Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.24, NO.1, Feb. 2014 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2014.24.1.145

행위기반의 프로파일링 기법을 활용한 모바일 악성코드 분류 기법*

윤 재 성,† 장 재 욱, 김 휘 강‡
고려대학교 정보보호대학원

Andro-profiler: Anti-malware system based on behavior profiling of mobile malware*

Jae-sung Yun,† Jae-wook Jang, Huy Kang Kim‡
Graduate School of Information Security, Korea University

요 약

본 논문에서는 범죄수사에서 사용되는 프로파일링 기법을 이용한 모바일 악성코드 행위 프로파일링을 통하여 효율적인 모바일 악성코드 분류 방법론 Andro-profiler를 제안한다. Andro-profiler는 클라이언트/서버 형태로, 클라이언트 앱이 모바일기기에 설치되어 사용자가 사용하고 있는 앱에 대한 정보를 서버에 전송하고, 서버에서는 해당 앱을 동적 분석 도구인 Droidbox가 설치된 에뮬레이터에서 실행시키면서 발생되는 시스템 콜과 에뮬레이터 로그를 이용하여 해당 앱의 행동을 프로파일링하며, 해당 앱의 프로파일링 목록을 저장된 악성코드 프로파일링 DB와 비교하여 악성유무를 판단하고, 악성코드로 판단될 경우 분류를 실시하여 클라이언트에게 결과를 통보한다. 실험결과, Andro-profiler는 1MB의 악성코드를 분류하는데 평균 55초가 소요되었고, 99%의 정확도로 악성코드를 분류하는 것을 확인하였으며, 기존 방법론보다 더 정확하게 악성코드를 분류할 수 있다.

ABSTRACT

In this paper, we propose a novel anti-malware system based on behavior profiling, called Andro-profiler. Andro-profiler consists of mobile devices and a remote server, and is implemented in Droidbox. Our aim is to detect and classify malware using an automatic classifier based on behavior profiling. First, we propose the representative behavior profiling for each malware family represented by system calls coupled with Droidbox system logs. This is done by executing the malicious application on an emulator and extracting integrated system logs. By comparing the behavior profiling of malicious applications with representative behavior profiling for each malware family, we can detect and classify them into malware families. Andro-profiler shows over 99% of classification accuracy in classifying malware families.

Keywords: Behavior profiling, Malicious behavior, Similarity, System call, Integrated system log, Android platform, Malware

접수일(2013년 10월 28일), 수정일(2013년 12월 26일), 게재확정일(2013년 12월 26일)
* 본 연구는 미래창조과학부 및 정보통신산업진흥원의 IT융합 고급인력과정 지원사업의 연구결과로 수행되었음 (NIPA-2013-H0301-13-3007)
† 주저자, yjs8888@korea.ac.kr
‡ 교신저자, cenda@korea.ac.kr(Corresponding author)

I. 서 론

가트너 보고서에 의하면 2012년 기준으로 안드로이드 플랫폼이 설치된 스마트 기기는 82억만대 정도가 판매되었고(전 세계 70%의 점유율), 2013년 4사분기에는 10억만대 정도가 더 판매될 것이라고 예상하고 있다. 이러한 기하급수적인 스마트폰의 보급화는 악성코드 제작자의 새로운 관심거리를 제공하고 있다.

백신업체인 F-Secure와 AV-TEST에서 각각 발행한 "Mobile Threat Report Q2 2012"[12]와 "AV-TEST Examines 22 Antivirus Apps" 보고서[1]에 따르면 악성코드의 양은 지속적으로 증가하고 있고, 특히 2011년 1월부터 2012년 10월까지 10만개 정도의 새로운 악성코드가 등장했고, 2012년 4사분기에는 이전분기보다 거의 2배에 달하는 96개의 새로운 패밀리와 변종들이 발견되었다고 한다. 백신회사는 끊임없이 쏟아지는 새로운 변종들을 포함한 많은 양의 악성코드들을 매일 분석하고 있고, 분석된 결과를 백신 업데이트 내용에 반영하여 악성코드들로부터 생기는 피해를 예방하고 있으나, 신속한 대응에 어려움이 많다.

모바일 기기로의 악성코드 확산을 막기 위해 백신업체들은 PC 기반의 악성코드 탐지방법을 모바일 기기에도 확장시키고 있다. 일반적으로 백신업체들이 사용하는 탐지방법은 시그니처 기반 탐지방법으로 난독화 기술을 사용할 경우 악성코드의 행위에 변화를 주지 않고도 코드의 일부 또는 전체를 변경함으로써 탐지를 피할 수 있다. 다시 말하면, 난독화 기술이 적용된 악성코드일 경우 코드가 변경되어 코드의 가독성을 낮추어 분석을 어렵게 하여 결과적으로 정확한 시그니처를 추출하기가 힘들다. 또한 시그니처 기반 탐지방법은 지속적으로 시그니처 데이터베이스의 업데이트를 통하여 새로운 시그니처를 등록하는 것이 필요한데, 이미 알려진 악성코드에 대해서는 효과적으로 탐지를 할 수 있는 반면, 제로데이 공격과 같은 알려지지 않은 악성코드에 대해서는 탐지하기가 어렵다. 뿐만 아니라, 모바일 백신 개발 시 고려해야 할 사항은 부족한 컴퓨팅 파워와 제한된 배터리 용량과 같은 스마트 기기에서의 리소스의 한계이고, 이러한 제한적인 리소스는 모바일 백신의 탐지 정확도에 상당한 영향을 끼칠 수 있다. 시그니처 기반의 탐지방법의 선천적인 문제점과 모바일 환경의 특수함으로 인해 기존에 많은 논문에서 동적 기반 악성 코드 탐지방법 및 분류방법을 제안하여 왔다.

이전까지의 연구들에서 제안된 악성코드 분류 및 탐지방법들은 요청된 퍼미션, 디컴파일된 코드들과 API 호출 셋들을 이용한 것들이다. 먼저 퍼미션 기반 탐지 방법들[6, 14, 15]은 보통 악성코드를 탐지하는 것에서는 효과적이지만 정상 앱들을 정상으로써 분류하는 것에서는 오탐(false positive)이 발생하여 효과적이지 않다. 디컴파일된 코드들을 기반으로 하는 악성코드 탐지방법[5]은 난독화 기술에 있어서 취약하다. 난독화 기술이 적용된 악성코드를 디컴파일했을 때 코드의 가독성을 떨어트리기 때문에 분석하는 데 굉장히 많은 시간이 소요된다. 그리고 API 호출기반 악성코드 탐지 방법[18, 20]은 디컴파일 과정이 완료되기 전까지 시그니처를 생성할 수 없고, 이 방법 역시 난독화 기술에 취약하다는 단점이 존재한다. 그동안의 논문에서 제시되는 대부분의 방법들[5, 6, 14, 15, 18, 20]은 모바일 기기의 하드웨어 제약을 크게 고려하지 않았으며, 기기 밖에서 운용되는 탐지 방법들은 빠르게 악성코드에 대처할 수 없었다.

본 논문에서는 이를 해결하기 위해 모바일 악성코드의 행위 프로파일링(behavior profiling) 기반으로 악성코드 탐지 및 분류기법을 제시한다. 제안하는 방법론은 크게 클라이언트인 모바일 기기와 서버로 이루어져 있으며, 현실세계의 범죄수사에 사용되는 프로파일링 기법을 확대 적용하였다. 보통 범죄자 프로파일링 기법으로 알려진 프로파일링 수사기법은 수사 중인 사건의 주제나 범죄자의 독특한 특징들을 프로파일링하여 수사관이 정확하게 예측하고 수사하는데 도움을 주기 위해 만들어진 방법이다[10, 13]. 이러한 프로파일링 수사기법을 악성코드 분석방법에 적용하면, 시스템 콜을 포함한 로그로부터 악성코드의 독특한 악의적인 행위 패턴을 프로파일링하여 악성코드 분석시 사용할 수 있다. 본 논문에서는 동적 분석 기반의 프로파일링 기법을 이용한 새로운 행위기반 탐지방법을 제안한다. 동적 분석 툴이 설치된 에뮬레이터에서 악성 어플리케이션을 설치 및 실행하여 산출되는 시스템 콜 정보 및 에뮬레이터 로그를 활용하여 악성코드의 고유의 행위 패턴을 추출한다. 추출된 정보를 기반으로 프로파일링 데이터를 만들어 다른 악성코드 프로파일링과의 유사도를 측정하고, 측정결과를 통해 악성 여부를 최종 판단하며, 악성코드 패밀리로 분류를 하는 방법론(Andro-profiler)을 제안한다.

본 논문은 2장에서는 제안된 행위 프로파일링을 설명하고, 3장에서는 제안된 Andro-profiler 시스템 구조를 설명한다. 4장에서는 실험 및 성능평가 결과를

설명하고, 마지막으로 5장에서 결론을 맺는다.

II. 행위 프로파일링(Behavior profiling)

보통 악성코드 제작자는 악성코드의 변종을 만들 때, 악성행위를 위한 핵심코드는 그대로 유지한 채, 기능을 추가 또는 삭제한다. 악성코드 군에는 고유의 악성행위 패턴을 가지고 있기 때문에 악성코드 분석가들은 이러한 악성코드 고유의 특정 시스템콜의 호출빈도, 시퀀스 등의 공통적인 특징을 바탕으로 악성코드를 분류한다.

일반적으로 프로그램이 실행될 때, 사전에 약속된 시스템 콜 함수들이 호출된다. 사용자 레벨에서의 콜 함수(API)는 악성코드 제작자에 의해서, 변조가 가능하지만, 커널 레벨에서의 콜 함수(이하 시스템콜)은 변조가 어렵다. 본 논문에서는 동적 에뮬레이터 도구인 Droidbox의 안드로이드 SDK 커널을 후킹하여, 앱이 실행될 때 발생되는 시스템 콜 함수와 그 매개변수, 그리고 Droidbox에서 제공하는 로그(퍼미션 정보, 네트워크 통신 정보 등)를 통합한 Integrated log를 활용하여 새로운 악성코드 분류기법을 제안한다. 그러나 통합 시스템 로그는 로우 데이터로, 악성코드 분석가들에게 직관적인 판단을 제공하는데 제약이 있다. 악성행위 분석에 꼭 필요한 정보를 직관적으로 제공하기 위한 방법으로 도표, 도식화 등이 있으나, 본 논문에서는 현실세계의 범죄수사에서 활용중인 범죄 프로파일링 기법을 악성코드 분석분야에 확대 적용한다[2].

한편, 악성코드간 유사도 비교시 계산 복잡도를 줄이기 위하여 각각의 악성코드 프로파일링의 메타데이터는 파이선 프로그래밍 언어의 자료처리형의 하나인 사전구조로 정의하고, 통합 시스템 로그로부터 행위 패턴을 추출하여 프로파일링 화하고 프로파일링간 유사도 측정까지 전 부분을 파이선 언어로 자동화하여 구현하였다. 프로파일링 형태에 대한 자세한 설명은 다음과 같다.

- 행위 프로파일링 정의(Definition) : 행위 프로파일링 P는 $P=(O, OP, \Gamma, \Delta)$와 같이 4개의 원소를 갖는 집합으로 구성되어 있는데, O는 모든 객체 셋을 의미하고, OP는 모든 오퍼레이션 셋을 의미하는 것으로 키와 값으로 구성되는 {name : {content : attribute}}와 같은 형태의 내제 사전 구조(Nested Dictionary)로 이루어져 있다. 그리고 $\Gamma \subseteq (O \times OP)$는 객체와 오퍼레이션과의 매핑 관계를 의미하고, $\Delta \subseteq ((O \times OP), (O \times OP))$는 전체 객체-오퍼레이션 관계 집합을 말하며, 모든 관계는 통합 시스템 로그로 설명가능하다.
- 객체(Object) : 객체는 악성코드가 각각의 행위를 하는데 필요한 추상적인 기능을 말한다. PC 기반 악성코드와는 다르게 모바일 악성코드의 행위 패턴은 크게 3가지로 단순화하여 설명할 수 있다. 과금을 유발시키는 번호로 전화를 걸거나 과금을 유발할 수 있는 번호로 SMS를 보내는 행위, 민감한 개인정보를 다른 곳으로 전송하는 행위가 바로 그것이다. 객체는 앞에서 설명한 3가지 행위를 의미하며, 다음과 같이 기호화하여 설명할 수 있다.

Object ::= Object-type
Object-type ::= Telephony | phone | Network

- 오퍼레이션(Operation) : 오퍼레이션은 구체적인 악성 행위를 나타낸다. 오퍼레이션은 오퍼레이션 이름, 타깃, 속성으로 구성되어 있으며, 오퍼레이션 이름은 악성 행위를 하는 구분자를 말하는 것이고, 타깃은 말 그대로 악성코드의 공격 목표를 말하는 것이다. 예를 들어, 외부 메모리의 내용이나 시스템 정보 등은 악성코드의 공격 목표가 될 수 있다. 또한 속성은 악성코드가

Table 1. Example of mapping of network object

형태	이름	목표	속성
네트워크	전송되는 데이터	디바이스 ID	13F7898990705EE3DDFE64DE0E5F0709
		디바이스 형태	Android
		IMEI	357242043237517
		IMSI	310005123456789
		MCC	310
		MNC	260
		운영체제 버전	10
		SDK 버전	2.3.4
		국가코드	en
		위치	US
	변환되는 데이터	암호화 모드	No, DES, AES, Blowfish
		목표 전송 URL	http://my365image.com
		포트	80
		인코딩 모드	gzip

목표로 삼고 있는 민감한 개인 정보들을 말하며, 국가 코드나 유심 정보 등을 가리킨다. 오퍼레이션을 기호화하면 다음과 같다.

Operation ::= {Operation-name : {Operation-target : Operation-attribute}}
Operation-name ::= Sending SMS | Calling number | Sending data | Converting data
Operation-target ::= Premium-rate SMS/number | Device ID | IMEI | IMSI | .... | etc

예를 들어 악성코드가 수집하는 개인정보 및 수집 정보가 Table 1.과 같다면, 이를 프로파일링 기호로 다음과 같이 표현할 수 있다.

{Network : {Sending data : {{IMEI : 357242043237517}, {MCC : 310}, {MNC :260}, ... , }}}

III. Andro-profiler 시스템

3.1 개 요

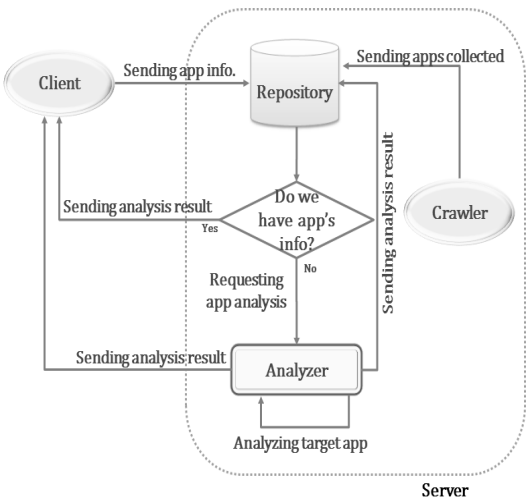

Fig.1. Overall procedure of Andro-profiler

Fig.1.에서와 같이 본 논문에서 제시한 방법인 Andro-profiler는 서버/클라이언트 방식으로 모바일 기기인 클라이언트와 분석 및 결과를 알려주는 서버로 구성되어 있다. 클라이언트인 모바일 폰에서 사용자가 악성행위가 의심되는 앱의 apk파일과 apk파일의 해쉬값을 서버에 전송하게 되면, 서버는 클라이언트로부터 전송받은 의심이 가는 앱의 해쉬값과 apk 파일을 기반으로 악성행위 유무를 판단한다. 만약 악성코드로 판단될 경우 기존에 데이터베이스화 한 악성코드와의 유사도를 비교하여 유사도가 높은 악성코드군으로 분류하고, 진단결과를 클라이언트로 전송한다. 전송된 정보는 사용자의 스마트폰 화면에 전시되어, 사용자가 해당 앱의 삭제여부를 결정하는데 보조역할을 한다.

분석 서버의 경우 크게 크롤러, 저장소(repository), 분석기의 3가지의 요소로 구성되어 있다. 크롤러는 평소에는 P2P사이트나 정상 앱 마켓 등으로부터 APK 파일을 지속적으로 수집하고, 수집된 파일은 서버 내에 저장소에 저장되게 된다. 저장소에 저장된 APK 파일들은 다시 분석기에 보내져 분석을 거치게 되고. 분석이 완료된 후 결과를 다시 저장소에 전송하게 된다. 저장소는 전송받은 결과들을 데이터베이스화하여 가지고 있다가 클라이언트로부터 APK 파일을 전송받게 되면 APK 파일의 해쉬값으로 결과를 찾게 되고 없다면 바로 분석기로 보내지게 된다. 분석기의 경우 통합 시스템 로그의 추출과정과 의사 결정 처리(decision process) 과정 이렇게 2가지로 나눠지는 데 전체적인 분석기의 구성은 Fig.2.와 같으며, 다시 의사 결정 처리 과정은 행위 프로파일 모듈, 유사도 모듈, 행위 범주화 모듈 이렇게 3가지로 나눠진다.

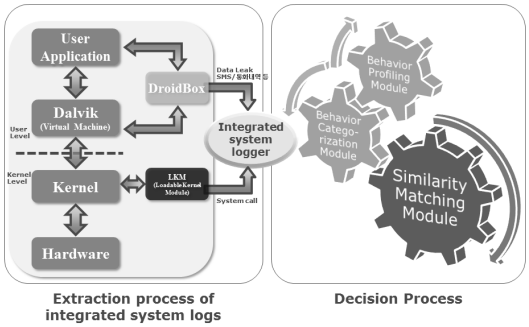

Fig.2. Overview of the analyzer component

3.2 통합 시스템 로그의 데이터 추출 과정

Andro-profiler는 동적 분석을 사용하여 악성코드를 분석하게 된다. 동적 분석은 에뮬레이터와 같은 고립된 환경을 구축하여 거기서 실제 악성코드를 실행시켜 이후 발생되는 변화들을 분석하는 분석방법이다. Andro-profiler는 드로이드 박스(Droidbox)라는 샌드박스 환경을 에뮬레이터 내에 구축하여 호출되는 API 함수들이나 네트워크 패킷, 데이터의 유출 등을

모니터링 하여 로그로 남기도록 구현하였다[4]. 또한 악성코드 제작자가 API 호출 사실을 숨기고 정상적인 행위로써 악성코드의 악성행위를 숨기기 때문에 어플리케이션의 행위를 파악하기가 어렵다. 그래서 정확한 어플리케이션의 행위를 조사하기 위해 시스템 콜 후킹을 통하여 어떤 시스템 함수가 어떤 매개변수를 가지고 호출되는지를 로그로 기록하게 구현하였다. 아무리 API 함수 호출 사실을 숨긴다고 하더라도 어떤 API 함수를 호출하든지 간에 시스템 함수를 무조건 호출하기 때문에 시스템 함수 호출 사실을 기록하여 분석하면 정확한 행위를 파악할 수 있다. 시스템 콜 후킹은 LKM(Loadable Kernel Module)을 구현하여 드로이드 박스가 설치된 안드로이드 에뮬레이터에 실행함으로써 안드로이드의 커널 후킹을 통하여 드로이드 박스 로그와 함께 기록되도록 하였다. 즉, 악성코드를 에뮬레이터 내에서 실행을 시키게 되면 드로이드 박스 로그와 시스템 콜 로그가 동시에 기록되는 통합 시스템 로그를 얻게 된다. 그리고나서 통합 시스템 로그를 파싱하여 데이터를 의사 결정 처리 과정을 통하여 행위 프로파일링으로 변환하고 다른 프로파일링 데이터와 유사도 측정을 통하여 악성유무 확인과 유사 패밀리를 알아낼 수 있다.

3.3 의사 결정 처리(Decision Process)

의사 결정 처리 부분은 크게 행위 프로파일 모듈과 행위 범주화 모듈 그리고 유사도 모듈 이렇게 3가지로 구성되어 있다. 각 모듈의 자세한 설명은 아래와 같다.

행위 프로파일 모듈(Behavior Profiling Module) 은 악성코드의 통합 시스템 로그를 파싱하고, 파싱된 데이터로부터 행위 프로파일링을 추출하는 역할을 수행한다. 행위 프로파일 모듈의 구현은 3장에서 설명된 구체적인 형태로 행위 프로파일링을 추출하도록 되어있다.

행위 범주화 모듈(Behavior Categorization Module)은 악성코드의 행위 패턴에 따라서 악성코드를 범주화 한다. 본 논문에서는 악성행위를 크게 4가지로 정의하였다. 과금을 유발할 수 있는 번호로 전화를 거는 행위, 과금을 유발할 수 있는 번호로 SMS를 보내는 행위, 민감한 개인정보를 수집하는 행위, 데이터를 전송하기 위해 가공하는 행위가 바로 그것이다. 4가지의 악성행위 패턴은 총 15가지 조합으로 악성행위 범주화를 만들 수 있고, 만약에 사전에 정의한 4가지의 악성행위를 하지 않을 경우 정상으로 간주하여 처리한다.

유사도 모듈(Similarity Matching Module)은 새로 입력된 어플리케이션의 행위 프로파일과 각 악성코드 패밀리 내 대표 행위 프로파일링들과 유사도를 측정하여, 가장 유사도가 높게 나온 패밀리로 분류하는 역할을 한다. 각 악성코드 패밀리 내 대표 행위 프로파일링은 패밀리 내 서브그룹에 속해있는 맴버들간의 공통적으로 존재하는 행위를 업데이트를 한다. 즉, 한 악성코드 패밀리에 대한 대표 행위 프로파일링은 각 악성코드 패밀리 내에서 공통적인 행위 특성을 나타내는 것이다.

본 논문에서는 유사도 점수(Similarity score)를 전화나 카메라 같은 하드웨어 자원, 시스템 정보와 개인 정보에 얼마나 접근할 수 있는지에 대한 수치로써 정의하였다. 각 악성코드 패밀리의 대표적인 행위 프로파일과 새로운 악성코드의 행위 프로파일의 유사도를 구하는 수식은 아래와 같다.

$$E(SS) = \sum_i w_i \cdot BFS_i, \quad \sum_i w_i = 1 \tag{1}$$

BFS_i와 w_i는 각각 유사성과 I번째 행위의 가중치를 나타낸다. BFS(Behavior Factor Similarity)는 악성 행위를 나타내는 4가지의 파트로 구성되어 있다. 과금을 유발시킬 수 있는 SMS 보내는 행위에 대한 유사성(SS: Similarity of Sending premium-rate SMS), 과금을 유발시킬 수 있는 번호로 전화를 거는 행위에 대한 유사성(CS: Similarity of Calling premium-rate number), 민감한 정보를 수집하는 행위에 대한 유사성(SIS: Similarity of Sensitive Information) 그리고 데이터를 암호화나 인코딩을 통하여 변환시켜 서버에 전송하는 행위에 대한 유사성(CDS: Similarity of Converting Data)으로 구성되어 있다. 그리고 각 유사성에 대한 가중치는 각각 SS는 0.35, CS는 0.35, SIS는 0.2, CDS는 0.1로 결정하였으며, 모든 가중치는 여러 번의 실험 결과를 토대로 가장 잘 유사도가 구분이 될 수 있는 값을 선정하였다. 몇몇 악성코드는 오직 특정한 조건이 만족되어야 실행되기 때문에 CDS의 가중치를 다른 요소들보다 적게 결정하였다. 악성코드가 실행되지 않거나 통신 서버가 막혀있어 통신이 안된다면 네트워크를 통한 데이터 포맷을 변경할 수 없기 때문이다. 아래 Table 1.은 각 요소들의 유사도 점수를 계산하는 방법을 나타낸 것이다.

Table 2. Similarity metric to apply to each behavior factor

행위 요소	행위 목표	유사도 기준
과금 문자 (SS)	과 금	Binary (0 or 1)
과금 전화 (CS)	과 금	Binary (0 or 1)
전송 정보 (SIS)	시스템 정보, 개인 정보 수집	Jacard index [0, 1]
전송을 위한 데이터 변환 (CDS)	목적지 URL	Longest prefix matching + Levenshtein distance [0, 1]
	암호화 모드 (DES, AES, Blowfish)	Binary (0 or 1)
	인코딩 모드 (Gzip or not)	Binary (0 or 1)

각 요소들에 대한 구체적인 계산방법은 다음과 같다.

1. 과금 번호로 SMS를 보내는 SS와 과금 번호로 전화를 거는 CS의 유사도 점수는 해당 하드웨어 자원에 접근할 수 있느냐 없느냐를 바이너리로 비교한다. 전화번호와 같은 문자열은 한 개의 번호라도 다르다면, 완전히 다른 전화번호가 되기 때문에 완벽한 문자열 매칭(perfect matching)을 제외한 부분 매칭은, 유사도 비교측면에서는 유의한 의미를 부여하지 못한다. 따라서 결국 의심스런 앱의 프로파일링 정보와 군집화된 그룹의 대표 프로파일링 정보들 중 CS와 SS의 항목에서 동일한 행위를 한다면 1, 그것이 아니라면 0을 부여한다.

2. 민감한 데이터를 전송하는 SIS는 Jaccard index 방법을 이용하여 유사도 점수를 계산한다.

3. 네트워크로 데이터를 전송하기 위해 데이터를 인코딩이나 암호화를 통하여 변경시키는 CDS는 목적지 URL 주소, 암호화 모드와 인코딩 모드에 대한 유사성을 각각 구하여 더한 평균값을 유사도 점수로 계산한다. 목적지 URL 주소의 유사도는 Longest Prefix Matching 방법을 적용한 후 불일치되는 부분에 대해서는 Levenshtein distance 방법을 적용한다. 예를 들어 www.abc. com과 www.def.net을 비교한다고 했을 때 Longest Prefix Matching 방법으로 두 URL의 일치되는 부분인 www을 구하게 된다. 나머지 불일치되는 부분인 abc.com과 def.net에 대한 유사도는 Levenshtein distance 방법을 이용하여 구하게 되며, 0에서 1사이의 값이 나오게 된다.

암호화 모드와 인코딩 모드에 대해서는 바이너리로 비교하여, 동일한 행위를 한다면 1, 그것이 아니라면 0을 부여한다.

4. 유사도 비교시, SS, CS, SIS가 Null인 경우는 정상파일로 간주하여, 해당 가중치를 0으로 두어 유사도를 계산하지 않고, CDS만 유사도를 비교한다. 이 때 CDS의 가중치는 0.1로 한다. SIS와 CDS에 사용되는 정보들은 아래와 같다.

```
Sensitive information ::= System information | Private
                          information
Converting data ::= Destination URL | Encryption
                    mode | Encoding mode
System information ::= IMEI | IMSI | Device id | Device type |
                       Device model | OS Version | Carrier |
                       MCC | MNC | .... |
Private information ::= External storage contents | Language |
                        GPS information | Country code | ... |
Encryption mode ::= Nothing | DES | AES | Blowfish
Encoding mode ::= Nothing | gzip | ... |
```

위와 같은 방법으로 구해진 각각의 BFS 점수들에 가중치를 곱한 후 더하면 전체적인 유사도 점수가 구해지게 된다.

Table 3. Malware samples for experiments

구 분	악성코드 군	샘플수	행위 특징
악성코드 (372개)	AdWo	100	개인정보 수집
	AirPush	60	SMS 전송, 개인정보 수집
	FakeBattScar	52	과금번호 전화, 개인정보 수집, SMS 전송
	Boxer	100	SMS 전송, 개인정보 수집
	FakeNotify	60	SMS 전송
정 상 (350개)	Application	327	유틸, 은행 등 앱
	Game	23	게임 앱

IV. 실험 및 성능 평가

4.1 실험 환경 설정

본 논문의 실험을 위해 2013년 1월부터 8월까지의 기간 동안 virusshare [16], contagion [3], malware.lu [11]와 같은 악성코드 저장소와 크롤러로 악성코드 샘플을 수집하였다. 또한 같은 기간 동안 GooglePlay 사이트에서 웹 크롤러를 통하여 정상 샘플을 수집하였다. 변종을 포함한 악성코드의 개수는 372개이고 정상 샘플은 350개정도를 수집하였다. 그리고 수집된 샘플들의 SHA256 해쉬값을 구하여 동일한 해쉬값을 가진 샘플들은 삭제하는 방법으로 중복되는 샘플들을 제거하였다. 정상 샘플들은 패키지 이름을 가지고 중복 샘플들을 확인하여 제거하였다.

VirusTotal 데이터 셋[17]을 포함하여 최소 9개 이하의 백신 회사들이 진단한 악성코드 샘플을 제외시켰다. 그리고 F-Secure[8]에서 만든 악성코드에 대한 설명을 참조하였다. 이러한 악성코드들과 정상 샘플들에 대한 자세한 설명은 Table 3.에 나열하였다. 그리고 본 연구에서 Andro-profiler가 알려지지 않은 악성코드들을 적절하게 분류하고 탐지기능이 얼마나 정확한지를 평가하기 위해 ADAM[19]이라는 변종 제작 프로그램을 이용하여 변종 50개를 만들었다. 이렇게 만들어진 변종은 F-Secure에 의해 탐지되지 않았고, 다른 백신회사들도 5군대 미만으로만 진단이 되었다.

Andro-profiler는 드로이드 박스[4]와 파이선 프로그래밍 언어로 구현되었다. 아래 Fig.3.은 FakeBattScar 악성코드를 구현된 Andro-profiler로 프로파일링 데이터를 추출한 결과이다. FakeBattScar 악성코드는 과금을 유발할 수 있는

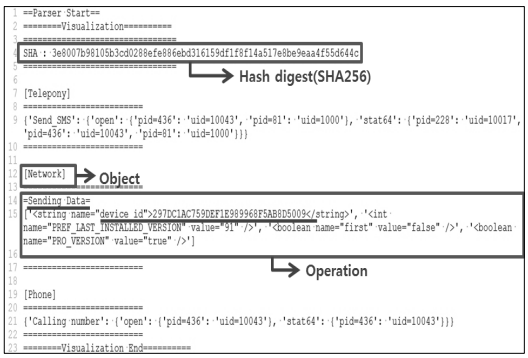

Fig.3. Behavior profiling of FakeBattScar

번호로 전화를 걸려는 시도가 있었고, 모바일 기기의 고유한 ID와 같은 민감한 정보를 수집하는 행위를 하였음을 알 수 있다.

본 논문에서 진행된 실험은 인텔 코어 i7(2GB) CPU와 32비트 우분투 12.04 LTS버전의 운영체제 그리고 4기가 램이 설치된 컴퓨터에서 진행되었다. 실험에서 성능을 평가하기 위해 10-fold cross-validation을 사용하였다. k-fold validation은 무작위로 k개의 크기의 서브 샘플들로 데이터 셋의 파티션을 나누는 방법이다. 하나의 서브 샘플은 테스트를 위한 유요한 데이터로서 사용되고 그 외에 k-1개의 서브 샘플들은 트레이닝 데이터로서 사용된다. 10-fold cross-validation을 본 연구의 실험에서는 10번을 반복 수행하였으며, 10번의 수행 결과들의 평균을 최종 결과로 선정하였다.

4.2 기존 연구와의 비교

본 논문에서 제안하는 방법론은 어플리케이션이 실행되는 동안 호출되는 시스템콜을 포함한 통합 시스템 로그를 이용하는 것으로, 기존의 방법론 중 가장 유사한 방법론은 Crowdroid [15]이다. Crowdroid는 클라이언트에서 호출되는 시스템콜을 모니터링 하여 시스템콜 빈도수 테이블을 작성하고, 서버에서는 해당 빈도수 테이블을 이용하여 K-means 알고리즘을 통해 악성여부를 분류한다.

4.3 실험 결과

본 연구의 실험 결과는 행위 프로파일링의 효과와 악성코드 분류 성능에 집중하였다.

첫 번째로 악성코드와 악성코드간의 유사도 그리고 악성코드와 정상 앱에 대한 유사도를 구하였다. Table 4.는 다른 악성코드 패밀리간의 유사도를 나타낸 것이다. Table 4.에서 FakeBattScar에 대한 유사도의 차이가 다른 패밀리들과 비교했을 때 좀 더 크게 차이가 난다. FakeBattScar는 오직 과금 번호로 전화를 걸고 민감한 데이터를 수집하는 악성 행위를 하는 데 다른 악성코드 패밀리들은 SMS를 보내고 민감한 데이터를 수집하는 악성행위를 하기 때문에 차이가 나는 것이다. 즉, FakeBattScar를 제외한 다른 패밀리끼리는 악성행위가 단순하고 유사하기 때문에 상대적으로 다른 패밀리끼리의 유사도 차이는 적게 나지만 FakeBattScar만 차이가 크게 나는 것이다. 그

리고 Table 5.는 악성코드와 정상 샘플간의 유사도를 나타내는 것인데, 정상 샘플의 경우 악성행위를 하지 않기 때문에 악성코드간의 유사도 차이가 크게 난다.

Table 4. The similarity comparison with different malware families

Malware	AdWo	AirPush	Boxer	FakeNotify	FakeBattScar
AdWo	1.0	0.7	0.7	0.7	0.4
AirPush	0.7	1.0	0.8	0.8	0.45
Boxer	0.7	0.8	1.0	0.8	0.45
FakeNotify	0.7	0.8	0.8	1.0	0.45
FakeBattScar	0.4	0.45	0.45	0.45	1.0

Table 5. The similarity comparison with different malware and benign samples

Benign \ Malware	AdWo	AirPush	Boxer	FakeNotify	FakeBattScar
Productivity	0.06	0.1	0.1	0.1	0.1
Business	0.06	0.1	0.1	0.1	0.1
Education	0.06	0.1	0.1	0.1	0.1
Shopping	0.06	0.1	0.1	0.1	0.1
Game	0.06	0.1	0.1	0.1	0.1

두 번째로 Andro-profiler가 악성 행위를 잘 탐지하고 악성 행위의 특성에 따라 잘 분류하는지를 확인하였다. 성능을 평가하기 위한 지표로써 모델의 예측 능력에 집중해야 하기 때문에 성능측정 요소로써 정확도를 사용하였다. 악성코드 분류모델의 성능은 어떻게 잘 그 모델이 다양한 악성코드를 분류하느냐에 따라 결정된다.

아래의 Table 6.은 Andro-profiler가 어떻게 잘 악성코드를 탐지하고 분류하는 지를 보여주고 있는데

Table 6. Classification accuracy for 372 malware, 350 benign samples

Category		Accuracy	Sum	Correct	Incorrect
Malware	Boxer	**1.00**	100	100	0
	FakeBattScar	**1.00**	52	52	0
	AdWo	**1.00**	100	100	0
	AirPush	0.95	60	57	3
	FakeNotify	**1.00**	60	60	0
Benign	Application	0.92	327	301	26
	Game	**1.00**	23	23	0
Average		0.96	722	693	29

Table 7. Classication performance for 372 malware and 350 benign samples

Category		Accuracy(%)	
		Andro-profiler	Crowdroid
Malware	Boxer	**100**	2
	FakeBattScar	**100**	92
	AdWo	**100**	65
	AirPush	95	6
	FakeNotify	**100**	15
Benign	Application	92	27
	Game	**100**	41
Average		96	37

평균 96%의 분류 정확도를 기록하였다. Andro-profiler는 거의 99%의 정확도를 가지고 적절한 악성코드 패밀리로 분류할 수 있지만 정상 어플리케이션을 정상으로 분류하는데 약 7%정도의 오탐(false positive)이 발생하였다. 그래서 정상 어플리케이션을 분류할 때 높은 거짓 양성이 발생했는지를 알아내기 위해 실험을 통하여 원인을 분석하였다. 결과적으로는 몇몇의 정상 어플리케이션이 사용자의 모바일 기기에 장비 ID, SDK 버전, 모델명과 같은 민감한 정보를 수집한다는 사실을 발견하였다. 거짓 양성이 나온 DU battery saver라는 정상 어플리케이션을 예를 들면 사용자의 동의 없이 모바일 기기의 장비 ID를 수집하는 것을 확인하였다.

위 Table 7.은 Andro-profiler를 이용한 악성코드 분류 정확도와 기존 방법론인 Crowdroid의 악성코드 분류 정확도를 나타낸 표이다. Crow droid를 이용한 실험에도 Andro-profiler의 실험과 똑같은 악성코드 샘플들을 가지고 실험을 실시하였다. 실험결과를 보았을 때 비교대상인 Crowdroid에 비해서 높은 분류 정확도를 보여주고 있다.

본 논문에서 제시한 시스템은 1MB의 APK파일을 가진 목표 어플리케이션을 분류하는데 55초를 기록하였다. 물론 에뮬레이터의 부팅시간과 같은 분석을 위해 세팅되는 시간을 제외한 순수하게 분석 및 분류되는 시간을 기록한 것이다. 또한 기록된 시간의 대부분은 행위 프로파일러를 만드는데 소비되었으며, 분류하는데 걸린 시간은 0.2초를 기록하였다.

V. 결 론

본 논문에서는 악성행위를 프로파일링하여 악성코

드 탐지 및 분류를 하는 방법론(Andro-profiler)을 제안하였다. 행위기반의 프로파일링은 드로이드 박스 로그와 시스템 콜 로그가 합쳐진 통합 시스템 로그에 의해 만들어 진다.

악성행위에 대한 프로파일링은 행위 프로파일 모듈에 의해 통합 시스템 로그를 파싱하여 악성 행위패턴을 추출하고, 행위 범주화 모듈이 추출된 악성 행위 패턴을 가지고 악성행위에 대한 특징을 범주화시킨다. 유사도 모듈은 새로운 악성 앱과 기존 데이터베이스화된 악성코드 군의 대표 행위 프로파일링 사이의 유사도 점수를 계산하여, 새로운 악성코드와 가장 유사도 점수가 높은 악성코드 군으로 분류한다.

앞에서 시행한 실험결과를 통해 제안된 시스템은 평균적으로 96%의 정확도로 악성코드군과 정상파일을 분류하였다. 제안된 시스템은 악성코드간 분류실험에서 99%의 정확도로 악성코드군을 분류하였고, 정상앱과 악성코드가 구성된 데이터셋에서는 93%의 정확도로 악성코드 및 정상코드를 분류하였다. 한편, 제안한 시스템은 평균적으로 1MB의 앱을 58초 안에 통합시스템 로그를 파싱 및 분류 가능하여, 악성코드 샘플들에 대해 신속하고, 효율적인 대처가 가능하다.

본 논문에서 제안된 동적 분석 방법은 악성코드 정상적인 실행가능 여부가 중요하다. 최근 보고되는 신종 악성코드의 경우에는 쉽게 에뮬레이터 등 동적분석 툴을 감지하여 실행을 중단하는 안티 가상머신 기능이 탑재되어 있다. 따라서 향후에는 Andro- profiler에 정적 분석 기법을 확대 적용하여 일반화된 프레임워크를 제공할 것이다.

References

〔1〕 "AV-TEST Examines 22 Antivirus Apps for Android Smartphones and Tablets", Av-Test, Accessed August 13, 2013, http://www.av-test.org/fileadmin/pdf/avtest_2013-01_android_testreport_english.pdf

〔2〕 Bayer, U. and Comparetti, P.M., Hlauschek, C, Kruegel, C, "Kirda, E.: Scalable, Behavior-Based Malware Clustering," NDSS, pp. 8 - 11, 2009

〔3〕 Contagion blog, http://contagiominidump.blogspot.kr/

〔4〕 Droidbox Android Application Sandox, http://code.google.com/p/dr oidbox/

〔5〕 Enck, W., McDaniel, P. and Chaudhuri, S, "A study of android application security," Proceedings of the 20th USENIX conference on Security, pp. 21-21, 2011

〔6〕 Enck, W., Ongtang, M. and McDaniel, P., "On lightweight mobile phone applica tion certification," Proceedings of the 16th ACM conference on Computer and communications security, pp. 235 - 245, 2009

〔7〕 FakeBattScar, https://www.f-secure.com/v-descs/trojan_android_fakebattscar.shtml

〔8〕 F-Secure.com, http://www.f-secure.com/en/web/labs_global/

〔9〕 "Gartner Says 821 Million Smart Devices Will Be Purchased Worldwide in 2012; Sales to Rise to 1.2 Billion in 2013," Nov 6, 2012, Accessed August 13, 2013, http://www.gartner.com/newsroom/id/2227215

〔10〕 Kocsis, R.N., "Applied criminal psychology: A guide to forensic behavioral sciences," Charles C Thomas Publisher, 2009

〔11〕 Malware.lu, http://malware.lu/

〔12〕 "Mobile Threat Report Q4 2012," F-Secure, Accessed August 13, 2013, http://www.f-secure.com/static/doc/labs_global/Research/Mobile%20Threat%20Report%20Q4%202012.pdf

〔13〕 Nick Nykodym, Robert Taylor and Julia Vilela, "Criminal profiling and insider cybercrime: Digital Investigation," pp 261-267, 2005

〔14〕 Pearce, P., Felt, A.P., Nunez, G., Wagner, D., "Addroid: Privilege sepa ration for applications and advertisers in android," Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security, pp. 71 - 72, 2012

〔15〕 Burguera, I., Zurutuza, U., Nadjm-Tehrani, S., "Crowdroid: behavior-based malware detection system for Andr oid," Proceedings of the 1st ACM work shop

on Security and privacy in sma rtphones and mobile devices, pp. 15 - 26, 2011
〔16〕 Peng, H., Gates, C., Sarma, B., Li, N., Qi, Y., Potharaju, R., Nita-Rotaru, C., Molloy, I., "Using probabilistic generative models for ranking risks of Android apps," Proceedings of the 2012 ACM conference on Computer and communications security, pp. 241 - 252, 2012
〔17〕 Virusshare, http://virusshare.com
〔18〕 VirusTotal, http://www.virustotal.com
〔19〕 Yang, C., Yegneswaran, V., Porras, P., Gu, G., "Detecting money-stealing apps in alternative Android markets," Proceedings of the 2012 ACM conference on Computer and communications security, pp. 1034 - 1036, 2012
〔20〕 Zheng, M., Lee, P.P., Lui, J.C., "Adam: An automatic and extensible platform to stress test android anti-virus systems. Detection of Intrusions and Malware, and Vulnerability Assessment," Springer, pp. 82-101, 2013
〔21〕 Zhou, Y., Wang, Z., Zhou, W., Jiang, X., "Hey, you, get off of my market: Detecting malicious apps in official and alternative android markets," Proceedings of the 19th Annual Network and Distributed System Security Symposium, pp. 5-8, 2012

〈저자소개〉

윤 재 성 (Jae-sung Yun) 학생회원
2012년 2월: 성균관대학교 컴퓨터공학과 졸업
2012년 3월~현재: 고려대학교 정보보호대학원 석사과정
〈관심분야〉 시스템보안, 모바일 기기 보안, 악성코드 분석, 데이터 마이닝, 패턴인식

장 재 욱 (Jae-wook Jang) 학생회원
2004년 2월: 해군사관학교 전산과학과 졸업
2010년 2월: 서울대학교 전자·컴퓨터공학부 석사
2012년 3월~현재: 고려대학교 정보보호대학원 박사과정
〈관심분야〉 악성코드 분석, 네트워크 보안, 데이터 마이닝, 패턴인식

김 휘 강 (Huy Kang Kim) 종신회원
1998년 2월: KAIST 산업경영학과 졸업
2000년 2월: KAIST 산업공학과 석사
2009년 2월: KAIST 산업및시스템공학과 박사
2004년 5월~2010년 2월: NC소프트 정보보안실장, Technical Director
2010년 3월~현재: 고려대학교 정보보호대학원 조교수
〈관심분야〉 온라인게임 보안, 네트워크 보안, 네트워크 포렌직, 침입탐지시스템, 봇넷탐지