악성코드나 개인정보 유출을 시도하는 어플리케이션들은 획득한 개인정보를 외부로 유출하기 위해 인터넷 연결을 시도할 것이다. 따라서 인터넷 연결을 제어하면 효과적으로 개인정보 유출을 차단할 수 있다. 본 연구에서는 인터넷 연결을 제어하기 위해 IP주소 기반의 white-list를 이용한다. 감시대상 어플리케이션이 인터넷 연결을 위한 시스템 콜을 사용하면, 후킹 모듈은 시스템 콜의 파라미터에서 연결 IP주소를 확인한다. white-list에 없는 IP주소에 인터넷 연결을 시도할 경우 Alarm_app에 해당 IP주소를 넘겨준다.

〔표 4〕 인터넷 연결탐지 시스템 콜

시스템 콜	함수 선언부
connect()	int connect (int sockfd, const struct sockaddr ***addr**, socklen_t addrlen);
sendto()	ssize_t sendto (int sockfd, const void *buf, size_t len, int flags, const struct sockaddr ***dest_addr**, socklen_t addrlen);

Alarm_app은 IP주소 정보를 조회하여 사용자에게 인터넷 연결시도 사실과 해당 IP주소의 세부정보를 알리고, 인터넷 연결의 허가나 차단을 요청한다. 인터넷 연결에 대한 탐지를 위해 후킹 하는 시스템 콜은 〔표 4〕와 같다.

3.5 white-list 구성

white-list는 Alarm-app에서 관리되며 어플리케이션 단위로 접근 가능한 개인정보와 연결 IP주소를 포함하고 있다. white-list는 사용자의 결정에 의해 추가되어 질 수 있다.

IV. 제안하는 기법의 특징

본 논문에서 제안하는 기법은 허용되지 않은 개인정보의 접근과 인터넷 연결을 실시간으로 확인하고, 차단 할 수 있는 특징이 있다. 또한 기존 탐지기법들은 동적분석을 위한 별도의 가상환경이나 서버가 요구되지만, 제안기법은 사용자의 스마트폰에서 탐지와 차단이 가능하다. 〔표 5〕는 제안하는 기법과 기존에 제안된 악성행위탐지를 위한 분석기법〔11〕을 비교한 표이다.

기존에 제안된 분석기법은 Dalvik VM의 수정을 통해 안드로이드 API 호출 로그를 생성하고, 생성된 로그를 기반으로 악성행위를 판별해 낸다. 이 기법은 기존에 제안된 동적분석 기법들〔12〕〔13〕〔14〕과 같이

〔표 5〕 기존 분석 기법〔11〕과 제안기법 비교

구분	기존 기법〔11〕	제안기법
정적분석	시그니처 확인, 소스코드 분석	실시하지 않음
동적분석	Dalvik VM 수정을 이용한 안드로이드 API 후킹 및 분석	리눅스 커널에서 리눅스 시스템 콜 후킹 및 분석
	안드로이드 API 호출 순서에 따른 패턴 매칭	White-list 기반의 개인정보 파일 접근 및 인터넷 연결 제어
개인정보 유출탐지	어플리케이션 실행 후 로그분석	실시간으로 개인정보 접근을 확인
스마트폰 적용	어려움 (로그 분석위한 별도의 환경이 요구)	간단 (LKM으로 구현된 후킹 모듈 삽입)
유출차단	불가	가능
시스템의 수정	필수 (Dalvik VM 수정)	필수 (후킹모듈 삽입 권한)

실행 후 남겨진 로그를 분석하여 악성코드를 탐지한다. 때문에 개인정보 유출행위가 발생한 후에 탐지가 가능한 단점이 존재한다. 하지만, 본 논문에서 제안하는 기법은 간단히 사용자의 스마트폰에 LKM으로 구현된 시스템 콜 후킹 모듈을 삽입하고, white-list 기반의 접근 정책을 적용함으로써 허용되지 않은 개인정보의 접근과 인터넷 연결을 실시간으로 탐지하고, 차단 할 수 있다.

V. 구현 및 결과

본 절에서는 제안기법을 안드로이드 에뮬레이터 환경에서 구현한 내용과 구현 결과에 대해서 설명한다.

5.1 구현

• 구현환경

본 논문에서 제안하는 기법을 구현한 환경은 다음과 같다.

CPU : Intel core(TM) i5-2400 @3.10GHz
RAM : 4.00GB
OS : ubuntu 10.24
Android : Froyo 2.2, kernel 2.6.29, Emulator

• 시스템 콜 후킹 모듈

개인정보의 접근과 인터넷 연결에 대한 탐지 및 차단을 위한 후킹 모듈은 〔그림 5〕와 같이 LKM으로 구현하여 간단하게 동적으로 커널에 삽입이 가능하고, 모듈의 확장이나 제거가 용이하다.

• 접근정책 구현

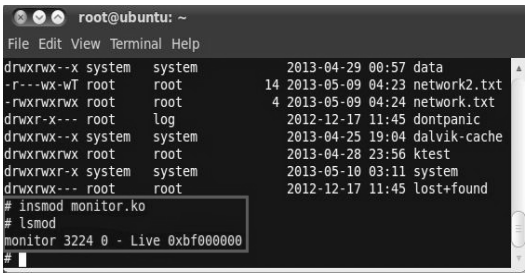
커널에 삽입된 후킹 모듈은 〔표 6〕과 같이 동작한다. 개인정보 유출 탐지 대상 어플리케이션의 UID를 확인하고, 시스템 콜 함수의 파라미터에서 IP주소를 읽어온다. 읽어온 IP주소가 white-list에 존재하지 않는 IP주소의 경우 사용자에게 접근제어를 요청한다. 커널 내부에서 후킹 모듈은 〔그림6〕과 같이 동작하게 된다.

〔표 6〕 인터넷 연결 접근제어 의사코드

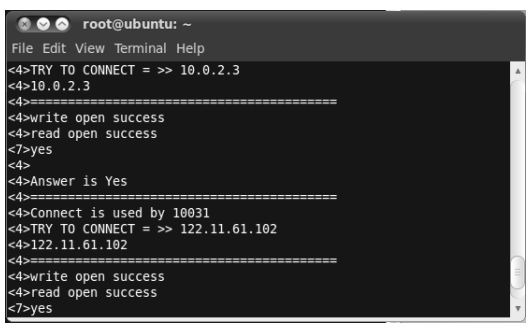
```
sys_connect( int sockfd, struct sock *addr, int len )
{
  int current_uid = get_current_uid(); //현재 어플 uid 값
  int selected_uid = get_selected_uid(); //감시 어플 uid 값
  ip_addr = addr.ip_address; //파라미터에서 IP주소 추출

  if( (current_uid == selected_uid)
        && (ip_addr is not in white-list) )
  {
    send( ip_addr to Alarm app ); //IP주소 전송
    while( true )
    { // 접근제어 요청 후 대기
      if( strlen(value = read(from Alarm App)) > 0 )
      {

        if ( value == "yes" ) //접근 허용
          return original_call_connect();
        else // 접근 거부
          return -1;

      }
    }
  }
  else
  {
    return original_call_connect();
  }
}
```

5.2 구현 결과

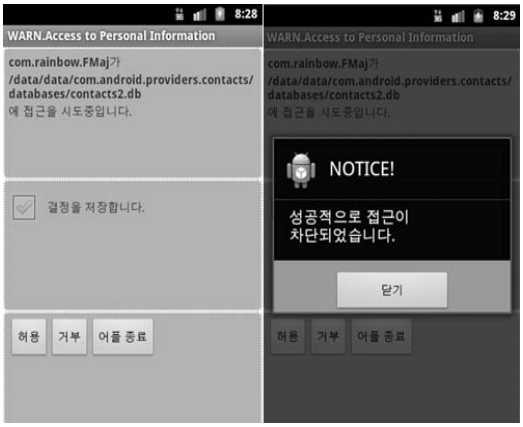
구현된 모듈이 정상적으로 동작하는지 시험하기 위



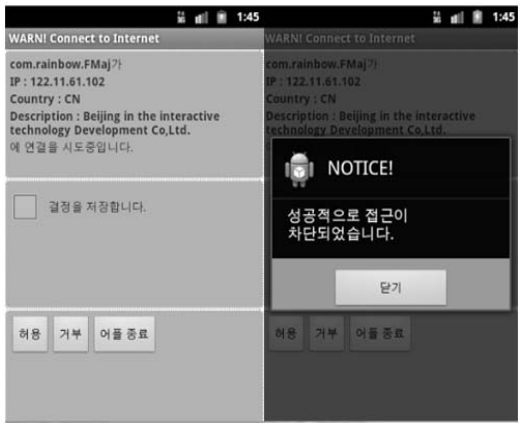
〔그림 5〕 LKM으로 구현된 시스템 콜 후킹 모듈



〔그림 6〕 connect() 함수 후킹 모듈의 동작



〔그림 7〕 개인정보 접근 시도 및 차단



〔그림 8〕 인터넷 연결 시도 및 차단

하여 com.rainbow.FMaj라는 악성코드를 설치하였다. com.rainbow.FMaj는 리패키징된 악성코드로써 사용자의 개인정보를 획득하고 인터넷을 통해 중국의 서버로 유출하는 행위를 한다〔16〕. 악성코드의 정보는 다음과 같다.

- Package Name : com.rainbow.FMaj
- MD5 : 89BB300CC1BF0B27C582327588EA7377

com.rainbow.FMaj를 개인정보 유출 탐지 대상 어플리케이션으로 〔그림 2〕과 같이 설정한다. 탐지 설정이 완료되면 Alarm_app은 백그라운드 상태로 동작한다. 허용되지 않은 개인정보의 접근이나 인터넷 연결 요청이 발생하면 푸시알람을 이용해 사용자에게 알린다. 실제, 악성코드 com.rainbow.FMaj가 white-list에 존재하지 않는 개인정보에 접근을 시도하면, 사용자는 〔그림 7〕과 같이 Alarm_app을 통하여 개인정보의 접근 사실을 확인하고 차단 할 수 있다. 또한, 악성코드가 개인정보 획득에 성공해도 외부 유출을 위해 white-list에 존재하지 않는 IP주소에 인터넷 연결을 시도한다면, 제안 기법은 〔그림 8〕과 같이 인터넷 연결시도를 탐지하고, 연결을 시도하는 IP주소의 세부정보와 함께 사용자에게 인터넷 연결 허가/차단을 요청한다. 사용자는 요청정보를 확인하고 차단함으로써 개인정보의 유출을 막을 수 있다.

VI. 결 론

본 논문에서는 커널 레벨에서 시스템 콜을 후킹하고 white-list 기반의 접근정책을 적용하여, 허용되지 않은 개인정보의 접근과 인터넷 연결을 효과적으로 차단 할 수 있는 기법을 제안하였다. 또한 구현을 통해 사용자의 스마트폰에 적용이 가능함을 보였다. 제안하는 기법은 커널 레벨에서 시스템 콜을 후킹하기 때문에 모든 파일의 접근과 인터넷 연결을 탐지 할 수 있다. 따라서 기존 시그니처 기반에서 탐지가 어려운 개인정보 유출행위를 탐지 할 수 있으며, 기존에 제안된 기법들과는 다르게 사용자의 스마트폰에서 실시간으로 탐지와 차단이 가능하다. 하지만 본 논문에서 제안하는 기법은 시스템 콜 후킹 모듈을 삽입하기 위해 시스템 레벨의 권한이 요구되는 단점이 존재한다. 이는 플랫폼 제조사에서 시스템 권한의 어플리케이션을 추가함으로써 해결 될 수 있다.

향후에는 커널 레벨에서 플랫폼의 취약점을 이용한 권한상승 공격과 제안기법을 우회하기 위해 다른 후킹 모듈을 삽입하는 잠재적 위협을 탐지하기 위한 연구를 진행 할 것이다.

참고문헌

〔1〕 NQ Mobile, “2012 Security Report,” http://www.nq.com/2012_NQ_Mobile_Security_Report.pdf, 2013년 5월 11일.

〔2〕 Gartner, “Gartner Says Worldwide Mobile Phone Sales Declined 1.7 Percent in 2012,” http://www.gartner.com/newsroom/id/2335616, 2013년 2월 13일.

〔3〕 ZDNET Korea, “안드로이드 악성코드 결제 피해 국내 발생,” http://www.zdnet.co.kr/news/

news_view.asp?artice_id=20121202133055, 2012년 12월 02일.

[4] F. Di Cerbo, A. Girardello, F. Michahelles, and S. Voronkova, "Detection of malicious applications on android os," Proceedings of the 4th international conference on Computational forensics, IWCF'10, pp 138-149, November 2011.

[5] S. Kim, J. I. Cho, H. W. Myeong, and D. H. Lee, "A study on static analysis model of mobile application for privacy protection," Computer Science and Convergence, vol 114, pp 529-540, 2012.

[6] Y. Zhou, Z. Wang, W. Zhou, and X. Jiang. Hey, you, "get off of my market: Detecting malicious apps in official and alternative Android markets," Proceedings of the 19th Annual Network & Distributed System Security Symposium, Feb 2012.

[7] Dong-Jie Wu, Ching-Hao Mao, Te-En Wei, Hahn-Ming Lee, Kuo-Ping Wu, "DroidMat: Android Malware Detection through Manifest and API Calls Tracing," 2012 Seventh Asia Joint Conference on Information Security (Asia JCIS), pp 62-29, August 2012.

[8] A. P. Fuchs, A. Chaudhuri, and J. S. Foster, "SCanDroid: Automated Security Certification of Android Applications," Technical Report CSTR-4991, Department of Computer Science, University of Maryland, November 2009.

[9] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "Taintdroid: An information-flow tracking system for realtime privacy monitoring on smartphones," Proceedings of the 9th USENIX conference on Operating systems design and implementation, October 2010.

[10] M. Zhao, F. Ge, T. Zhang, and Z. Yuan, "Antimaldroid: An efficient svmbased malware detection framework for android," Communications in Computer and Information Science, vol 243, pp 158-166, 2011.

[11] 심원태, 김종명, 류재철, 노봉남, "안드로이드 앱 악성행위 탐지를 위한 분석 기법 연구," 정보보호학회논문지, 21(1), pp. 213-219, 2011년 2월.

[12] Iker Burguera, Urko Zurutuza, Simin Nadjm-Tehrani,"crowdroid: behavior-based malware detection system for Android," Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices, pp 15-26, 2011.

[13] T. Isohara, K Takemori, A.kubota, "Kernel-based Behavior Analysis for Android Malware Detection," 2011 Seventh International Conference on Computational Intelligence and Security, pp 1011-1015, Dec 2011.

[14] T. Bläsing, L.Batyuk, A. D. schmidt, S. A. Camtepe, S. Albayrak, "An Android Application Sandbox system for suspicious software detection," 2010 5th International Conference on Malicious and Unwanted Software (MALWARE), pp 55-62, October 2010.

[15] Rubin Xu, Hassen Saïdi, Ross Anderson, "Aurasium: practical policy enforcement for Android applications," Proceeding of Security'12 Proceedings of the 21st USENIX conference on Security, pp 27-27, 2012.

[16] Foresafe, "App Report : Attention," http://www.foresafe.com/report/89BB300CC1BF0B27C582327588EA7377, 2013년 3월 10일.

〈저 자 소 개〉

최 영 석 (Youngseok Choi) 학생회원
2010년 2월: 한국외국어대학교 컴퓨터공학과 졸업
2012년 3월~현재: 고려대학교 정보보호대학원 석사과정
〈관심분야〉 금융보안, 스마트폰 보안, 네트워크 보안

김 성 훈 (Sunghoon Kim) 학생회원
2006년 8월: 서울시립대학교 수학과 졸업
2009년 2월: 고려대학교 정보보호대학원 석사
2009년 1월~2011년 2월: (주)알티캐스트 CAS개발본부
2011년 3월~현재: 고려대학교 정보보호대학원 박사과정
〈관심분야〉 소프트웨어 보안, 소프트웨어 난독화

이 동 훈 (Dong Hoon Lee) 종신회원
1983년 8월: 고려대학교 경제학과 졸업
1987년 12월: Oklahoma University 전산학과 석사 졸업
1992년 5월: Oklahoma University 전산학과 박사 졸업
1992년 8월: 단국대학교 전자계산학과 전임강사
1993년 3월~1997년 2월: 고려대학교 전산학과 조교수
1997년 3월~2001년 2월: 고려대학교 전산학과 부교수
2001년 2월~현재: 고려대학교 정보보호대학원 교수
〈관심분야〉 암호프로토콜, 암호이론, USN 이론, 키 교환, 익명성 연구, PET 기술