ISSN 1598-0170 (Print)
ISSN 2287-1136 (Onine)
http://www.jksii.or.kr

안드로이드 플랫폼에서 악성 행위 분석을 통한 특징 추출과 머신러닝 기반 악성 어플리케이션 분류☆

Malware Application Classification based on Feature Extraction and Machine Learning for Malicious Behavior Analysis in Android Platform

김 동 욱[1] 나 경 기[1] 한 명 묵[1*] 김 미 주[2] 고 웅[2] 박 준 형[2]
Dong-Wook Kim Kyung-Gi Na Myung-Mook Han Mijoo Kim Woong Go Jun Hyung Park

요 약

본 논문은 안드로이드 플랫폼에서 악성 어플리케이션을 탐지하기 위한 연구로, 안드로이드 악성 어플리케이션에 대한 위협과 행위 분석에 대한 연구를 바탕으로 머신러닝을 적용한 악성 어플리케이션 탐지를 수행하였다. 안드로이드의 행위 분석은 동적 분석 도구를 통해 수행할 수 있으며, 이를 통해 어플리케이션에 대한 API Calls, Runtime Log, System Resource, Network 등의 정보를 추출할 수 있다. 이 연구에서는 행위 분석을 통한 특징 추출을 머신러닝에 적용하기 위해 특징에 대한 속성을 변환하고, 전체 특징에 대한 머신러닝 적용과 특징들의 연관분석을 통한 주성분분석으로 특징간의 상관분석으로 얻은 머신러닝 적용을 수행하였다, 이에 대한 결과로 악성 어플리케이션에 대한 머신러닝 분류 결과는 전체 특징을 사용한 분류 결과보다 주요 특징을 통한 정확도 결과가 약 1~4%정도 향상되었으며, SVM 분류기의 경우 10%이상의 좋은 결과를 얻을 수 있었다. 이 결과를 통해서 우리는 전체적인 특징을 이용하는 것보다, 주요 특징만을 통해 얻을 결과가 전체적인 분류 알고리즘에 더 좋은 결과를 얻을 수 있고, 데이터 세트에서 의미있는 특징을 선정하는 것이 중요하다고 파악하였다.

☞ 주제어 : 안드로이드, 행위 분석, 특징 추출, 상관 분석, 악성 어플리케이션 분류

ABSTRACT

This paper is a study to classify malicious applications in Android environment. And studying the threat and behavioral analysis of malicious Android applications. In addition, malicious apps classified by machine learning were performed as experiments. Android behavior analysis can use dynamic analysis tools. Through this tool, API Calls, Runtime Log, System Resource, and Network information for the application can be extracted. We redefined the properties extracted for machine learning and evaluated the results of machine learning classification by verifying between the overall features and the main features. The results show that key features have been improved by 1~4% over the full feature set. Especially, SVM classifier improved by 10%. From these results, we found that the application of the key features as a key feature was more effective in the performance of the classification algorithm than in the use of the overall features. It was also identified as important to select meaningful features from the data sets.

☞ keyword : Android, Behivavr Analysis, Feature Extraction, Correlation Analysis, Malware Application Classification

1. 서 론

구글은 2008년 최초로 공개한 모바일 OS '안드로이드'를 통해 전 세계적으로 많은 인기를 끌었으며, 가트너 보고서에 의하면 2017년 1분기 안드로이드 기반 모바일 기기의 시장점유율이 약 86% 가량으로 가장 높다[1]. 안드로이드가 빠른 시간에 가장 큰 성장을 이루게 된 이유는 '오픈소스 플랫폼', '어플리케이션 개발 용이', '공개마켓을 통한 어플리케이션 유포' 등의 개방형 구조를 이점으로 한다. 하지만 안드로이드의 개방형 구조는 보안에 매우 취약하여 수많은 사이버공격을 초래하게 되었다. 누구나 모바일 악성코드를 제작하고 배포할 수 있으며, 구글

1 Department of Computer Engineering, GachonUniv, Seongnam si, 13120, Korea.
2 SecurityR&D Team 1, KOREA INTERNET& SECURITY AGENCY. Naju-si, 58324Korea.
* Corresponding author (mmhan@gachon.ac.kr)
[Reviewed 11 December 2017, Reviewed 16 December 2017, Accepted 25 December 2017]
☆ This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2015-0-00003, Development of Profiling-based Techniques for Detecting and Preventing Mobile Billing Fraud Attacks)]

이 직접 운영하는 공식 마켓인 'Google Play Store'에서도 배포되는 어플리케이션에 대해 아무런 사전 검수가 이루어지지 않는다. 이러한 이유로 안드로이드 환경에서의 모바일 악성코드는 최근 몇 년간 기하급수적으로 증가하고 있다[2]. 안드로이드 악성코드의 제작 방법으로는 다양한 방법이 존재하는데, 그 중에서 리패키징(repackaging) 기법이 가장 많이 사용된다. 리패키징 기법은 정상 어플리케이션에 일부 악성 기능을 추가하여 정상 어플리케이션처럼 위장하거나 기존 안티 바이러스 시스템을 우회하기 위해 기존 악성코드를 일부 수정하고 이를 다시 다양한 경로를 통해 배포하는 기법이다. 해커나 개발자들은 APK 파일의 압축을 해제하여 나온 바이너리 형태의 파일을 디컴파일(decompile)하여 소스 코드로 변환시킨다. 결과물로 나온 소스코드를 수정하여 다시 컴파일과 패키징 작업을 거치면 새로운 APK 파일이 생성된다. 마지막으로 개발자 인증서로 코드 사인 과정만 거치면 실행 가능한 어플리케이션이 완성되는데, 개발자 인증서는 APK 파일을 압축 해제하는 언패킹 과정에서 추출할 수 없으므로 새로운 인증서로 코드 사인을 해야 한다[3].

최근 연구 결과에 따르면, Android Malware Genome Project에서 수집한 안드로이드 악성코드 샘플 1260개 중에서 리패키징된 악성코드의 비율은 86%로 다양한 악성코드 유형들 중에 비율이 가장 높았다[4]. 그리고 TrendMicro 사에서 진행한 연구 결과에 따르면 Google Play Store에 등록된 다양한 카테고리의 상위 50개 어플리케이션 가운데 약 77%의 어플리케이션이 리패키징 기법을 통한 변종 어플리케이션이었다고 밝혔다[5]. 이렇게 날이 갈수록 발전하는 안드로이드 악성코드를 막기 위해 구글은 2012년 2월 자사에서 운영중인 Google Play Store에 적용하는 악성 어플리케이션 분석 시스템인 '바운서(Bouncer)'를 발표하였으며, 이 시스템의 도입으로 악성코드 발생량이 약 40% 가량 감소하였다고 밝혔다[6]. 바운서는 Google Play Store에 어플리케이션이 등록되면 구글 인프라 기반의 에뮬레이터에서 동적 분석을 수행하며, 일정 시간동안 어플리케이션을 실행하여 해당 어플리케이션의 악성 행위 여부를 판단하는 시스템이다. 대부분의 안드로이드 악성코드는 어플리케이션이 실행되어야 악성 여부를 판단이 가능해지기 때문에, 정적 분석보다는 동적 행위 분석을 통한 악성코드 탐지 방법이 더 효율적이다. 동적 행위 정보 기반의 악성코드 탐지 방법은 시스템에서 발생하는 프로세스 행동과 미리 정해진 공격의 패턴을 비교 분석하여 침입을 탐지하는 방법이다. 안드로이드 스마트폰을 사용할 때 발생되는 자원 소모량 등의 이벤트 정보를 모니터링하고 정보를 수집하여 악성어플리케이션의 비정상적인 패턴을 탐지한다[7].

따라서 본 논문에서는 행위 기반 분석 정보를 이용한 방법으로 악성 어플리케이션을 탐지하는 기법을 제안한다. 2장에서는 악성 어플리케이션 동적 행위 정보 분석에 관련된 기존 연구에 대해 살펴보며, 3장에서는 어플리케이션의 행위 기반 분석 결과로 얻을 수 있는 특징을 설명한다. 4장에서는 실제 악성 어플리케이션 분류를 위해서 본 연구에 대한 제안 기법을 소개하고, 5장에서는 머신러닝을 이용한 성능을 평가한다, 마지막으로 6장에서는 결론을 맺고 향후 연구 방향을 제시한다.

2. 안드로이드 행위 분석 연구와 위협

2.1 관련 연구

Andromaly[8]에서는 안드로이드 기반 모바일 장치에 대한 악성코드 탐지 시스템을 제안한다. 사용자가 안드로이드 기반의 모바일 장치를 감지하고, 기기에서 발생하는 위협적인 행동에 대한 탐지를 돕는다. 악성코드 탐지 프로세스에서 수집되는 데이터는 CPU 소비, Wi-Fi를 통한 전송 패킷 수, 실행중인 프로세스 수, 배터리 수준과 같은 실시간 모니터링 수집, 전처리 및 다양한 시스템 사용 파라미터 등을 수집하여 기계학습을 이용하여 악성코드 탐지를 수행한다. 하지만 실험 결과 구체적으로 악성코드 여부를 판단하지 못하는 결과가 나타났다.

Crowdroid[9]는 '크라우드소싱'의 개념을 차용한 경량 클라이언트로써, 리눅스 커널의 시스템 호출을 모니터링하고, 수집된 로그를 중앙 서버로 전송한 후 아웃소싱을 통해 악성코드를 탐지하는 방법이다. 전송받은 데이터를 중앙 서버에서 구문 분석을 진행한 후 시스템 호출 벡터를 생성하고 k-means 알고리즘을 통해 데이터를 클러스터링한다. Crowdroid 어플리케이션을 사용하는 사용자가 많아질수록 시스템이 더욱 정확해진다.

VirusMeter[10]는 모바일 디바이스는 일반적으로 배터리를 사용하여 구동되며, 악성 행위는 필연적으로 배터리를 소모한다는 가정 하에 제안된 방법이다. VirusMeter는 모바일 장치의 배터리 소모량을 모니터링함으로써 비정상적인 전력 소비로 이어지는 악성 행위를 포착한다. VirusMeter는 일반적인 사용자 행위의 전력 소모를 특성화하는 사용자 중심의 전력 모델을 사용한다. 실시간 모드에서는 간단한 런타임 오버헤드로 빠르게 악성코드 탐지를 할 수 있으며, 배터리 충전 중에는 보다 정교한 기계

학습 기술을 적용하여 탐지 정확도를 향상시킨다. 실험 결과에 따르면 VirusMeter는 1.5% 미만의 추가 전력 소비로 실시간 악성코드 탐지에 효과적인 결과를 보였다.

SmartSiren[11]은 프록시 기반의 아키텍처를 사용하여 리소스가 제한된 스마트폰에서 처리 부담을 줄여주고 스마트폰 간의 공동 작업을 단순화하는 바이러스 탐지 및 경고 시스템이다. 악성코드를 탐지하기 위해 스마트폰으로부터 SMS, 블루투스 등의 통신 활동 정보를 수집하고, 이 로그 데이터를 프록시 서버로 전송한다. 프록시 서버에서는 사용자의 평균 통신 활동 수치와 전달된 로그 데이터를 비교하여 악성코드의 침입 여부를 판단한다.

2.2 안드로이드 악성코드 유형

모바일 디바이스의 성장과 규모의 빠른 증가로 인해 모바일 환경의 위협 요인들도 다양화되어가고 있다. 그 원인으로는 악의적 목적을 가지고 악성코드 제작, 유통이 가능한 개방형 단말기 증가, 블루투스나 Wi-Fi, USB 등 외부 접속의 다양화를 뽑을 수 있다. 다양한 모바일 악성코드의 유형과 특징은 다음과 같다.

- Trojans : 정상적인 어플리케이션으로 가장한 악성 어플리케이션. 사용자의 기밀 정보를 유출하거나 사용자를 피싱하고, 암호와 같은 중요 정보를 훔치는 등 사용자의 동의 없이 악성 활동 수행[12].
- Backdoor : 시스템에 침입하는 다른 악성코드들의 일반적인 보안 절차에 대한 우회를 용이하게 한다. 백도어는 root 권한을 사용하여 수퍼유저 권한을 얻고, 안티바이러스 스캐너를 회피할 수 있다.
- Worm : 웜은 스스로 복사본을 만들어서 네트워크나 이동식 미디어를 통해 전파된다. 호스트 시스템에서 파일을 지우거나, 파일을 악의적으로 암호화하기도 하며, 자기 복제 과정에서 생성되는 네트워크 트래픽 자체만으로도 피해를 줄 수 있다.
- Botnet : 봇넷은 장치를 감염시켜 봇을 생성하며, '봇 마스터'라는 원격 서버에 의해 감염된 기기들이 원격으로 제어된다. 이는 은밀하게 뒤에서 실행되기 때문에 사용자 본인은 자신의 기기가 봇넷에 감염된 것을 모르는 경우도 많다. 특정 온라인 서버를 표적으로 DDoS 공격을 하거나 대규모 스팸을 전송
- Spyware : 사용자 동의없이 설치되어 사용자의 주민등록번호같은 신상정보, 신용카드같은 금융정보 등을 수집하여 원격서버로 보내는 행위를 한다.
- Aggressive Adware : GPS 정보로 위치서비스를 오용하여 사용자 장치에 광고를 보내거나, 홈 화면에 바로가기 생성, 북마크 도용, 기본 검색 엔진 설정 변경 등으로 불필요한 알림 푸시를 통해 기기 사용 방해[13].
- Ransomware : 사용자의 장치나 파일들을 암호화하여 온라인 지불 서비스를 통해 몸값을 지불할 때 까지 인질로 삼아 액세스를 불가능하게 하는 악성코드.[14]

2.3 분석 관련 도구

위와 같이 점점 악성코드 종류가 증가하고 고도화되어감에 따라 이에 대응하기 위한 여러 가지 분석 도구들도 개발이 활발하다. Virustotal과 RAPID7의 후원을 받아 개발된 프로젝트인 'Androguard'는 2011년 Blackhat 컨퍼런스에서 발표되었으며, 일반에 공개되었다. 악성 어플리케이션 탐지와 분석을 위해 개발된 이 도구는 안드로이드와 관련한 여러 가지 형태의 파일(Dex, APK, xml, asrc 등)을 대상으로 분석하여 그 결과를 사용자에게 보고하는 방식이다[15]. Androguard를 이용하여 악성 어플리케이션 분석을 진행한 [16]에서는 Androguard를 통해 APK 파일에서 Permission과 Control Flow Graph를 추출하여 악성코드 탐지 실험을 진행하였다.

'DroidBox'는 에뮬레이터 환경에서 악성 어플리케이션을 실행시킨 다음, 진행 시간대별로 디바이스 사용 권한, 어플리케이션 권한, logcat 정보 등을 모니터링하는 오픈소스 도구이다. [17]에서는 DroidBox를 이용하여 APK 파일을 분석하였다. DroidBox는 어플리케이션을 실행한 후 일정 시간동안 네트워크에서 데이터를 보내는 작업을 나타내는 'action_sendnet'과 같은 동적 행위를 포함하는 정보 로그를 생성한다.

'Wireshark'는 설치와 사용이 쉽고 기능들도 매우 다양하여 세계에서 가장 널리 쓰이는 네트워크 분석 프로그램 중 하나이다. 이 프로그램은 네트워크 상에서 캡쳐한 데이터에 대한 네트워크/상위레이어 프로토콜의 정보를 제공한다. 다른 네트워크 프로그램과 같이 Wireshark는 패킷 캡쳐를 위해 pcap 네트워크 라이브러리를 사용한다. [18]은 네트워크 트래픽 데이터를 특징으로 하여 기계학습을 적용한 이상탐지 기반 악성 어플리케이션 탐지 기법이다. 여기서는 Wireshark를 통해 악성코드의 TCP 패킷에서 모든 관련 정보를 추출한 뒤 True Positive Rate를 가장 높은 값으로 끌어낼 수 있는 가장 관련성이 높은 특징 11개를 선택하여 실험을 진행하였다.

3. 어플리케이션 행위 기반 특징 분석

동적 행위 분석 방법은 실제로 어플리케이션을 실행시켜서 발생하는 행위들을 직접 관찰하여 악성 여부를 판단하는 분석 방법이다. 기존의 안드로이드 악성코드 동적 분석에 관한 연구로는 안드로이드 플랫폼 수정을 통한 분석과 후킹 기법을 이용해 사용되는 API를 분석하는 방법이 있다. 플랫폼 수정 방법은 안드로이드 어플리케이션이 실제로 동작하는 DVM(Dalvik Virtual Machine)의 코드를 수정하여 사용되는 API를 추적하거나 로그를 남겨 악성 행위를 탐지한다[19]. 후킹 기법을 이용한 API 분석 방법은 안드로이드 플랫폼 구조 중 리눅스 커널 계층에서 사용되는 API를 후킹하여 동적 분석을 수행한다. 후킹 기법을 적용하기 위해 LKM(Loadable Kernel Module) 또는 공유 라이브러리 파일을 삽입하여 안드로이드 플랫폼에 발생되는 API를 모니터링하고, 그 결과로 악성코드를 탐지한다[20]. 대부분의 안드로이드 악성코드는 어플리케이션 개발을 위해 제공되는 SDK를 이용하여 제작되기 때문에, 어플리케이션에 사용되는 API를 추출하여 분석하면 악성 여부를 식별할 수 있다. [19]에서는 추출된 API를 기반으로 행위기반 분석을 크게 정적 분석과 동적 분석으로 나누어 더 많은 정보를 얻을 수 있도록 실험하였다. 정적 분석의 경우 해당 dex 파일을 디스어셈블하여 해당 어플리케이션에서 사용된 API에서 정보유출 API가 존재하는지 여부를 판단한다. 동적 분석에서는 추출된 dex 파일을 Dalvik 가상 머신에서 구동시켜 실제 메소드를 출력시키는 함수를 찾아 해당 부분을 수정하고, 새로운 안드로이드 이미지를 생성하고 이를 실행하여 생성되는 로그를 출력시키도록 했다. 이에 따라 탐지된 API가 실제로 어떤 순서로 호출되었는지에 대한 정보를 알 수 있다. 이렇게 분석을 통해 얻은 정보를 자동화된 악성코드 분류와 판별을 위해 기계학습 기법을 적용하였다.

(표 1) 행위 기반 분석에 사용되는 특징
(Table 1) Features used in behavior-based analysis

특징	설명
API Calls	API는 어플리케이션의 수행 작업을 파악할 수 있으며, 간접적으로 행위의 의도를 추론 가능
Runtime Log	입출 네트워크 데이터, 파일 읽기 및 쓰기 작업, DexClassLoader를 통한 서비스 시작, 로드된 클래스, 네트워크를 통한 정보 유출, 파일 및 SMS 정보 포함
System Resource	네트워크 액세스, 연락처, SMS 송수신 기능 같은 정보 포함, 어플리케이션이 리소스를 사용할 경우 개발자는 컴파일할 때 해당 리소스와 관련된 권한 요청
Network	악성코드가 발생시킨 패킷의 시작과 종료 시간, 송·수신 패킷의 플로우, IP 주소, Port 번호 등의 정보를 통해 악성코드 탐지

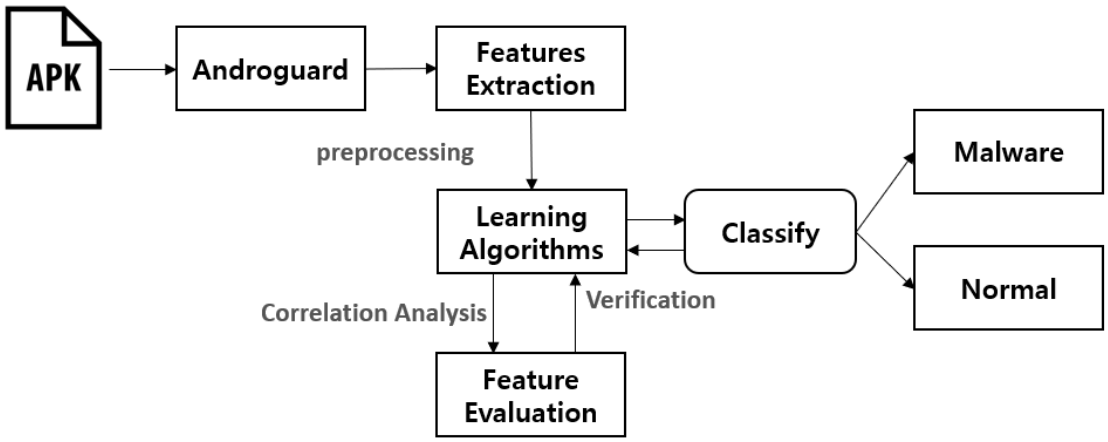
모바일 악성코드의 행위 프로파일링 기반으로 악성코드 탐지 및 분류 방법을 제시한[21]에서는 DroidBox의 안드로이드 SDK 커널을 후킹하여, 어플리케이션이 실행될 때 발생되는 시스템 호출 함수와 그 매개변수, 그리고 DroidBox에서 제공하는 로그(Permission 정보, 네트워크 통신 정보 등)를 통합한 정보를 활용하여 악성코드 분류를 진행하였다. 이 로그로부터 악성코드만의 행위 패턴을 프로파일링하여 악성코드 분석에 사용한다. 안드로이드 악성코드의 행위 기반 분석을 위해 추출되는 특징들은 대표적으로 API Calls, Runtime Log, System Resource, Network 등이 있으며, 자세한 설명은 표 1과 같다.

4. 머신러닝 기반의 악성 어플리케이션 분류

본 제안 방법에서는 안드로이드 어플리케이션을 동적 분석 도구를 이용하여, 행위 분석에 대한 주요 특징을 추출하고, 이를 머신러닝 분류기를 위한 속성을 재정의함으로써 머신러닝을 통해 악성 어플리케이션을 분류한다. 이러한 악성 어플리케이션을 분류하기 위해 (그림 1)과 같이 프로세스를 수행한다. 먼저 APK 파일에 대한 분석 정보 수집을 위하여 Androdguard[분석기를 통하여, 어플리케이션에 대한 분석을 추출을 진행한다. Androguard는 오픈 소스 분석도구로 안드로이드 어플리케이션을 분해하고, 디컴파일이 가능한 도구로 어플리케이션 분석에 유용하다. 추출된 분석 정보를 머신러닝 알고리즘에 적용하기 위하여 전처리 작업을 수행하며, 이에 대해 안드로이드 행위 기반에 대한 속성을 재정의하여 데이터세트를 구성한다. 구성된 데이터세트를 통하여 머신러닝을 적용하여 악성앱과 정상앱을 분류하는 작업을 수행한다. 주요 특징을 분별하기 위해 특징들 간의 상관관계를 수행하고, 전체 특징을 이용한 결과와 상관분석을 수행한 결과를 비교하여, 악성앱을 분류하기 주요 특징을 선별하고 머신러닝에 대한 성능을 향상 시킬 수 있는 근거를 평가한다.

4.1 데이터 수집

본 연구를 위한 데이터 수집은 일반 정상적인 apk는



(그림 1) 머신러닝을 이용한 어플리케이션 분류 프레임워크
(Figure 1) Application Classification Framework Using Machine Learning

Google Play Maker과 같은 미러링 사이트에서 수집이 이루어졌으며, 악성 apk 파일은 koodous[22], Virusshare와 같은 Malware 분석 사이트에서 수집이 이루어졌다. 수집된 apk은 SHA256 값을 통하여 Koodous의 분석 사이트에서 지원하는 Androguard 분석 정보를 수집하여 특징 추출이 이루어졌다. Koodous의 분석 사이트는 apk 파일에 대한 안드로이드 분석 도구를 통해 apk 파일에 대한 동적 분석을 수행을 지원해주며, 이 과정에서는 안드로이드 apk 파일을 분석할 수 있는 도구 androguard를 통해 동적 분석 프레임워크에 대한 기록을 추출할 수 있다.

4.2 특징 추출 및 전처리

데이터 수집 이후 Androguard 분석 도구를 통해 출력된 특징 값에 대해 전처리 과정을 수행한다. 일반적으로 apk 파일 분석 도구를 이용하여, 결과를 출력하였을 경우 각 특징에 대한 내용이 다양한 문자열과 숫자로 나타난다. 이는 apk 파일에 대한 주요 요소(API Call, Permission, sdk version 등)를 담고 있는 정보로 apk 파일에 대한 주요 특징으로 사용하여 분류가 가능하다. 이러한 주요 특징을 머신러닝 알고리즘에 적용하기에는 부적절하기 때문에 특징에 대한 속성들의 전처리가 필요하다. 출력된 값은 행위 분석에 대한 특징으로 문자열들을 정수형의 레벨로 변환한다. 행위에 대한 특징은 표 2와 같이 속성을 변형하였다.

(표 2) 분석 결과 값의 속성 정의
(Table 2) Define attribute of analysis result value

특징	속성 변환
API Call	연속형
Permission	범주형
Network	연속형
SMS	연속형
system info	범주형
서명키	범주형

4.3 머신러닝 적용

특징 추출과 전처리 작업이 진행되어지면, 머신러닝 모델에 적용하기 위한 데이터세트가 생성이 이루어질 수 있다. 머신러닝의 적용을 통해 변환된 속성에 따라 각 모델에 따른 성능 분석을 수행한다. 본 연구에서는 Navive Bayes, SVM, Random Forest, MultilayerP erceptron, BayesNet의 분류 알고리즘을 이용해 분류 정확도를 수행하였다.

4.4 특징 상관관계 분석

머신러닝을 위한 주요 특징을 분별하고 분류 향상을 위해, 특징들간의 상관분석을 수행한다. 이는 주성분분석(PCA, Principal Component Analysis) 방법을 적용한다.

전체 특징과 주성분석으로의 주요 특징간의 비교를 통

해 머신러닝 기술의 분류 평가를 검증하고, 각각 머신러닝 분류 모델에 따른 머신러닝 분류 평가를 수행한다. 이는 전체 특징의 불필요한 특징을 최소화하기 위한 방안과 성능 향상을 위한 단계이다. 또한 연구 목표로써, 머신러닝의 알고리즘의 분류 정확도를 향상시키고, 안드로이드 악성 어플리케이션 탐지에 대한 행위 분석 특징이 악성 어플리케이션의 분류에 적절한지 검증한다.

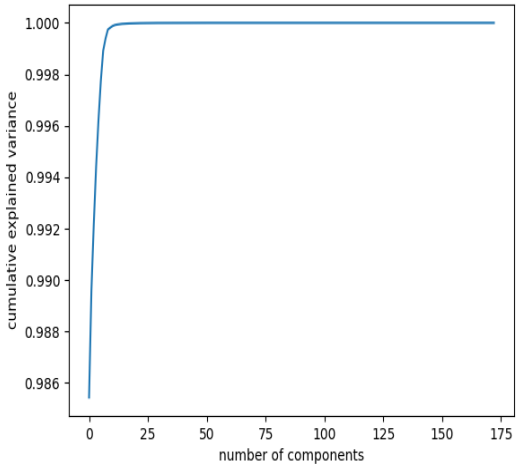
5. 실험 결과

본 연구의 안드로이드 악성 어플리케이션을 탐지하기 위해 안드로이드 어플리케이션의 행위 분석 분석 결과를 통해 악성어플리케이션을 분류한다. 이를 위해 수집된 데이터 세트는 총 400개의 어플리케이션을 수집하였으며, 정상 315개 악성 85개의 데이터로 구성되었다. 이후 총 400개의 샘플에서 추출되어지는 특징은 총 176개의 특징을 추출하였으며, 각각 표 2에 나타난 특징과 관련된 것으로 나열되어있다. 총 400개의 데이터샘플에서 75%는 트레이닝 데이터세트로 구성하였고, 나머지 25%는 테스트 데이터로 활용하였다. 이에 따른 전체 특징의 머신러닝 분류 결과는 표 3과 같다.

(표 3) 전체 특징을 이용한 머신러닝 분류 결과
(Table 3) Machine learning classification result using full feature

classifier	TP	FP	Accuracy	ROC
NB	0.935	0.069	93.5	0.961
SVM	0.845	0.574	84.5	0.635
RF	0.988	0.029	98.75	0.999
MP	0.98	0.04	98	0.998
BN	0.98	0.014	98	0.998

전체적으로 SVM을 제외한 정확도는 90%이상을 나타내는 것을 확인 하였으며, SVM 분류기 경우에는 84%의 성능을 나타내었다. 이에 대한 문제는 SVM 분류기가 정상적인 분류를 84%정도를 수행해주나, SVM 특성에서 범주형 데이터의 경우 다른 표본 특성들과의 Overfitting이 발생하여 나타나는 현상으로 바라볼 수 있다. SVM은 범주형 데이터가 적절하게 이루어지지 않았을 경우에 매우 취약한 현상이 발생하기도 한다. 이는 학습 데이터의 부족으로 평가되어짐으로 판단되어진다. 다음 실험으로는 전체 특징에서의 특징간의 상관분석을 통한 주성분석을 통하여 주요 특징만으로 학습 분류를 수행하였다. 주성분분석에 대한 차원 축소는 (그림 2)의 기울기에 따른 component 10개로의 가장 관련성 있는 76개의 특징으로의 PC 표본으로 수행하였다. 주성분분석은 도메인에 따라 무의미한 특징들이 있기 때문에 이를 제거할 필요가 있고, 특징간의 연관성을 분석하여 데이터의 특성을 반영한 특징들을 선정하기 위한 고차원의 데이터를 저차원의 데이터로 환원시키는 특징 선택 방법이다.



(그림 2) 주성분 분석을 통한 특징 component 설정
(Figure 2) Feature set using PCA

PCA를 적용한 머신러닝의 실험결과는 표 4에 나타내었다. 전체적으로 모든 분류기에서 정확도가 향상되었음이 확인이 되며, 특히 SVM 분류기의 성능이 대폭 향상되었다. 이는 특징간의 상관계수의 보정으로 인해 앞서 SVM의 특성에서 나타난 Overffiting 문제가 완화되어짐으로 나타난 것으로 확인되어진다.

(표 4) PCA 적용한 머신러닝 분류 결과
(Table 4) Machine Learning Classification Result Using PCA

classifier	TP	FP	Accuracy	ROC
NB	0.97	0.025	0.972	0.97
SVM	0.98	0.23	0.981	0.98
RF	0.98	0.04	0.98	0.98
MP	0.968	0.026	0.97	0.968
BN	0.975	0.05	0.975	0.975

6. 결 론

본 연구는 안드로이드 악성 어플리케이션 분류를 위하여 악성 어플리케이션에 대한 행위 분석에서 특징을 추출하고, 머신러닝 분류기를 통해 악성 어플리케이션을 탐지하는 연구를 진행하였다. 우리는 본 실험 결과를 통해서 일반 행위 분석을 통해 얻은 결과에서 전체적인 특징을 사용하는 것보다는 데이터의 특성에 따라 특징을 선정하는 것이 머신러닝 분류에 향상되어짐을 확인하였다. 이 결과는 SVM 분류기에서 10% 이상의 분류 결과가 향상된 검증을 통해서 알 수 있다. 분류 검증에서 악성 어플리케이션을 정확하게 분류하기 위해서는 좋은 데이터 세트를 생성하고, 의미있는 특징을 선정하는 것이 중요하다. 이를 위해서 행위 분석 결과 값에 대한 특징에 대한 속성 정의가 제대로 이루어져야 하고, 분석 결과에서 나타나는 전체 특징을 적용하는 것보다는 특징에 대한 연관분석으로 데이터의 특성을 반영하는 방법이 전체적인 머신러닝 분류 알고리즘 성능에 정확한 결과를 얻을 수 있다. 본 논문에서는 어플리케이션에 대한 분석 정보에서 다양한 텍스트 출력 값을 일반적인 속성으로 변환하여 다루었지만, 향후연구에는 데이터에 대한 특성이 고유하게 반영될 수 있도록 연구가 필요하다. 이는 데이터 분석의 식별의 이해와 주요 특징선택에 대한 상관분석 연구로 진행되어져야 할 것이다. 우리는 이 향후 연구에서는 텍스트 마이닝 기법으로 텍스트 정보에 대한 의미를 추출하고, 분석하여 자동으로 머신러닝에 적합한 유형으로 변환할 수 있는 연구로 나아갈 것이다.

참고문헌(Reference)

[1] http://www.itworld.co.kr/news/104914

[2] https://www.gdata-software.com/news/2017/02/threat-situation-for-mobile-devices-worsens

[3] AhnLab 보안 이슈 : http://www.ahnlab.com/kr/site/securityinfo/secunews/secuNewsView.do?seq=19269.

[4] Yajin Zhou and Xuian Jiang, “Dissecting Android Malware: Characterization and Evolution”, In security and Privacy(SP), 2012 IEEE Symposium on, IEEE, pp. 95-109, May, 2012.

[5] http://blog.trendmicro.com/trendlabssecurity-intelligence/a-look-into-repac kaged-apps-and-its-role-in-the-mobile - threat-landscape/

[6] Google Mobile Blog, “Android and Security”, 2012.

[7] 김기현, 함효식, 최미정, “SVM을 이용한 안드로이드 기반의 악성코드 탐지”, 제 40회 정보처리학회 추계종합학술대회, 2013.

[8] Shabtai, Asaf, et al. ″“Andromaly”: a behavioral malware detection framework for android devices.″ Journal of Intelligent Information Systems 38.1, pp. 161-190, 2012.

[9] Burguera, Iker, Urko Zurutuza, and 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. ACM, pp. 15-26, 2011.

[10] Liu, Lei, et al. ″VirusMeter: Preventing Your Cellphone from Spies.″, RAID. Vol. 5758. pp. 244-264, 2009.

[11] Cheng, Jerry, et al. ″Smartsiren: virus detection and alert for smartphones.″, Proceedings of the 5th international conference on Mobile systems, applications and services. ACM, pp. 258-271, 2007.

[12] Castillo, C. ″Spitmo vs Zitmo: Banking Trojans Target Android.″, 2011.

[13] Z. Yajin and J. Xuxian, “Dissecting android malware: Characterization and evolution,”, Proceedings of 33rd IEEE Symp Security Privacy, Oakland, CA, USA, pp. 95–109. 2012.

[14] C. A. Castillo, “Android malware past, present, future”, Mobile Working Security Group McAfee, Santa Clara, CA, USA, Tech. Rep. 2012.

[15] Kim Jun-Hyoung, Im Eul-Gyu. ″Androguard: Similarity Analysis for Android Application Binaries.″, Korea Computer Congress, pp. 101-103, 2014

[16] SAHS, Justin; KHAN, Latifur. ″A machine learning approach to android malware detection″, In: european Intelligence and security informatics conference (eisic), IEEE, pp. 141-147, 2012.

[17] Yuan, Zhenlong, et al. ″Droid-Sec: deep learning in android malware detection″, ACM SIGCOMM Computer Communication Review. Vol. 44. No. 4. ACM, pp. 371-372, 2014.

[18] Narudin, Fairuz Amalina, et al. ″Evaluation of machine learning classifiers for mobile malware detection″, Soft Computing 20.1, pp. 343-357, 2016.

[19] Seungwook Min, Hyungjin Cho, Jinseop Shin, Jaecheol Ryou, ″Android Malware Analysis and Detection Method

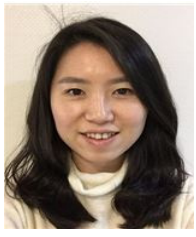
Using Machine Learning", Journal of KIISE : Computing Practices and Letters, 19(2), pp. 95-99, 2013.
[20] Yun-sik Jeong, Seong-wook Kang, Seong-je Cho and In-sik Song, "A Kernel-based Monitoring Approach for Analyzing Malicious behavior on Android," Korea Computer Congress, pp. 127-129, Jun. 2013
[21] Jae-sung Yun, Jae-wook Jang, Huy Kang Kim, "Andro-profiler: Anti-malware system based on behavior profiling of mobile malware", Journal of the Korea Institute of Information Security & Cryptology, 24(1), pp. 145-154, 2014.
[22] https://koodous.com/

◐ 저 자 소 개 ◑

김 동 욱 (Dong-Wook Kim)
2015년 가천대학교 컴퓨터공학과(공학사)
2017년 가천대학교 일반대학원 컴퓨터공학과(공학석사)
2017년~현재 가천대학교 컴퓨터공학과 박사과정
관심분야 : Data Mining, 인공지능, Data fusion, Anomaly Detection
E-mail : kog7306@naver.com

나 경 기 (Kyung-Gi Na)
2016년 가천대학교 컴퓨터공학과(공학사)
2016년~현재 가천대학교 일반대학원 컴퓨터공학과(공학석사)
관심분야 : 데이터 마이닝, 기계학습, 인터넷 보안
E-mail : rudrl15@hanmail.net

한 명 묵(Myung-Mook Han)
1980년 연세대학교 공과대학 (공학사)
1987년 뉴욕공과대학교 컴퓨터공학과(공학석사)
1997년 오사카시립대학교 정보공학과(공학박사)
1998년~현재 가천대학교 컴퓨터공학과 교수
관심분야 : Cyber Security, Data Mining
E-mail : mmhan@gachon.ac.kr



김 미 주(Mijoo Kim)
2006년 순천향대학교 정보보호학과(공학사))
2008년 순천향대학교 대학원 정보보호학과(공학석사)
2008년~현재 순천향대학교 대학원 정보보호학과 박사과정
2008년~현재 한국인터넷진흥원 선임연구원
관심분야 : 사이버 보안, 모바일 보안, IoT 보안, 정보보호 표준화
E-mail : mijoo.kim@kisa.or.kr

◐ 저 자 소 개 ◑

고 웅(Woong Go)
2008년 순천향대학교 정보보호학과(공학사)
2010년 순천향대학교 대학원 정보보호학과(공학석사)
2013년 순천향대학교 대학원 정보보호학과(공학박사)
2014년~현재 한국인터넷진흥원 선임연구원
관심분야 : 사이버보안, 인공지능, 모바일 악성앱
E-mail : wgo@kisa.or.kr

박 준 형(Jun-Hyung Park)
1999년 아주대학교 정보및컴퓨터공학대학(공학사)
2002년 전남대학교 대학원 멀티미디어학과(이학석사)
2004년 전남대학교 대학원 정보보호학과(공학박사)
2017년~현재 한국인터넷진흥원 보안기술R&D1팀 팀장
관심분야 : 사이버 보안, 사물인터넷
E-mail : junpark@kisa.or.kr