Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.28, NO.5, Oct. 2018 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2018.28.5.1119

악성코드 패킹유형 자동분류 기술 연구*

김 수 정,† 하 지 희, 이 태 진‡
호서대학교

A Study on Automatic Classification Technique of Malware Packing Type*

Su-jeong Kim,† Ji-hee Ha, Tae-jin Lee‡
Hoseo University

요 약

대부분의 침해공격은 악성코드를 통해 발생하고 있으며, 침해공격으로 인한 피해는 사물인터넷/사이버 물리 시스템과 연결되면서 사이버공간에만 국한되지 않고 실생활에 큰 위협이 되고 있다. 이에 따라, 다양한 악성코드 동적분석, 정적분석기술들이 연구되었는데, 악성코드 동적분석들은 결과적인 악성행위를 쉽게 확인할 수 있어 널리 사용되었으나 VM 환경탐지 시 동작하지 않는 anti-VM 악성코드가 증가하면서 어려움을 겪고 있고, 악성코드 정적분석 기술들은 코드자체를 해석할 수 있어 많은 정보를 얻을 수 있으나 난독화, 패킹 기술들이 적용되어 분석가를 어렵게 하고 있다. 본 논문에서는 정적분석기술의 주요 장애물인 난독화 유형을 자동식별, 분류하는 기술을 제안한다. 특히, 제안하는 모델을 통해 알려진 패커나 알려지지 않은 패커와 상관없이 일정한 기준에 의해 모든 악성코드를 분류할 수 있는 것이 가능하다. 악성코드 분류는 다양한 활용이 가능하지만, 예를 들면 악성코드 정적 feature에 기반하여 머신러닝 기반 분석을 할 때, 전체 파일에 대해 학습 및 분석하는 방식보다 악성코드 유형별 학습 및 분석이 더욱 효과적일 것이다. 이를 위해, PE구조에서 활용 가능한 feature에 대해 지도 학습 및 비지도 학습 방식의 모델을 설계했고, 98,000여개 샘플을 통해 결과 검증을 진행하였다.

ABSTRACT

Most of the cyber attacks are caused by malicious codes. The damage caused by cyber attacks are gradually expanded to IoT and CPS, which is not limited to cyberspace but a serious threat to real life. Accordingly, various malicious code analysis techniques have been appeared. Dynamic analysis have been widely used to easily identify the resulting malicious behavior, but are struggling with an increase in Anti-VM malware that is not working in VM environment detection. On the other hand, static analysis has difficulties in analysis due to various packing techniques. In this paper, we proposed malware classification techniques regardless of known packers or unknown packers through the proposed model. To do this, we designed a model of supervised learning and unsupervised learning for the features that can be used in the PE structure, and conducted the results verification through 98,000 samples. It is expected that accurate analysis will be possible through customized analysis technology for each class.

Keywords: Packing, Malware classification, Section name, Clustering, Deep Learning

Received(06. 01. 2018), Modified(1st: 08. 10. 2018, 2nd: 09. 19. 2018), Accepted(09. 19. 2018)
* 이 성과는 2018년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(No. 2018R1C1B5029849).
† 주저자, sjkim395@gmail.com
‡ 교신저자, kinjecs0@gmail.com(Corresponding author)

I. 서 론

AV-Test사의 통계에 따르면 2017년 기준 모든 운영체제의 전체 악성 프로그램 수는 6.4억 개 이상 존재한다. AV-Test사의 분석시스템에서는 하루 평균 35만 개의 새로운 악성코드가 발생한다[1].

이러한 악성코드를 통해 대부분의 침해 공격이 발생하고 있으며, 이로 인한 피해는 PC를 넘어 스마트폰, 사물인터넷(Internet of Things, IoT), 사이버 물리 시스템(Cyber-Physical Systems, CPS)과 연결되면서 사이버공간에만 국한되지 않고 실생활에 큰 위협이 되고 있다. 악성코드들의 대부분은 탐지를 회피하기 위해 기존의 악성코드를 조작한 변종 악성코드가 차지한다. 악성코드를 탐지하기 위한 분석이 한계가 존재하는 이유는 신규 악성코드의 등장도 있겠지만 악성코드가 패킹과 난독화로 발생하는 변종 악성코드들이 기존과 다른 유형을 가지기 때문이다. 패커(packer)는 파일의 크기를 축소시키기 위해 압축을 수행하며, 패커에 의해 패킹된 파일은 언패킹을 수행하지 않으면 코드를 수정하거나 확인할 수 없고 파일을 실행하는 경우에만 코드의 패킹이 해제되면서 메모리에 로드된다. 따라서 패킹된 악성코드는 일반적으로 동적 분석을 활용한 탐지는 가능하나 정적분석에 의한 탐지는 어려움이 존재한다[12]. 변종 악성코드로 인하여 악성코드의 유형이 점차 다양해지고 있으므로 여러 분석 방법을 동원하여 악성코드를 탐지해야 할 필요가 있다.

이에 따라, 다양한 악성코드를 탐지하기 위해 동적분석, 정적분석 기술들이 연구되고 있으나 동적분석의 경우 탐지를 회피하기 위해 VM(Virtual Machine)환경에서 동작하지 않는 anti-VM악성코드가 증가하고, 정적분석의 경우 패킹기술이나 난독화 등이 적용된 변종 악성코드들이 코드 분석을 어렵게 한다. 정적분석기술을 이용하여 패킹기술이나 난독화가 적용된 악성코드를 탐지하기 위해서는 악성코드별 특징에 따른 분석탐지 전략이 필요하다. 이를 위해서는 악성코드의 유형을 분류하는 작업이 선행되어야 한다. Kaspersky Lab은 악성코드 탐지를 위해 기계 학습(machine learning)을 적용할 때 유사한 파일에 대한 분류작업을 수행한다. 이를 위해 LSH(Locality Sensitive Hashing)와 유사한 방식의 유사성 해싱 접근법을 구현하고 2단계 분석을 설계하여 2단계에서 보다 강력한 기계 학습 적용한다[2].

PE(Portable Excutable)파일에 패킹기술을 적용하게 되면 해당 파일의 섹션명은 적용된 패킹기술의 특징을 가질 수 있다. PE파일은 프로그램의 안정성을 고려하여 속성이 서로 다른 코드와 데이터들을 구분하기 위해 각각의 섹션으로 나눠서 저장한다. 섹션의 여러 가지 특징 중 섹션명은 사용된 패킹기술과 데이터의 구성과 밀접한 관련이 있으므로 본 논문에서는 섹션명으로 생성한 feature를 이용하여 정적분석기술의 주요 장애물인 난독화 유형을 자동식별, 분류하는 악성코드 유형별 분류기술을 제안한다.

II. 관련 연구

2.1 관련 연구

악성코드의 정적 분석을 이용한 탐지방식에는 시그니처 기반 방식과 기계 학습 기반 방식 같은 것들이 이용되고 있다. 기존 AV 프로그램에서 사용하던 시그니처 기반 탐지기법은 수집된 악성코드의 특징을 분석하여 사전에 생성한 시그니처에 의존하기 때문에 최근에는 신규 악성코드나 변종 악성코드가 탐지를 회피할 수 있으므로 비효율적이라는 평가를 듣고 있다[3]. Schultz 등이 진행한 악성코드 식별 연구에서는 시그니처 기반 탐지기법의 성능이 49.28%의 정확도를 보였다[4].

시그니처 기반 기법에 속하는 파일 해싱을 이용한 분석의 경우 사전에 해싱한 데이터로 패턴을 생성해 두었다가 분석대상 파일의 해시값과 비교하여 얼마나 유사한지 확인한다. 악성코드들의 유사성을 빠르게 식별하기 위한 퍼지 해싱(fuzzy hashing) 방식의 탐지기법은 파일을 블록단위로 해시한다[6]. N-gram 기법으로 시그니처를 생성하여 유사도를 비교하는 탐지분석 기법의 연구도 이루어지고 있다. N-gram은 데이터를 N의 길이로 나누면서 1씩 shift하는 방법이다. 예를 들어, 'abcdef' 문자열에 4-gram을 적용한다면 'abcd', 'bcde', 'cdef'로 나뉘게 된다. N-gram을 이용하여 시그니처를 생성하는 경우 N-gram을 추출하는 소요시간이 수행시간의 대부분을 차지한다는 단점이 존재한다[7]. 전통적인 시그니처 기반 방식의 탐지기법은 신규 및 변종 악성코드를 탐지하는 데 한계가 존재하기 때문에 여러 개선 방안들이 연구되고 있다.

기계 학습을 이용한 연구로는 결정 트리 기법을 활용하여 악성코드와 정상파일 사이의 분류를 진행한

연구[3]와 동적 방식으로 feature를 추출하고 클러스터링 기법으로 악성코드를 분류하는 연구[5], DNN(Deep Neural Network) 기법을 활용하거나[8], CNN(Convolutional Neural Network)를 활용하여 악성코드를 탐지하는 연구[9] 등이 진행되고 있다.

지속적으로 발생하는 신규 및 변종 악성코드의 발생으로 패킹된 변종 악성코드의 경우 정적분석 탐지에 어려움이 존재한다. 정적분석이 가능하도록 하기 위해 패킹여부 확인 후 언패킹하는 기술을 추가하거나 악성코드의 언패킹을 자동화하는 것과 같은 연구가 진행되고 있다[12,13].

Kaspersky Lab은 유사한 파일을 분류하기 위해 LSH을 기반으로 한 해시 매핑 기법을 이용하여 파일을 분류한다[2]. 유사한 데이터 해시를 같은 bucket에 매핑하는 기법인 LSH를 이용하여 유사한 파일을 근접한 위치로 매핑하는 1단계 파일 분류를 진행한다. 악성코드와 정상파일이 다른 경향을 띄는 경우에 1단계 방식에서 충분히 분류가 되지만 악성코드와 정상파일이 유사한 경우 이러한 방식으로는 구분이 어렵다. 파일의 분류만으로는 탐지분석이 완벽하게 완료되지 않으므로 2단계 분석 설계로 기계학습을 이용한 탐지기법을 적용한다. 결과적으로 1단계에서 경량화한 탐지기법으로 악성코드 여부를 판단하며, 충분한 탐지결과가 나타나지 않는 판단이 어려운 유형에 강력한 기계 학습 모델을 이용하여 2단계 분석을 진행한다.

최근에는 악성코드 탐지를 위해 여러 탐지기법이 연구되고 있다. 본 논문에서는 시그니처 기반 방식이나 기계 학습과 같은 여러 종류의 탐지기법들을 파일의 유형별로 적용할 수 있도록 파일을 유형별로 분류한다. 취약한 유형의 파일에만 강력한 탐지기법을 적용하면 탐지정확도를 높이면서도 탐지시간을 크게 늘이지 않을 수 있다. 또한 파일들이 알려지지 않은 패커로 패킹된 경우에도 특정 유형으로 분류되어 해당 유형의 파일에 적합한 탐지기법을 적용할 수 있도록 하는 취약한 class 식별 연구를 진행한다.

2.2 패킹과 섹션의 관계 분석

하나의 파일에 존재하는 여러 섹션들은 각각 execute, read, write와 같은 속성을 통해 각각의 코드와 데이터를 구분한다. 알려진 섹션명의 경우 이미 알려진 기능에 따라 특징의 추측이 가능하며, 이러한 특징에 기반하여 섹션명을 이용하면 해당 파일이 지니는 속성도 파악이 가능하다. 이처럼 섹션은 동일한 성질의 데이터가 저장되어 있는 영역으로 예를 들어 .edata는 Export data section으로 Export할 API(Application Programming Interface)에 대한 정보를 담고 있으며 .idata 섹션은 Import할 DLL(Dynamic-Link Library)과 관련 API들에 대한 정보를 담고 있는 영역이다[10,11].

악성코드 분석 시 패킹기술이 적용된 변종 악성코드들은 섹션명을 이용하여 클러스터링하면 데이터의 유형별로 분류할 수 있다. 패커 종류에 따라 생성되는 섹션명에 일정한 규칙이 존재함을 확인하고 이를 활용하여 파일을 패킹기술을 기준으로 유형별 분류하여 취약한 유형을 파악하고자 한다. Table 1.을 통해 패킹기술에 따라 섹션명이 달라지는 것을 확인할 수 있다. aspack 패커는 기존 파일이 가지고 있는 섹션에 .aspack, .adata 섹션이 추가되고, nspack 패커는 기존에 가지고 있던 섹션이 사라지고 .nsp0, .nsp1, nsp2 섹션으로 대체된다. pecompact 패커는 .text섹션과 .rsrc섹션, .reloc 섹션으로 대체되며, petite 패커는 .petite섹션과 이름이 없는 섹션으로 대체되는 것을 확인할 수 있다. Table 1.의 *은 섹션명으로 공백을 가지는 섹션을 나타낸다.

Table 1. Distribution of Section by Packing Technique

file name	packer	section name
backdoor.win32.bo2k.10	unpacking	.text .rdata .data .plugins
backdoor.win32.bo2k.10.packed.aspack.1	aspack	.text .rdata .data .plugins .aspack .adata
backdoor.win32.bo2k.10.packed.nspack.1	nspack	.nsp0 .nsp1 .nsp2
backdoor.win32.bo2k.10.packed.pecompact.1	pecompact	.text .rsrc
backdoor.win32.bo2k.10.packed.petite.1	petite	.petite (NULL)*

III. 제안 모델

패킹기술이 적용된 악성코드의 탐지를 개선하기 위해서는 패킹기술의 유형별 분류가 필요하다. 본 장에서는 다양한 악성코드를 유형에 따라 파일을 구분할 수 있도록 2차원 좌표값을 이용하여 class 값을 생성하고 이에 따른 유형별 분류에 대해 시각적으로 확인하고 분석하는 모델을 제안한다.

Fig.1.과 같은 유형별 분류기법은 알려진 패킹기술뿐 아니라 알려지지 않은 패킹기술이 적용된 파일도 유사한 그룹끼리 분류할 수 있다. 사용된 패킹기술을 모르더라도 파일을 분류할 수 있도록 패킹기술의 특징이 드러나는 섹션명을 활용한다. 섹션들 중 파일의 코드와 데이터에 관련된 섹션 9개를 알려진 섹션, 나머지 섹션은 알려지지 않은 섹션으로 정하고 각각 x축과 y축값을 생성하는 feature로 이용한다. x축, y축의 값을 이용하여 class 값을 생성하고 2차원 형태로 파일을 시각화하여 분류된 파일의 취약한 유형을 파악하고 개선책을 제안한다.

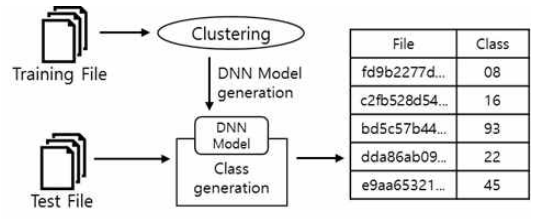

Fig. 1. System configuration

3.1 feature 분류 기준

PE파일은 코드 섹션과 여러 가지 다양한 데이터 섹션으로 구성된 파일로써 .text 섹션은 코드 섹션에 해당하며 .data, .rdata 등의 섹션은 데이터 섹션에 해당한다. 넓은 관점에서 보면 .idata와 .edata, .rsrc섹션 또한 데이터 섹션에 포함된다고 볼 수 있다. 섹션명을 이용하면 패킹과 난독화 기술이 적용된 파일 분류뿐만 아니라 모든 파일에 대해 데이터 유형별 분류가 가능하다. 여러 섹션들 중 일반적으로 앞쪽에 위치한 코드와 데이터 관련 섹션을 기준으로 대표적인 9개의 섹션을 Fig.2.의 좌측과 같이 알려진 섹션 리스트로 정하고 알려진 섹션 리스트에 포함되지 않는 섹션명은 알려지지 않은 섹션명으로 한다.

Known_Section
.bss
.data
.edata
.idata
.rdata
.rsrc
.text
.reloc
.tls

UnKnown_Section
V7#x/ao%
eIOW*Hp;
F;S16Y?y
.kmhjnn
.rttgf
.W
.rMBF
kakxcjb
:

Fig. 2. List of known and unknown section names

3.2 악성코드 유형별 class 분류기법

패킹기술과 난독화 기법에 따라 탐지성능은 한계를 보이기 때문에 해당 특성에 따라 달라지는 섹션명을 이용하여 파일을 분류하게 된다. 섹션명을 이용하여 유형별 분류를 진행하게 되면 적용된 패킹기술이 무엇인지 파악하지 않고 분류를 할 수 있다. 분류를 위해 x축과 y축의 좌표값으로 파일의 2차원 매핑이 가능하며 해당하는 좌표값을 class값으로 변경하여 파일의 분류를 용이하게 한다.

패킹 및 난독화 기술이 적용된 파일 간의 분류인 class 분류는 알려진 섹션명을 x축, 알려지지 않은 섹션명을 y축으로 사용한다. x축과 y축으로 분류하면 파일에 알려진 섹션명과 알려지지 않은 섹션명이 쓰이는 경우 두 가지를 모두 고려하여 2차원 방식으로 분류할 수 있다.

3.2.1 유형별 class 매핑

파일의 그룹을 분류하고 특정 기준으로 분류된 파일 목록에 label을 붙이는 방식으로 좌표값을 생성하기 위해 클러스터링 기법의 하나인 k-means알고리즘을 이용한다. k-means는 k개의 클러스터로 데이터를 그룹 지어 분류시키는 알고리즘으로, 각 클러스터와 거리 차이의 분산을 최소화하는 방식으로 동작한다. 사전에 학습이 필요하지 않은 비지도 학습의 일종이기 때문에 파일의 label을 모르는 상태의 파일을 분류하여 label을 부여할 수 있다. k-means를 이용하면 feature 데이터에 따라 파일을 0부터 k-1사이인 k개를 가지는 값의 클러스터로 분류하며, 파일이 해당하는 클러스터의 값이 파일의 label이 된다. 파일의 feature를 생성하여 k-means 알고리즘으로 획득한 label은 파일의 2차원 분류를 위한 좌표값으로 사용한다.

학습에 사용된 78,630개 실험데이터를 분석한 결과 99종류의 packer가 사용된 것을 확인하여 파일을 100여개의 유형으로 분류할 수 있다고 판단하여 본 논문에서는 class를 100개로 분류한다. 실 환경에서는 알려지지 않은 패킹을 포함한 다양한 기법이 존재할 것으로 예상하나, class는 패커와 별개의 일정한 기준으로 분류되며, 추후 class별 맞춤형 보안 대책을 위해서는 너무 많은 분류가 오히려 부담될 수 있다. 따라서 k값을 10으로 설정하여 x축과 y축을 0부터 9 사이의 값으로 추출하고 class값을 0부터 99까지 총 100개로 분류하였다. class는 수식(1)로 값을 구한다.

$$Class = XLabel \times 10 + YLabel \quad (1)$$

k-means는 데이터 증가에 따라 새로운 클러스터를 형성하는 것은 불가능하므로 좌표값을 저장하여 이후에 활용하기 위해 지도 학습 방식인 DNN 알고리즘으로 학습을 시킨다. DNN은 모델 학습 시 label이 필요하며 학습된 모델을 이용하면 신규 파일의 유입 시 학습된 모델로 기존의 k-means의 결과와 유사한 좌표값을 부여할 수 있다. 이러한 방법으로 파일의 갱신이 가능한 2차원 그룹분류의 구성이 가능하다. Fig.1.은 파일의 섹션명을 추출하여 가공 후 클러스터링과 DNN으로 좌표값을 추출하는 것을 나타낸다.

3.2.2 고정된 feature 기반 X축 식별방안

알려진 섹션명은 Fig.2.의 좌측과 같이 섹션명의 목록이 지정되어 있기 때문에 파일에 해당하는 목록의 섹션명이 존재하는지 여부를 이용하여 파일의 그룹을 분류한다. 예를 들어 A파일에 .bss, .rdata라는 섹션명이 존재한다면 A파일의 feature는 1,0,0,0,1,0,0,0,0이 된다. 존재 여부를 카운트하기 때문에 x축을 위한 feature 값은 0과 1로 이루어지게 된다.

3.2.3 고정되지 않은 feature 기반 Y축 식별방안

y축은 알려지지 않은 섹션명을 이용하게 되는데 알려지지 않은 섹션명은 Fig.2.의 우측과 같이 형식이 일정하지 않아 x축처럼 존재 여부를 카운트하여 사용하기 어렵다. 알려지지 않은 섹션명으로 y축을 구성하기 위해서는 카운트가 가능한 형태로 가공이 필요하며, 가공법으로 n-gram기법을 활용하였다. n-gram은 문자열을 n의 길이씩 잘라주는 기법으로 Fig.3.와 같이 문자열을 n의 길이로 한 글자씩 shift 하여 자르게 되는데, 알려지지 않은 섹션명은 파일마다 가진 섹션명의 종류와 길이가 다양하기 때문에 섹션명을 하나의 문자열로 연결한 후 n-gram으로 섹션명을 n의 길이로 자른다. n의 길이가 된 단어들은 vector화 시키는 것으로 feature를 생성하기 위해 단어장을 기준으로 존재 여부를 카운트한다. 단어장은 n-gram 기법을 이용하여 사전학습데이터로 n 길이의 문자열 결과를 생성한 후 중복을 제거하여 만들어진다. 테스트할 파일들도 n-gram을 이용하여 n 길이의 문자열로 나눈 후 단어장에 존재하는 문자열에 해당하는 개수를 카운트하여 y축의 feature를 생성하게 된다.

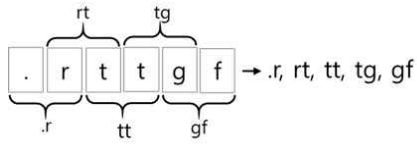

Fig. 3. An example of n-gram techniques using unknown section names

IV. 실험결과

4.1 실험환경

실험데이터는 KISA에서 제공하는 대용량 정상, 악성파일 샘플 15,000개와 G사의 샘플 83,128개를 이용하여 총 98,128개의 데이터로 진행하였다. 실험

Table 2. The number of experimental data

packer group	total	malware	normal
unpacking	3963	3004	959
msvisualc	8403	5740	2663
borland	979	705	274
upx	1551	1408	143
pecompact	555	545	10
aspack	159	154	5
nspack	47	47	0
asprotect	44	33	11
themida	7	7	0
the others	3790	2919	871

에 사용한 데이터의 개수는 악성코드 58,277개, 정상 39,851개이며, 이 중 학습을 위해 사용한 사전학습데이터는 악성코드 43,715개, 정상 34,915개이고 나머지 악성코드 14,562개와 정상 4,936개는 실험데이터로 사용했다. 기계 학습에서 악성코드와 정상파일의 비율이 비슷해야 오탐율이 높아지지 않기 때문에 사전학습데이터의 정상파일이 실험데이터의 정상파일보다 비중이 클 수 있도록 분류했다. 실험데이터들은 패킹기술인 upx와 pecompact, aspack 등을 적용한 파일들과 패킹 되지 않은 언패킹, 컴파일러가 사용된 msvisualC 및 borland가 주를 이룬다. Table 2.는 실험데이터로 사용한 19,498개의 파일들을 적용기법으로 분류한다.

4.2 주요 패커 분포분석

Table 3.는 실험데이터들을 class 값으로 분류한 후 분류된 데이터들의 개수가 100개 이상인 class에서 패커별 파일이 분포한 비율을 표시한 결과이다. 대표적인 class를 확인해보면 upx로 패킹된 파일의 경우 Table 2.와 같이 1551개의 파일이 있으며, class16에 그중 91.68%가 분포된다. pecompact의 경우 class82에 71.5% class25에 24.32%에 집중적으로 분포되어 있다. Table 3.를 통해 섹션명을 이용하면 파일에 적용된 패킹기술을 알지 못하더라도 비슷한 유형의 파일들이 유사한 성향을 띄고 있음을 검증하였다.

Table 3. Distribution of packers by class

class	upx	pe compact	as pack	ns pack	as protect
01	0.39	0	4.40	0	0
05	0	0	0.63	0	0
12	2.97	3.96	10.06	100	13.64
16	91.68	0	0	0	0
25	0	24.32	0	0	0
34	0.06	0	0	0	0
47	0.19	0	18.24	0	0
53	0	0	15.72	0	0
62	0.19	0	0	0	0
70	0	0	1.26	0	0
74	0.06	0	33.96	0	2.27
82	4.13	71.53	0	0	0
93	0	0	1.26	0	0

4.3 class 결과 분석

Table 4.는 실험데이터를 대상으로 하여 Table 3.와 동일한 class들의 패커별 기계 학습을 이용한 탐지기법의 정확도를 나타낸다. Table 3.의 파일 분포와 연계하여 보면 class82에서 pecompact의 정확도가 93.70%이면서 파일의 분포는 71.53%이다. 이러한 class는 일정량 이상의 파일이 분포되어 있고, 다른 class에 비해 정확도가 높지 않아 탐지에 취약한지 확인할 수 있다.

Fig.4.는 Table 4.의 기계 학습 정확도를 0부터 1 사이의 값으로 변경하여 그린 heatmap 그래프이

Table 4. Accuracy of packers by class

class	upx	pe compact	as pack	ns pack	as protect
01	100	0	100	0	0
05	0	0	100	0	0
12	100	100	100	100	50
16	96.98	0	0	0	0
25	0	99.26	0	0	0
34	100	0	0	0	0
47	100	0	96.55	0	0
53	0	0	96	0	0
62	100	0	0	0	0
70	0	0	100	0	0
74	100	0	100	0	100
82	100	93.70	0	0	0
93	0	0	100	0	0

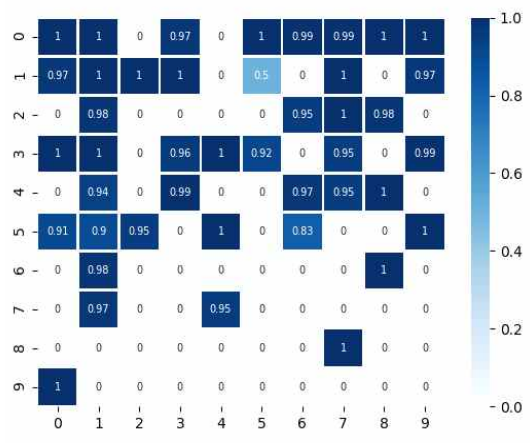

Fig. 4. Heatmap graph according to machine learning accuracy

다. heatmap 그래프를 통해 2차원 분류에 따른 결과를 시각적으로 이해할 수 있다. x축이 5이고 y축이 2인 class51에 0.5는 정확도가 약 50%인 것을 의미하며 시각적으로 색상의 차이를 통해 정확도가 낮은 class를 쉽게 찾을 수 있다.

4.3.1 분석기술을 이용한 class별 정확도 분석

class별 정확도를 분석하기 위하여 ssdeep과 TLSH(Trend micro Locality Sensitive Hashing), 기계 학습을 이용하였다. ssdeep과 TLSH는 파일이 유사할수록 해쉬값도 유사하게 계산되는 퍼지 해싱 방식을 이용하여 파일의 해시 시그니처를 생성하고 실험데이터에 대해 생성된 시그니처와 유사한 정도를 판단해주는 도구이다.

Table 5.의 커버리지(coverage)는 ssdeep과 TLSH가 파일 개수의 데이터에서 탐지한 데이터의 비율을 나타내고, 파일 개수는 class별로 분류된 파일의 개수이다. ssdeep과 TLSH는 해쉬값을 비교하여 파일의 유사도를 계산하며, 유사도 값은 ssdeep의 경우 100에 가까울수록 TLSH의 경우 0에 가까울수록 유사한 파일임을 나타낸다. 유사하지 않은 파일의 분석 결과를 적용하는 것은 결과의 신뢰도를 하락시키기 때문에 본 논문의 실험에서는 ssdeep의 유사도는 90이상인 경우, TLSH의 유사도는 10이하인 경우의 결과에 대해서만 탐지한다. 기계 학습의 경우 입력되는 모든 파일의 결과를 도출하므로 커버리지는 100이 된다.

악성코드 탐지분석 결과인 정확도를 이용하는 것으로도 취약한 class를 파악할 수 있다. Table 5.는 ssdeep, TLSH, 기계 학습을 이용한 탐지기법의 정확도를 class별로 분류한 실험데이터로 낸 결과이다. 또한, 여러 탐지기법을 사용한다면 취약한 class에 대해 별도의 정책을 적용할 수 있다.

V. 결 론

본 논문에서는 유형별 class 분류기법을 활용하여 파일에 사용된 패킹기술 등을 알지 못하더라도 유사한 파일들을 유형별로 분류하여 취약한 class를 파악할 수 있는 기법을 제안하였다. 제안된 방법은 8만여개의 사전학습데이터를 활용하여 클러스터링으로 좌표값을 부여할 수 있는 모델을 생성하고, 2만여개의 실험데이터의 class 값으로 파일을 분류하였다. 분류된 파일에서 특정 패킹기술이 사용된 파일들은 특정한 class에 분류되는 것을 확인하여, 파일의 유형별 분류가 가능함을 검증했다. 또한, 파일이 분포된 정도와 여러 탐지기법의 커버리지와 정확도를 이용하여 취약한 class를 파악하였다.

특정 class가 취약한 것을 확인하는 방법으로 해당 class의 패킹비율에 대한 탐지기법의 정확도와 커버리지를 이용하면 된다. 해당 class에 속하는 파일의 비율이 높고 분석정확도가 낮을수록 취약한

Table 5. Accuracy of detection techniques by class

class	number of files	ssdeep coverage	TLSH coverage	ssdeep accuracy	TLSH accuracy	machine learning accuracy
01	4792	39.52	42.49	99.84	99.95	97.45
05	184	29.35	31.52	96.30	96.55	90.76
12	917	28.79	26.28	100	100	98.36
16	1588	48.80	41.69	100	100	97.73
25	1746	31.90	34.42	100	100	94.67
34	275	22.18	5.82	100	100	98.91
47	1013	60.12	46.99	99.51	98.95	94.87
53	401	25.44	24.94	100	100	91.77
62	940	35.43	33.83	100	100	95.21
70	347	44.38	38.62	100	100	98.85
74	3294	37.19	34.03	99.92	99.91	94.54
82	687	27.07	26.20	99.46	99.44	98.40
93	178	25.28	28.09	100	100	98.88

class라고 볼 수 있다. 취약한 class에는 기존의 탐지분석 기술보다 강력한 기술을 적용하거나 여러 기술의 앙상블을 이용하는 등 취약한 파일들을 집중적으로 연구하여 새로운 탐지분석 방식을 통한 개선이 가능하다. 특정 class의 탐지분석 개선을 위한 방법으로 기계 학습에 이용되는 유의미한 feature를 늘리거나 동적 분석 활용을 통한 강력한 모델을 생성할 수 있고, 탐지율과 오탐율의 기준이 되는 threshold를 다시 정할 수 있다. ssdeep, TLSH과 같은 LSH 기법을 활용하는 방식은 파일의 유사도를 완화하여 적용할 수 있으며, 분석탐지 정책 측면에서 여러 기법을 앙상블하는 경우 적합한 탐지기법별 가중치를 부여할 수 있다. 전체 파일들을 대상으로 강력한 기술을 적용하기에는 탐지 정확도의 상승에 비례해 탐지시간이 커질 수 있으나 이처럼 특정 취약한 class에만 적용한다면 탐지 성능이 개선될 수 있을 것이다.

References

[1] AV-TEST, The AV-TEST security report 2016/2017, AV-TEST report, Jul. 2017.

[2] Kaspersky Lab, Machine learning for malware detection, Kaspersky Lab, 2017.

[3] Deok-jo Jeon and Dong-gue Park, "Real-time malware detection method using machine learning," *The Journal of Korean Institute of Information Technology*, 16(3), pp. 101-113, Mar. 2018.

[4] M.G. Schultz, E. Eskin, F. Zadok, and S.J. Stolfo, "Data mining methods for detection of new malicious executables," *Proceedings of the 2001 IEEE Symposium on Security and Privacy*, pp. 38 - 49, Aug. 2001.

[5] U. Bayer, P.M. Comparetti, C. Hlauschek, C. Kruegel, and E. Kirda, "Scalable, Behavior-based malware clustering," *NDSS*, vol. 9, pp. 8 - 11, Feb. 2009.

[6] Chang-wook Park, Hyun-ji Chung, Kwang-seok Seo and Sang-jin Lee, "Research on the classification model of similarity malware using fuzzy hash," *Journal of the Korea Institute of Information Security & Cryptology*, 22(6), pp. 1325-1336, Jan. 2012.

[7] Hee-jun Kwon, Sun-woo Kim and Eul-gyu Im, "An malware classification system using multi n-gram," *Journal of Security Engineering*, 9(6), pp. 531-542, Dec. 2012.

[8] J. Saxe, and K. Berlin, "Deep neural network based malware detection using two dimensional binary program features," *Proceedings of the 10th International Conference on Malicious and Unwanted Software,* pp. 11-20, Oct. 2015.

[9] D. Gibert, "Convolutional neural networks for malware classification," master's thesis, Computer Science Department, Universitat Politècnica de Catalunya, Oct. 2016.

[10] Ho-dong Lee, Reverse engineering 1 (file structure section), Hanbit Media, Oct. 2016.

[11] Ho-dong Lee, Structure and principles of windows system executables, Hanbit Media, May 2005.

[12] Hea-eun Moon, Joon-young Sung, Hyun-sik Lee, Gyeong-ik Jang, Ki-yong Kwak and Sang-tae Woo, "Identification of Attack Group using Malware and Packer Detection," *Journal of KIISE*, 45(2), pp. 106-112, Feb. 2018.

[13] Seon-gyun Kim, "Design and Implementation of PE File Unpacking Automatic System for Malware Analysis," master's thesis, Kangwon National University, Feb. 2018.

〈저자소개〉

김 수 정 (Su-jeong Kim) 학생회원
2018년 2월: 호서대학교 정보보호학과 졸업
2018년 3월~현재: 호서대학교 정보보호학과 석사과정
〈관심분야〉 시스템 보안, 악성코드 분석, 기계학습

하 지 희 (Ji-hee Ha) 학생회원
2018년 2월: 호서대학교 정보보호학과 졸업
2018년 3월~현재: 호서대학교 정보보호학과 석사과정
〈관심분야〉 정보보호, 악성코드 분석, 암호학

이 태 진 (Taejin Lee) 종신회원
2003년 1월~2017년 2월: 한국인터넷진흥원 팀장
2017년 3월~현재: 호서대학교 정보보호학과
〈관심분야〉 시스템 보안, 악성코드 분석, 침해사고 대응