Journal of KIIT. Vol. 16, No. 3, pp. 101-113, Mar. 31, 2018. pISSN 1598-8619, eISSN 2093-7571 101
http://dx.doi.org/10.14801/jkiit.2018.16.3.101

머신 러닝(Machine Learning)기법을 활용한 실시간 악성파일 탐지 기법

전덕조*, 박동규**

Real-time Malware Detection Method Using Machine Learning

Deok-Jo Jeon*, Dong-Gue Park**

이 논문은 2017학년도 순천향대학교 교수 연구년제에 의하여 연구하였음.

요 약

최근 위협 행위자들은 기존 보안 솔루션이 시그니처를 기반으로 알려진 위협을 식별한다는 사실을 인지하고 악성코드의 모양을 변형함으로써 변종을 생성한다. 상용 안티바이러스 및 안티 맬웨어 소프트웨어는 일반적으로 일련의 시그니처에 의존하므로 신종 악성코드를 탐지하기에는 적절하지 못한 단점을 가지고 있다. 이런 문제를 해결하기 위하여 본 논문에서는 악성코드를 탐지하기 위한 비시그니처 기반 머신 러닝 방법을 제안한다. 제안한 방법은 실행파일에서 추출된 각 특징의 정보이득(IG)을 계산한 결과에 결정 트리(Decision Tree) 기법을 사용하여 높은 탐지율과 낮은 오탐율을 확보한다. 또한 최근 심각한 사이버 위협으로 대두되고 있는 랜섬웨어에 본 연구에서 제안한 방식을 적용하고 그 결과를 제시함으로써 본 논문의 효율성을 증명하고자 한다.

Abstract

Recently, threat actors know that traditional security solutions are based on signatures that identify known threats, and so they create variants by altering the appearance of malware. Most commercial anti-virus and anti-malware products generally rely on signatures to detect malware and thus cannot detect unknown or zero-day malwares effectively. In order to solve this problem, we propose a non‐signature-based machine learning technique for detecting malware. Our method uses a decision tree technique on the Information Gain (IG) of each feature extracted from executable files and it shows high accuracy at a low false positive rate. Since the recent rise of ransomware as a cybersecurity threat has become serious and widespread, we verify the effectiveness of our method by applying it to ransomware domain.

Keywords
malware detection, data mining, machine learning, decision tree, ransomware detection

* ㈜ 시큐비스타 대표이사
- ORCID: http://orcid.org/0000-0002-0343-3384
** 순천향대학교 정보통신공학과 교수(교신저자)
- ORCID: http://orcid.org/0000-0002-5864-8825

· Received: Oct. 16, 2017, Revised: Jan. 11, 2018, Accepted: Jan. 14, 2018
· Corresponding Author: Dong-Gue Park
Dept. of Information and Communication Engineering Soonchunhyang Univ.
Tel.: +82-41-530-1347, Email: dgpark@sch.ac.kr:

Ⅰ. 서 론

상용 안티바이러스 및 안티 맬웨어 소프트웨어는 일반적으로 이전에 수집한 악성코드로부터 추출한 일련의 시그니처 (바이트 문자열)에 의존하므로 신종 악성코드를 탐지하기에는 적절하지 못한 단점을 가지고 있다. 또한 최근 악성코드 작성자는 패킹 기법을 사용하여 원래의 바이트 순서를 무작위로 보이는 데이터로 변환함으로써 시그니처 기반 악성코드 탐지 기법을 무력화시키고 있다[1]. 따라서 많은 연구에서 PE(Portable Executable) 형식[2]에서 추출한 특징에 데이터 마이닝 기법을 적용하는 비 시그니처 방식의 악성 코드 탐지 기법을 제안하고 있다[3]-[10]. PE 형식은 윈도우 운영 체제에서 사용되는 실행 파일, DLL, 객체 코드, 폰트 파일 등을 의미한다[2]. PE 형식에서 추출한 특징에 데이터 마이닝 혹은 머신 러닝 기법을 적용한 많은 연구에서 낮은 오탐율과 높은 정확성을 제시하였다. PE 형식에서 추출한 특징을 사용하는 방식은 파일 분석에 소요되는 시간이 상용 안티바이러스 제품과 마찬가지로 실시간 악성코드 분석에 적용할 수 있다는 장점이 있다[3]-[10]. 그러나 대다수의 연구에서 PE 형식에서 추출한 특징을 그대로 사용하지 않고 추출한 특징을 축약 및 변형하는 기법을 사용하고 있기 때문에 네트워크 보안 분석가에게 분류 이유에 대한 정보를 제공하기 어렵다. 이러한 문제점을 해결하기 위하여 Anselm[3]은 사람이 이해하기 용이한 결정 트리(DT, Decision Tree)법을 적용하는 방법을 제안하였다. 그러나 Anselm[3]이 제시한 방안도 결정 트리의 규모가 방대하여 네트워크 보안 분석가가 전체 분류 과정을 추적하기가 어려운 단점을 가지고 있다. 또한 Bai[11]도 결정 트리 방법을 사용하여 실행 파일의 분류과정을 보안 분석가가 이해할 수 있도록 하였으나, 사용한 특징의 수가 제한적이어서 실행 파일의 분류과정에 대한 정보 또한 제한적인 단점이 있었다. 따라서 본 논문에서는 이런 문제를 해결하기 위하여 새로운 PE 특징을 추출하고, 정보이득(IG, Information Gain) 계산 기법을 이용하여 이들 중에서 소수의 특징을 선별함으로써 Anselm[3]이 제안한 방법보다 높은 탐지율과 낮은 오탐율을 갖는 새로운 방안을 제시하고자 한다. 또 최근 심각한 사이버 위협으로 대두되고 있는 랜섬웨어를 대상으로 본 논문의 실험 결과를 적용해 봄으로써 본 논문에서 제시한 방안의 랜섬웨어 탐지에 대한 효율성을 증명하고자 한다.

Ⅱ. 관련 연구

PE는 윈도우즈 실행 파일을 의미하며 윈도우즈의 실행 가능한 파일 형식이다. PE 구조의 명세는 원래 유닉스 COFF(공통 객체 파일 형식)에서 유래되었으며, PE 파일의 형식 정보는 그림 1과 같다.

PE 파일은 그림 1과 같이 PE 파일 헤더와 섹션 테이블 (섹션 헤더)과 그 다음 섹션 데이터로 구성된다. PE 파일에는 다수의 중복 필드와 공백이 있으므로 악성코드 전파 및 은닉이 가능하다. 또한 악성코드와 정상파일의 PE 형식정보에는 다음 예와 같이 많은 차이점이 존재할 수 있다. (1) 마지막 섹션에서 시작하는 코드 실행, (2) 의심스러운 섹션 특성, (3) 의심스러운 코드 리디렉션, (4) 의심스러운 코드 섹션 이름, (5) 어떤 섹션도 포인트 하지 않은 엔트리 포인트, (6) 패치 되지 않은 IAT, (7) 다수의 PE 헤더, (8) 헤더 내 코드 크기의 오류. 그러므로 이러한 정보들을 효과적으로 분석하면 악성코드를 탐지할 수 있다.

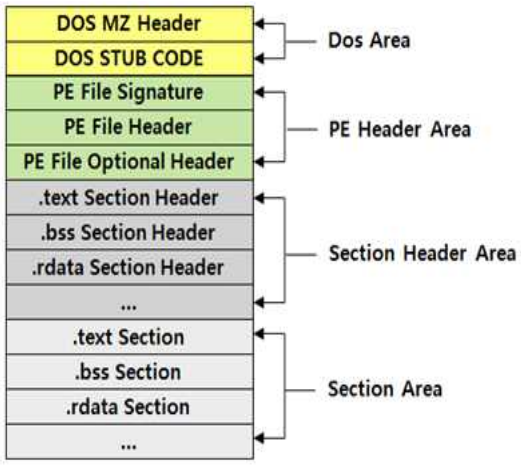

그림 1. PE 구조
Fig. 1. PE structure

비 시그니처 방식의 악성코드 탐지 기법은 분석 유형에 따라 정적 분석 및 동적 분석 기법으로 분류되며, 본 논문에서는 PE 파일의 정적 분석을 기반으로 하는 비 시그니처 기반 악성코드 탐지 방식을 제안하고자 한다. 이와 관련된 이전의 주요연구는 다음과 같다.

Shafiq[4][5]는 악성 PE 파일을 탐지하는 두 가지의 연관된 기법을 제안하였다. Shafiq[4][5]가 제안한 기법은 실시간 탐지 성능 및 높은 정확성을 제시하였으나, PE 특징의 차원을 줄이기 위하여 특징 사전 처리기법을 이용하였고 다중 퍼셉트론(Multi-layer Perceptron)을 분류 알고리즘으로 선택하여 악성코드 분류과정에 대한 정보를 제공하지 못하는 문제점이 존재한다.

Schultz[6]은 악성코드와 정상 프로그램의 정적 특징을 이용하는 데이터 마이닝 기반의 악성코드 탐지 방법을 제안하였다. 그들은 실행 파일의 다양한 정적 특징을 가지고 다양한 데이터 마이닝 분류기(RIPPER, Naive Bayes, 다중 Naive Bayes)를 사용하여 3265 개의 악성코드 및 1001 개의 정상 프로그램으로 학습시켰다. 그들은 다중 Naive Bayes를 사용할 경우, 97.6 % 정확도를 확보할 수 있었지만 특정한 형태의 악성코드에만 적용이 가능하다는 단점을 가지고 있다.

Kolter[7]는 겹치지 않는 시퀀스 대신 바이트 n-그램을 사용하여 Schulz[6]의 기법을 향상시켰다. 4-그램을 특징으로 사용하였으며, 정보이득 계산을 통하여 상위 500개의 4-그램을 선택하였다. 그들은 Naive Bayes, 인스턴스 기반 학습기, 의사 결정 트리, TFIDF 및 서포트 벡터 머신(SVM)을 사용하였으나, 너무나 많은 특징 벡터가 생성되므로 실시간 적용이 불가능하다는 문제점이 존재한다.

Robert[8][9], Asaf[10] 등은 PE 파일을 디스어셈블한 결과에서 OpCode 시퀀스를 n-그램으로 사용하는 방법을 제안하였다. 그러나 Robert[8][9]와 Asaf[10] 등의 방법은 성능 문제뿐만이 아니라 일부 역어셈블 할 수 없는 PE파일들에 대한 분석이 불가능하다는 단점이 존재한다. 또한 두 가지 접근 방법 모두 악성코드의 전체가 아니라 악성코드의 일부분만을 이용하여 해당 파일의 악성 여부를 판별해야 하는 문제점이 존재한다.

Belaoued[12]는 PE 옵션 헤더 필드에 저장된 정보 분석에 기반한 실시간 PE 악성코드 탐지 시스템을 발표하였다. 카이 제곱 및 파이 계수를 사용하여 특징을 선택하고 선택된 특징에 대해 로테이션 포레스트(Rotation Forest) 분류기를 실험하였다. 그러나 이들이 사용한 기법 또한 사람이 이해할 수 있는 악성코드 분류과정에 대한 정보를 제공하지 못하는 문제점이 존재한다.

Yan[13]은 이미지 자원 NameID 필드 및 특성을 제외한 모든 PE 헤더 필드를 숫자 특징으로 사용하였으며, 특성, DLL 및 시스템 호출 정보의 각 비트는 부울 특징으로 표현되었다. 이들은 Relief, 카이 제곱(Chi-Square), F-통계 및 L1-정규화된 SVC의 4가지 특징 선택 방법과 결정 트리 및 부울 피처 세트로 실험하였으나, 이들이 사용한 기법 또한 사람이 이해할 수 있는 악성코드 분류과정에 대한 정보를 제공하지 못하는 문제점이 존재한다.

박진일[14]는 정상 실행파일에 대하여 API 호출 리스트를 화이트리스트로 정의한 후, 악성코드 표본에 대하여 정적 분석과 동적 분석을 통해 문자열과 API 호출 목록과 빈도를 수집한 후, 머신 러닝 알고리즘을 적용하는 악성코드 분석 방안을 제안하였다. 그러나 이들의 방법은 동적 분석 기법을 이용하여야 하므로 악성코드 사전 실행을 위한 시간이 소요되므로 실시간 탐지에는 적용이 불가능한 문제점이 존재한다.

대다수의 이전 연구들은 PE 특징이 분류과정에서 어떻게 이용되었는지에 대한 정보를 제공할 수 없기 때문에 분류과정에 대한 정보를 보안 분석가에게 제공할 수 없다는 한계가 있다. 이러한 문제점을 해결하기 위하여 Anselm[3]은 결정 트리 알고리즘을 사용하였다. 또한 마찬가지 이유로 PE 특징을 추출하여 전처리 기법을 사용 하지 않고 원시 특성을 그대로 이용하였다. 하지만 189개의 특성을 그대로 이용할 경우, 결정 트리의 규모가 너무 방대하여 보안 분석가가 분류 특성을 추적하기가 쉽지 않다는 단점이 존재하기 때문에 정보이득을 계산을 통하여 상위 50% PE 특징만을 사용하는 방안을 제안하였다. 그러나 정보이득의 상위 50%만을 사용하는 경우에도 결정 트리의 규모가 여전히 방대하여 분석가가 분류 과정을 이해하기 어렵다는 문제점이

존재한다.

Bai[11]은 PE 헤더 필드에서 추출한 197 개의 특성으로 구성된 초기 특징 세트를 구성하였으며, 필터와 래퍼 기법을 이용하여 특징을 축약하여 다양한 실험을 하였다. 저자들은 필터 및 래퍼 선택 방법을 사용하여 특징을 20개 이하로 줄인 특징 세트를 이용하였고, 다양한 실험을 하였다. 그 중에서 결정 트리 알고리즘 적용 시 탐지 정확성은 차치하고 원시 특징에 의한 실행 파일의 분류과정을 보안 분석가가 이해하는데 도움을 줄 수 있으나 사용한 특징의 수가 매우 제한적이므로 실행 파일의 분류 과정에 대한 정보 또한 제한적일 수 밖에 없다는 단점이 있다.

따라서 본 연구에서는 이러한 문제점을 해결하기 위하여 Anselm[3]이 사용한 특징보다 더 적은 원시 특징을 사용하여 결정 트리의 규모를 줄여서 보안 분석가에게 정보를 제공할 수 있으며, 높은 탐지율과 낮은 오탐율을 확보할 수 있는 방안을 제시하고자 한다.

III. 연구 방법

본 논문에서는 PE 파일의 원시 특징을 그대로 이용하고 결정 트리 알고리즘을 사용함으로써 보안 분석가에게 분류과정에 대한 통찰을 제공하는 방안을 제시하고자 한다. 또한 높은 탐지율 및 낮은 오탐율을 제공하는 동시에 실시간 탐지가 가능한 비시그니처 기반 악성코드 탐지 방안을 제시하고자 한다. 또한 본 논문에서 제안한 방식의 신뢰성 및 효율성을 검증하기 위하여 최근 유행하고 있는 랜섬웨어에 대해서 제안한 방식을 적용 및 검증하고자 한다.

본 논문의 연구를 위하여 그림 2와 같은 실험 환경과 그림 3과 같이 두 가지 실험 방안을 설계하여 알려지지 않은 악성코드 및 신종 악성코드 탐지 능력을 평가한다.

그림 2의 실험 환경에서 먼저 악성 코드 및 정상 파일을 수집하여 검증한 후, 검증된 데이터 샘플로부터 PE 파일 구조 속성에서 추출 가능한 특징을 추출한다. 추출된 각 특징에 대해 정보이득을 계산하여 정보 이득이 높은 특징을 특징으로 하는 서브 세트를 정의한다. 서브 세트를 이용하여 결정 트리를 학습 시키고 나온 결과를 10중 교차 검증으로 검증하여 최종 결과를 도출한다.

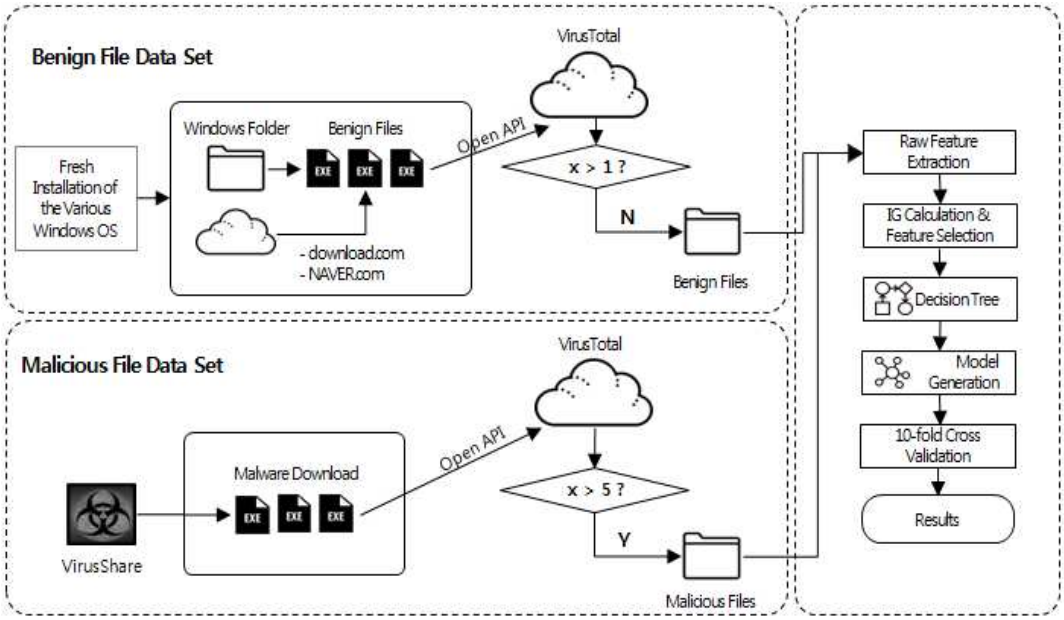

그림 2. 실험 환경
Fig. 2. Experiment environment

그림 3의 실험 I에서는 Anselm[3]이 사용한 정보이득 계산기법에 기반한 특징 선택 및 결정 트리 기반 분류기법을 실험한다. Anselm[3]이 정보이득의 상위 50%를 선택한 반면, 본 논문에서는 결정 트리의 규모를 줄이기 위하여 정보이득 상위 10%, 20%, 30%를 선택하여 실험한다. 실험 II에서는 실험 I의 결과 중에서 가장 좋은 결과를 랜섬웨어 데이터 집합에 적용하여 실험 한다. 즉, 실험 II는 실험 I의 랜섬웨어가 배제된 악성코드에 대한 실험 결과가 랜섬웨어의 경우에도 동일하게 적용할 수 있는 지를 검토하기 위한 실험이다.

실험 I에서 3가지 방식을 적용하여 실험 결과 중에서 최상의 결과를 선택하고자 한 이유는 PE 파일 데이터 특징 중에서 최적의 서브 세트를 선택하는 다양한 방법이 존재하기 때문이다. 본 논문에서는 정보이득 계산 기법을 선택하였는데, 정보이득 계산법을 이용한 이유는 Anselm[3]과 유사한 방식으로 특징을 선택하되, 특징의 수를 줄임으로써 결정 트리의 규모를 축소하기 위해서이다. 또 특징의 수를 줄여서 실험한 결과 중에서 어떤 방법으로 특징을 선택하는 것이 가장 좋은 탐지 결과를 도출하는 지를 비교한 후, 최선의 방법을 선택하기 위해서이다.

3.1 정상파일 수집 및 검증

본 연구에서는 그림 4와 같이 정상 PE파일을 수집하기 위하여 Shafiq[4], Anselm[3] 및 Bai[11]이 사용한 방법과 유사한 방법 즉, 마이크로소프트 윈도우즈 운영체제를 신규로 설치하여 수집하였다.

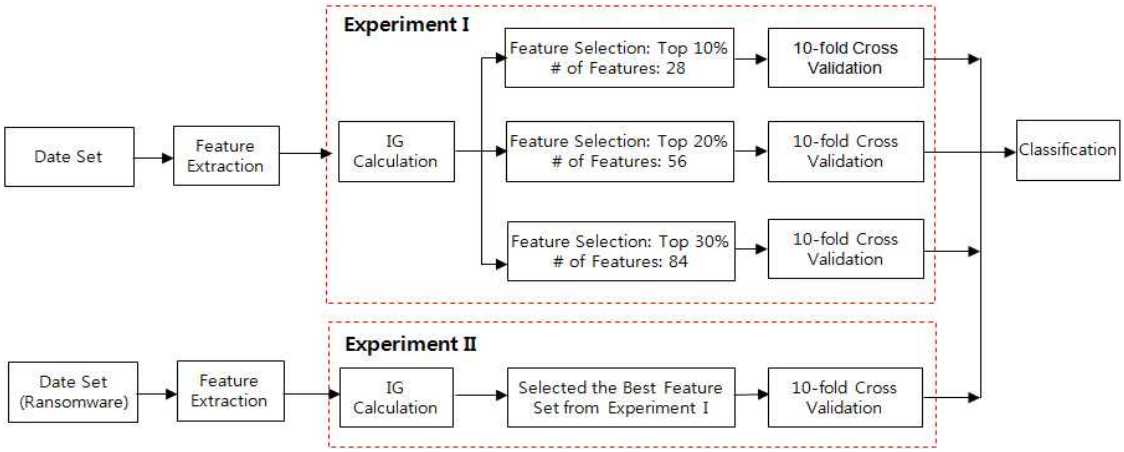

그림 3. 실험 아키텍처
Fig. 3. Experiment architecture

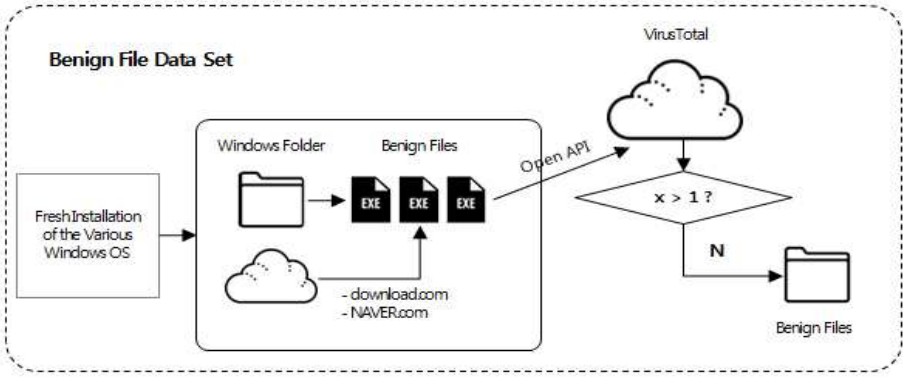

그림 4. 정상파일 수집 및 검증
Fig. 4. Benign file collection and verification

추가적으로 CNET이 운영하는 download.com 사이트로부터 수집하였으며, 이외에 네이버에서도 PE 파일을 추가 수집하였다.

그림 4에서와 같이 정상 파일에 대한 검증은 CNET의 download.com 및 네이버에서 다운로드 받은 파일의 경우에만 바이러스토탈(Virustotal)의 오픈 API를 사용하여 검증하였고, 검증 기준은 전체 약 69개의 안티바이러스 엔진 중에서 1개라도 악성으로 판정되는 경우에는 배제하였다. 윈도우즈의 다양한 버전을 신규로 설치하여 수집한 PE 파일은 정상 파일로 그대로 이용하였다.

바이러스토탈은 구글의 자회사로 편입된 회사이며, 약 69개 안티바이러스 회사들과 제휴하여 악성코드 정보를 공유하고 있으며, 특히 전 세계 일반 사용자들이 업로드한 악성코드, URL및 PCAP등을 약 69개 안티바이러스 회사들의 엔진으로 검사하고 해당 결과를 제공하는 클라우드 서비스이다. 바이러스토탈에서는 동일한 과정을 자동화된 서비스로 제공하기 위하여 API를 제공하는 데, 오픈 API와 유료 API로 구분된다. 바이러스토탈 사이트는 사용자가 수동 또는 오픈 API를 통해 업로드 한 악성파일에 대하여 여러 개의 백신 엔진으로 검사한 결과를 투명하게 제공한다.

3.2 악성 파일 수집 및 검증

Shafiq[4][5]및 Anselm[3]의 경우, 악성 PE 파일의 수집은 VX Heavens[15]와 Offensive Computing[16]의 2 곳에서 진행하였다. VX Heavens 의 경우, 레이블된 많은 악성코드가 존재하기는 하지만 2007년 이전의 악성코드가 대부분이므로 현실적으로 양질의 악성 PE 데이터를 수집하는 것이 불가능하다는 문제점이 있다. 양질의 악성코드 및 최신 악성코드를 수집하는 가장 좋은 방법은 바이러스토탈을 이용하는 것이지만, 현실적으로 유료 API를 구매하여야 한다는 문제점이 있다. VirusShare의 경우, 최신 악성코드를 수집하는 것은 가능하지만, 레이블이 되어 있지 않기 때문에 해당 악성코드를 그대로 이용할 수 없다는 문제점이 있다.

본 연구에서는 이러한 문제점들을 해결하기 위하여 그림 5와 같이 악성코드 샘플들을 VirusShare로부터 수집하고 이들을 바이러스토탈의 오픈API를 이용하여 레이블 하는 방법을 사용하였다.

바이러스토탈 오픈 API를 사용하는 경우, 속도, 일간 쿼리 건수 등의 제약이 존재하여 시간이 소요되지만, 전 세계 69개 안티바이러스 엔진에 의한 탐지 결과를 이용하여 정확하게 악성코드 샘플을 선택할 수 있다는 장점이 있다.

표 1에서 기존 연구와 본 논문에서 사용되는 악성 및 정상 PE 파일을 비교하였다. 그림 4와 같은 과정을 통해서 수집한 정상 파일 데이터 세트 및 그림 5와 같은 과정을 통하여 수집한 악성 파일 데이터 세트와 이전의 유사 연구에서 사용한 데이터 세트를 비교하여 나타내고 있으며, 이전의 유사 연구에 비해서 다소 많은 양의 데이터 세트를 사용하고 있음을 알 수 있다.

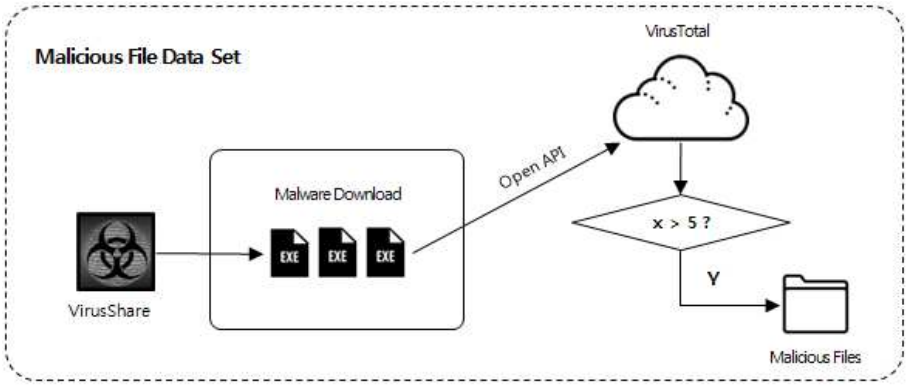

그림 5. 악성파일 수집 및 검증
Fig. 5. Malicious file collection and verification

표 1. 악성 및 정상 데이터 샘플의 비교
Table 1. Comparison of benign file and malware samples

Results	[4]	[3]	[14]	Our methods	
Benign Files	1,447	1,009	8,592	15623	
Malicious Files	15,925	1425	10,521	15,000	-
Ransomware	-	-	-	-	10,415
Total	17,402	2434	19,103	30,623	26,038

3.3 특징 추출

Shafiq[4], Anselm[3] 및 Bai[11]이 정상 및 악성 파일을 구분하기 위하여 PE 파일을 이용하여 추출한 PE 특징은 표 2에 요약되어 있으며 PE 파일 구성요소는 Microsoft사의 PE 파일 명세서에 상세히 설명되어 있다[2].

표 2. 선택된 PE 특징의 비교
Table 2. Comparison of Selected PE features

Feature Type	Selected Features (Shafiq[4], Anselm[3])		Selected Features (Bai[14])		Our methods	
DLLs referred	73	Which of a list of 73 core DLLs are referred to by the PE file - giving 73 Boolean features	30	Dumped each executable file's import table of the PE format and counted all the DLLs that are used; and then removed those DLLs appeared less than 100 times; and then computed information gain for each DLL and elected the best 30 different DLLs	35	Dumped each executable file's import table of the PE format and counted all the DLLs that are used; and then removed those DLLs appeared less than 100 times; and then computed information gain for each DLL and selected the best 35 different DLLs
Image File Header Fields	7	"COFF file header" The values of 7 fields from this header, which indicate things such as the target processor type, the number of sections and the number of symbols	53	IMAGE FILE HEADER (5) IMAGE OPTIONAL HEADER (16) IMAGE DATA DIRECTORY (32)	23	IMAGE FILE HEADER(7) IMAGE OPTIONAL HEADER (16)
Optional Header Fields	61	The content of 9 standard fields, 22 Windows-specific fields, and 30 fields relating to data directories	6	# of DLLs, # of APIs referred, # of sections, # of symbols in export table, # of items in reloc section, Dos header－e lfanew	76	Data Directory Entry(32) Optional Header(29) DLL related (15)
Section Header Fields	27	The content of the 9 header fields from the .text, .data and .rsrc sections of the file, for a total of 27 fields.	55	Considered text, data, rsrc, rdata, and reloc sections and selected 11 fields from each section	120	text, .data, .rsrc section header field(36) .rdata, .reloc, .bss, .edata, .idata, .pdata, .debug section (84)
Resource directory table and resources	21	Counts of various types of resources (such as icons and dialog boxes) used by the program, giving a total of 21 integer attributes	23	Extracted the number of 23 kinds of resources type	22	Resource Directory Entry (22)
Others			30	Dumped each executable file's import table of the PE format and counted all the APIs that are used; and then removed those APIs appeared less than 100 times; and then computed information gain for API and selected the best 30 APIs	4	NumberOfDLLs, NumberOfAPIs, ProductVersion, DOS_HEADER.HEADER.e_lfanew (4)
Total	189		197		280	

PE 파일에서 정상 파일은 일반적으로 광범위한 동적 링크 라이브러리(DLL)들을 사용하는 반면, 악성 파일은 훨씬 적은 수의 라이브러리를 사용한다. 그리고 PE 파일의 "옵션 헤더"는 실행파일에서는 필수적이지만, 일부 파일(즉, 객체 파일)에서는 요구되지 않는다. "옵션 헤더"는 다음과 같이 3가지 부분으로 구성된다.

"표준 필드" (일부 UNIX 변종에서도 정의됨), "윈도우즈-전용 필드" (윈도우즈 전용 특징을 지원함), "데이터 디렉터리" 필드 - Import Address Table 또는 IAT와 같은 이미지 파일에 포함된 테이블의 주소 및 크기. 그리고 각 섹션 헤더는 1 개의 텍스트 필드(이름) 및 섹션 속성을 기술하는 9 개의 숫자 필드 (예를 들면, 메모리에 로드 되었을 때의 전체 크기)로 구성되어 있다. 그런데 악성 파일은 흔히 이러한 헤더 필드 중의 일부가 전형적이지 않은 값을 가진다. 예를 들면, major version number 필드가 흔히 0으로 설정되어 있고, 아이콘 및 대화상자와 같은 리소스의 수가 일반적으로 정상파일보다 작다.

본 논문에서는 PE 파일에서 추출 가능한 모든 특징을 추출하고, 이 특징을 기반으로 정보이득을 계산하여 상위 10%, 20%, 30%를 선택하여 사용함으로써 결정 트리의 규모를 줄이는 동시에 실시간으로 보안 분석가에게 보다 상세한 정보를 제공할 수 있는 방안을 제시하고자 한다. 본 논문에서 사용한 특징과 기존 연구 Shafiq[4], Anselm[3] 및 Bai[11]에서 사용한 특징의 비교는 표 2와 같다. 본 논문에서 추출한 원시 특징의 수가 다소 증가한 이유는 PE 파일에서 추출 가능한 모든 특징을 추출하였기 때문이다.

표 2에서 알 수 있듯이 본 논문에서는 사용한 PE 특징은 Shafiq[4], Anselm[3] 및 Bai[11]이 사용한 PE 특징과 거의 유사하나, 전체적으로는 사용한 필드 수가 많다. 표 2에서 Shafiq[4]와 Anselm[3]와 본 논문에서 선택한 PE 특징 중에서 가장 상이한 특징은 참조된 DLL의 선택방법이다. 일반적으로 실행파일에서 참조된 DLL 목록은 해당 DLL의 기능을 효과적으로 제공할 수 있다. Schultz[6]는 DLL 이름 및 유사한 기능을 결합하여 특징으로 이용하였는데, DLL 관련 특징이 탐지 정확도를 높이는 데 도움이 된다는 사실을 확인할 수 있다. Shafiq[4]는 DLL 특징을 개별 바이너리 특징으로 사용하면 더 많은 정보를 얻을 수 있기 때문에 악의적인 PE 파일을 탐지하는 데 도움이 되며 따라서 73개의 핵심적인 DLL을 선정하여 특징으로 선택하여 사용하였다. 그러나 Shafiq[4]이 선택한 핵심 DLL은 최신 PE 파일의 특징을 충분히 반영하고 있지 못하고 있다. 예를 들면, 핵심 DLL중에서 ADVAP132같은 32bit API DLL, AWFAXP32과 같은 메일 API 팩스 전송관련 DLL, MFC30과 같은 공유 MFC DLL, NWNET32와 같은 NetWare 클라이언트 DLL 등대부분이 최신 PE 파일과는 무관한 DLL들로 구성되어 있다.

따라서 본 연구에서는 이러한 문제점을 보완하기 위하여 전체 데이터 집합에서 Import Table을 dump하여 최소 100회 이상 사용된 DLL의 정보이득을 계산하여 상위 35개만을 선택하는 방법을 사용하였는데, 이는 Bai[11]이 사용한 방법과 거의 동일하다.

본 논문에서는 Shafiq[4][5] 및 Anselm[3]이 선택한 189개의 원시 특징과 Bai[11]이 선택한 197개의 원시 특징 보다 약간 많은 수의 280개의 특징을 사용하였다. 본 논문에서 선택한 원시 특징의 수가 다소 증가한 이유는 Shafiq[4][5] 및 Anselm[3] 그리고 Bai[11]이 선택한 특징 선택방법과 유사한 방법을 사용하였으나, PE 파일에서 추출 가능한 모든 특징을 추출하였기 때문이다. 본 논문에서 선택한 PE 특징은 표 2에 나타나 있다. 표 2에서 알 수 있듯이 Shafiq[4][5] 및 Anselm[3] 그리고 Bai[11]이 선택한 PE 특징과 거의 유사하나 전체적으로 사용한 필드 수가 많다는 점을 확인할 수 있다.

그림 6은 본 논문에서 선택한 PE 특징을 보여주고 있다.

Referred DLLs	73
File Header Fields	7
Optional Header Fields	61
Section Header Fields	27
Resource Directory Table & Resources	21

그림 6. 본 논문에서 사용한 PE 특징
Fig. 6. Selected PE features for this research

그림 6에서 “참조된 DLL”은 PE 파일의 행위를 예측할 수 있는 DLL로서 각 PE 파일의 Import 섹션을 덤프하여 사용된 모든 DLL을 카운트하여 100회 이하로 사용된 DLL을 배제한 후, 각 DLL의 정보이득을 계산하여 상위 35개를 선택하였다. “파일 헤더 필드”는 COFF 파일 헤더로서 각 필드의 값은 대상 프로세서 유형, 섹션 수, 심볼 수를 나타내는데, 데이터 디렉터리 엔트리 32개와 옵션 헤더 29개 및 옵션 헤더 중에서 DLL 관련 파트를 세분화하여 15개를 선택하였다. “섹션 헤더 필드”는. text, .data, .rsrc 이외에도 .rdata, .reloc, .bss, .edata, .idata, .pdata, .debug 각 섹션의 헤더 필드 120 개를 선택하였다. “리소스 디렉터리 테이블 및 리소스”는 프로그램에서 사용하는 다양한 유형의 자원 수를 나타내며 본 논문에서는 22개를 선택하였다.

기타 특징으로는 NumberOfDLLs, NumberOfAPIs, ProductVersion, DOS_HEADER.HEADER.e_lfanew의 4개를 추가로 선택하였다.

3.4 분류기(Classifier) 구성

샘플을 대상으로 특징 추출 프로그램을 실행하면 각 파일에 대한 특징 목록이 생성된다. 원시 특징 목록은 일반적으로 정확성을 향상시키기 위해 사전 처리한다. 예를 들면, Shafiq[4][5]는 특징 사전 처리를 위해 주성분 분석(PCA) 기법 등을 사용하였다. Yan[13]은 모든 PE 헤더 필드를 숫자 특징으로 사용하였으며, 특성, DLL 및 시스템 호출 정보의 각 비트는 부울 특징으로 표현하여 Relief, 카이제곱(Chi-Square), F-통계 및 L1-정규화된 SVC 특징으로 변환하여 사용하였다. Belaoued[12]는 PE 옵션 헤더 필드에 저장된 정보를 카이 제곱 및 파이 계수를 사용하여 특징을 선택하였다. 이러한 방법으로 선택된 특징은 보안 분석가에게는 “블랙 박스”로 작용하여 보안 분석가가 PE의 어떤 특징이 분류에 이용하였는지 파악하기 어렵다. 그 이유는 원시 특징을 사전 처리하는 경우, 원시 특징 속성을 조합한 형태의 합성 속성이 생성되어 보안 분석가에게 분류과정에서 어떤 특징이 어떻게 분류에 이용되는 지에 대한 정보를 제공할 수 없기 때문이다. 이러한 문제는 분류 알고리즘의 선택에도 동일하게 적용된다. 즉, 딥 러닝에서 사용하는 멀티 레이어 퍼셉트론을 사용하는 경우, 입력 레이어와 출력 레이어 사이에 은닉 레이어가 생성되며 보안 분석가는 은닉 레이어에 대해서는 직관적으로 이해할 수 없다.

이러한 문제가 존재하지 않는 거의 유일한 분류 알고리즘은 결정 트리 알고리즘이다. Anselm[3] 및 Bai[11]은 C4.5(J48) 결정 트리 알고리즘을 사용한 후, 10 중 교차 검증 기법(Han and Kamber[17])을 사용하여 결과를 검증하였는데, 상대적으로 적은 데이터 세트를 사용하는 경우, 일반적으로 "홀드 아웃" 방법과 같은 대안에 비해 10 중 교차 검증을 사용하여 정확도를 추정하는 것이 유리한 것으로 알려져 있기 때문이다.

10 중 교차 검증을 사용하는 이유는 머신 러닝에서 사용자가 사용하는 데이터는 훈련 세트와 테스트 세트인데, 이때 데이터 세트 내 샘플 수는 일반적으로 제한적이다. 따라서 데이터의 양이 충분치 않을 때, 분류기 성능 측정의 통계적 신뢰도를 높이기 위해서 재샘플링(Resampling) 기법을 사용하는데 대표적인 방법으로는 10 중 교차 검증 기법과 부트스트랩(Bootstrap) 기법이 있다. 10 중 교차 검증 기법은 데이터 세트 내에 총 N개의 샘플이 존재하는 경우, 샘플을 10개 집단으로 나누고, 이때, 각 집단의 평균이 비슷한 정도로 나눈다. 분류기를 10-1 (9)개의 집단으로 학습을 시키고 나머지 1개의 집단으로 테스트하여 분류기의 성능을 측정한다. 이 과정을 서로 겹치지 않도록 10번 수행한 후, 10번의 정확도를 평균하여 그 값을 분류기의 성능으로 정의할 수 있다. 교차 검증의 장점은 보유하고 있는 데이터 샘플의 대부분을 학습에 재사용할 수 있다는 점이다.

또 Bai[11]은 필터 기법을 적용하여 197개의 원시 특징 중에서 약 20개를 선택하여 결정 트리 알고리즘을 실험한 경우, 다른 방안들에 비해서 상대적으로 낮은 정확성을 나타냈다.

Anselm[3]이 원시 특징의 정보이득을 특징 평가 척도로 사용하여 특징의 상위 50%, 즉, 약 95개의 특징을 사용한 반면, 본 논문에서는 Anselm[3]이 사용한 정보이득 계산과 유사한 계산을 통하여 상위 10% (26개), 20% (56개) 및 30% (84개) 특징을 선택하는 3가지 방법을 이용하였다.

본 연구에서는 보안 분석가들에게 분류과정에 대한 직관을 제공하기 위해서 결정 트리 알고리즘을 분류기로 사용하였다. 의사 결정 트리 분석은 의사 결정규칙(Decision Rule)을 도표화하여 관심의 대상이 되는 집단을 몇 개의 소집단으로 분류 또는 예측하는 분석방법이다. 일반적으로 의사 결정 트리는 분석하고자 하는 자료의 여러 변수들 중 각 변수들 사이의 연관성 정도에 따라 중요 변수를 선별한 후 선별된 변수를 통하여 트리 구조를 생성하기 때문에 다른 분석방법들에 비해 분류 과정을 쉽게 이해하고 설명할 수 있는 장점이 있다.

머신 러닝 계열에서 널리 사용하는 결정 트리 알고리즘은 ID3(엔트로피), C4.5(정보이득), C5.0(정보이득) 등이며, 결정 트리 알고리즘 중 대표적인 알고리즘 중 하나로 널리 쓰이고 있는 C4.5는 J.Ross Quinlan에 의해 1993년 개발된 알고리즘으로 1986년에 개발된 ID3를 개선한 알고리즘이다. J48은 C4.5와 동일한 결정 트리 알고리즘이며, C5.0은 C4.5 (J48)를 개선하여 개발되었으나, 라이선스를 지불하여야 사용할 수 있으므로, C4.5 (J48)가 널리 이용되고 있으며, 본 논문에서도 J48을 결정 트리 반 데이터 분류 알고리즘으로 사용하였다.

IV. 결과 및 분석

4.1 분류 속도

Shafiq[4]는 파일당 평균 분류 시간과 2가지의 상용 안티바이러스의 평균 파일 스캔 소요 시간을 측정하였다. Shafiq[4]의 파일 당 분류 시간은 약 0.25초가 소요되었으며, 따라서 자신들이 제안한 기법을 실시간 탐지에 이용할 수 있다.

Anselm[3]도 실시간 탐지를 목표로 하였으며, 특징 추출에 평균 0.608초가 소요되었으며, 파일 분류에도 밀리 초(Milliseconds)) 수준이 소요되어 그들의 기법 또한 실시간 판단에 이용할 수 있다.

본 논문에서 사용한 특징 추출 및 분류 기법의 경우에도, 파일 당 약 평균 0.3초 이내의 시간이 소요되어 실시간 판에 적용할 수 있음을 확인하였다.

4.2 결과 분석

Shafiq[4]의 경우, 다양한 특징 처리 방법과 다양한 분류 알고리즘을 실험 하였다. 그 중에서 J48 분류기를 사용한 결과 중에서 가장 좋은 결과는 RFR (중복 특징 제거) 기법을 사용하여 특징을 추출하였을 경우이며, 해당 조건의 탐지 결과는 표 4에 요약되어 있다.

Anselm[3]의 경우, 파일이 패킹되어 있을 경우와 그렇지 않은 경우를 Perdisci[18]이 사용한 특징을 이용하여 먼저 분류(J48 사용)하고 해당 결과를 기반으로 패킹 및 패킹되어 있지 않은 경우에 대해서 각각의 전용 J48분류 모델을 사용하여 실험하였다.

본 논문에서 사용한 PE 파일들은 패킹되어 있는 경우와 패킹되어 있지 않은 경우가 모두 포함되어 있으나, 별도로 분류하지는 않았다. 본 논문의 실험 I(그림 3 참조)에서는 정보이득 상위 10%, 20%, 30% 특징에 대하여 결정 트리(Decision Tree) 기법을 적용하였다.

표 3. 샘플 당 처리 시간 비교
Table 3. Scan time per sample

Methods	Average classification time per file
[4]	0.25 sec.
[3]	0.608 sec.
Our Methods	0.3 sec.

표 4. 비교 대상 연구의 악성 코드 탐지 성능 비교
Table 4. Malicious file collection and verification

Methods		Algorithm	Feature Selection	Accuracy	TPR (%)	FPR (%)
Anselm [3]	Unpacking	J48	Information Gain Top 50%	96.2	96.4	3.9
	Packing			92.7	92.7	7.4
Shafiq[4]		J48	RFR (Redundant Features Reduction)	-*	97.9 ~ 99.8	-*
Bai[14]		J48	Filtering (19 Features)	-*	90.7	2.1

표 5의 실험 결과를 표 4의 Bai[11] 실험 결과와 비교해보면 PE 헤더 필드에서 추출한 197개의 특징과 유사한 초기 특징 세트를 구성한 후, 필터 특징 선택 방법을 사용하여 20개로 줄인 특징 세트를 이용하여 결정 트리에 적용하였을 때 보다 모든 경우에 우수한 결과가 도출됨을 알 수 있다.

표 5에서 확인할 수 있듯이 3 가지 정보이득 기반 특징 선택 방법 중에서 상위 10% 보다는 상위 20% 및 30%를 선택하였을 때, 탐지 성능이 약간 더 우수함을 알 수 있다.

표 5의 수치상으로 상위 30%를 선택했을 경우, 상위 20%를 선택한 경우 보다 근소하게 우수하였다. 따라서 실험 I의 특징 선택 방법 중에서는 정보이득 상위 30%를 사용하는 방법이 가장 우수한 탐지 성능을 나타냈다.

정보이득의 상위 10%, 20%, 30%를 선택하였을 경우에 사용되는 특징의 수는 각각 28개, 56개 및 84개로 약간의 차이가 있으나, PE 파일을 파싱 하는 데 걸리는 시간은 약 0.01초 이하의 차이가 발생하므로 정확도와 처리속도의 트레이드 오프는 거의 발생하지 않는다.

표 5의 실험 I을 종합할 때, 가장 우수한 특징 선택방법은 정보이득을 계산하여 상위 30% 특징을 선택한 경우이다. 따라서 실험 II(그림 3 참조)에서는 악성코드 데이터 집합을 랜섬웨어 데이터 집합으로 변경하고, 정상파일 데이터 집합은 동일한 데이터 집합을 이용하여 실험하였다. 랜섬웨어 집합을 실험한 결과는 표 6과 같다.

그림 7은 실험 II를 위한 렌섬웨어 수집 관련 내용으로 VirusShare에서 랜섬웨어를 수집하여 바이러스토탈의 오픈 API를 이용하여 레이블링하여 실험하였다. 바이러스토탈의 약 69 엔진에서 적어도 5개 이상의 안티바이러스 엔진에서 랜섬웨어로 판명한 데이터 집합을 선택하였고, 정상 파일의 경우에는 실험 I, II 모두 동일한 데이터 집합을 이용하였다.

표 5. 실험 I의 악성 코드 탐지 성능 비교
Table 5. Malware classification performance of experiment I

Experiment		Algorithm	Feature Selection	Accuracy	TPR (%)	FPR (%)	Processing Time
I	Generic Malware	J48	Information Gain Top 10%	96.37	96.37	3.62	Avg. 0.3
			Information Gain Top 20%	98.08	98.08	1.91	Avg. 0.31
			Information Gain Top 30%	98.17	98.17	1.82	Avg. 0.33

표 6. 실험 II의 랜섬웨어 분류 성능
Table 6. Malware classification performance of experiment II

Experiment		Algorithm	Feature Selection	Accuracy	TPR (%)	FPR (%)
II	Ransomware	J48	Information Gain Top 30%	97.78	97.78	2.21

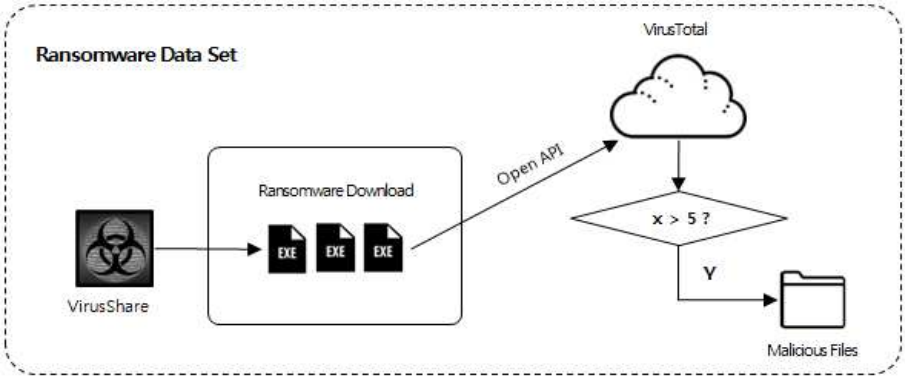

그림 7. 실험 II를 위한 랜섬웨어 수집
Fig. 7. Ransomware collection and verification for experiment II

표 5와 표 6을 비교해 보면, 일반 악성코드(랜섬웨어 제외) 데이터 집합에서 근소하게 우수한 탐지 성능이 나타났음을 알 수 있다. 실험 결과에서 알 수 있듯이 일반 악성코드의 정탐율이 0.39% 우수하며, 오탐율도 0.39%정도 적게 발생하는 것으로 나타났으며, 따라서 전체 정확성이 0.39% 더 우수한 것으로 나타났다. 이러한 현상은 본 실험에서 사용한 랜섬웨어 데이터 집합의 수가 일반 악성코드 데이터 집합에 비해 4,585개 적기 때문에 나타난 현상으로 추정되며 보다 정확히 이유를 판단하기 위하여 향후에는 랜섬웨어 데이터 집합을 일반 악성코드 집합 수준으로 늘려서 실험을 수행할 계획이다.

V. 결 론

본 논문에서는 Anselm[3] 및 Shafiq[4][5]. Bai[11]이 제안한 특징 선택 방법과 유사한 특징을 선택하여 결정 트리 법을 분류알고리즘으로 선택하여 실험한 후, 실험 결과를 Anselm[3]실험 결과 그리고 Bai[11]의 필터기법 적용 후 결정 트리 실험 결과와 비교하였다. 본 논문 제시한 방안의 실험 결과를 분석해 보면 정보이득 상위 30% 특징을 이용하여 Anselm[3] 및 Bai[11]의 가장 좋은 결과보다도 정탐율은 1.77% 개선되었고, 오탐율도 2.08% 개선된 결과를 확보할 수 있었다. 결론적으로 본 연구에서 제안한 방식을 사용하면 98.17%의 탐지율과 1.82 %의 오탐율로 신종 악성코드를 탐지할 수 있다. 마지막으로 동일한 탐지 방법을 최근 심각한 위협으로 대두되고 있는 랜섬웨어 탐지에 적용한 결과, 일반적인 악성코드 탐지의 경우와 유사한 탐지율 및 오탐율을 확보할 수 있었다.

향후 PE 파일이 패킹되어 있는 경우와 그렇지 않은 경우를 구분하고, 특징의 사전처리 기법을 적용하여 실험할 경우, Anselm[3]과 보다 상세한 비교 및 차이점 식별이 가능할 것으로 판단되며, 향후 연구 주제로 남기고자 한다.

References

[1] R. Lyda and J. Hamrock, "Using entropy analysis to find encrypted and packed malware", IEEE Security & Privacy, Vol. 5, No. 2, pp. 40-45, Apr. 2007.

[2] Microsoft. Microsoft PE and COFF Specification. Microsoft Corporation, Redmond, WA, revision 8.2, http://www.microsoft.com/whdc/system/platform/firmware/PECOFF.mspx. [Accessed: Jan. 01, 2010]

[3] T. Anselm and S. Arran, "Human-Readable Real-Time Classifications of Malicious Executables", Proceedings of the 10th Australian Information Security Management Conference, Dec. 2012.

[4] M. Z. Shafiq, S. M. Tabish, F. Mirza, and M. Farooq, "PE-Miner: Mining structural information to detect malicious executables in realtime", Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection (RAID '09), Berlin, Heidelberg, Springer-Verlag, pp. 121-141, Sep. 2009.

[5] M. Shafiq, S. Tabish, and M. Farooq, "PE-Probe: Leveraging Packer Detection and Structural Information to Detect Malicious Portable Executables", Proceedings of Virus Bulletin Conference, pp. 29-33, Jun. 2009.

[6] M. G. Schultz, E. Eskin, F. Zadok, and S. J. Stolfo, "Data mining methods for detection of new malicious executables", In: Security and Privacy, S&P 2001. Proceedings, IEEE Symposium on. IEEE, pp. 38-49, May 2001.

[7] J. Z. Kolter and M. A. Maloof, "Learning to detect malicious executables in the wild", Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining (KDD '04), Seattle, WA, USA. ACM Press. pp. 470-478, 2004.

[8] M. obert, F. lint, T. Nir, B. Eugene, G. Marina, D. Shlomi, and E. Yuval, "Unknown malcode detection using opcode representation", in Intelligence and Security Informatics, D. Ortiz-Arroyo, H. L. Larsen, D. D. Zeng,D. Hicks, and G. Wagner, Eds., Vol. 5376 of Lecture Notes in Computer Science, pp. 204-215, 2008.

[9] M. Robert, S. Dima, F. Clint, N. Nir, J. Nathalie, and E. Yuval, "Unknown malcode detection and the imbalance problem", Journal in Computer Virology, Vol. 5, No. 4, pp. 295-308, Jul. 2009.

[10] S. Asaf, M. Robert, F. Clint, D. Shlomi, and E. Yuval, "Detecting unknown malicious code by applying classification techniques on OpCode patterns", Security Informatics, Vol. 1, No. 1, pp. 1-22, Feb. 2012.

[11] J. Bai, J. Wang, and G. Zou, "A Malware Detection Scheme Based on Mining Format Information", The Scientific World Journal, pp. 1-11, May 2014.

[12] Belaoued and Mazouzi, "A Real-Time PE-Malware Detection System Based on CHI-Square Test and PE-File Features", International Conference on Computer Science and its Applications, IFIPAICT, Vol. 456, pp. 416-425, 2015. DOI: 10.1007/978-3-319-19578-0_34

[13] G. Yan, N. Brown, and D. Kong, "Exploring discriminatory features for automated malware classification", International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Springer, LNCS 7967, pp. 41-61, Jul. 2013.

[14] J. I. Park, H. B. Park, and S. S. Lee, "Study on Automatic Analysis Method Based On Malware Behavior", Proceedings of KIIT Summer Conference, pp. 311-312, Jun. 2017.

[15] A. Baranovich, "VX Heavens", http://vx.netlux.org. [Accessed: Jan. 01, 2012]

[16] D. Quist, "Offensive computing", http://www.offensivecomputing.net. [Accessed: Jan. 01, 2012]

[17] J. Han and M. Kamber, "Data Mining: Concepts and Techniques", Morgan Kaufmann, 2nd edition. 2006.

[18] R. Perdisci, A. Lanzi, and W. Lee, "Classification of packed executables for accurate computer virus detection", Pattern Recognition Letters, Vol. 29, No. 4, pp. 1941-1946, Jun. 2008.

저자소개

전 덕 조 (Deok-Jo Jeon)

1989년 2월 : 아주대학교 전자계산학과(공학사)
1995년 5월 : 뉴맥시코대학교 전자계산학과(이학석사)
2005년 10월 ~ 현재 : (주)씨큐비스타 대표이사
관심분야 : 지능형 위협 대응, 지능형 보안관제, 네트워크 보안

박 동 규 (Dong-Gue Park)

1992년 2월 : 한양대학교 전자공학과(공학박사)
1992년 3월 ~ 현재 : 순천향대학교 정보통신공학과 교수
관심분야 : 제어 시스템 보안, 네트워크보안, 시스템 보안, 모바일 보안