머신러닝을 이용한 지능형 악성코드 분석기술 동향

저자
(Authors)

이태진
Taejin Lee

출처
(Source)

정보보호학회지 28(2), 2018.4, 12-19 (8 pages)
REVIEW OF KIISC 28(2), 2018.4, 12-19 (8 pages)

발행처
(Publisher)

한국정보보호학회
Korea Institute Of Information Security And Cryptology

URL

http://www.dbpia.co.kr/Article/NODE07424018

APA Style

이태진 (2018). 머신러닝을 이용한 지능형 악성코드 분석기술 동향. 정보보호학회지, 28(2), 12-19.

정 보 보 호 학 회 지
제28권 제2호, 2018. 4

머신러닝을 이용한 지능형 악성코드 분석기술 동향

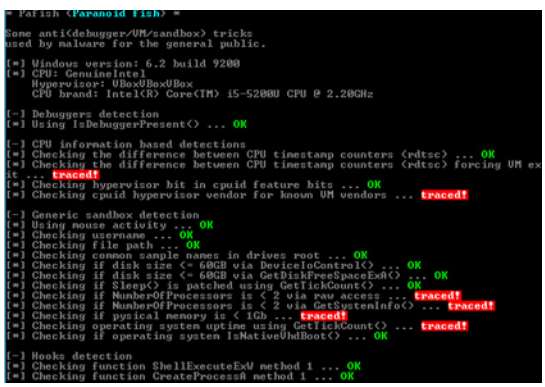
이 태 진*

요 약

사이버 침해공격은 단순히 사이버 공간에만 피해를 주는 것이 아니라, IoT/CPS와 연결되면서 실생활에 큰 피해를 줄 수 있는 중요한 문제로 대두되었다. 이러한 사이버 침해공격의 대부분은 악성코드를 사용하고 있으며, 점차 지능화된 형태로 발전하고 있다. 이에 대응하고자 다양한 악성코드 분석기술이 출현해왔으며, 최근의 연구들은 대부분 머신러닝을 이용하여 기존에 진행했던 Pattern, Heuristic 기반의 한계들을 보완하려 노력하고 있다. 본 논문에서는 머신러닝을 이용한 악성코드 분석기술의 동향을 기술하였다. 특히, 머신러닝을 이용한 악성코드 분석 목적을 7개로 분류하였고, 악성코드 분석에 핵심이 되는 Key Feature들에 대해 소개하였다. 본 논문을 통해, 다양한 악성코드 분석 방법에 있어 새로운 Approach로 연결되는 계기가 되기를 기대한다.

Ⅰ. 서 론

사이버 공격은 매년 큰 폭으로 증가할 뿐 아니라 전기, 가스 및 수도 등 사회 기반시설이 모두 연결되어 가면서 사이버 상의 피해를 넘어서 우리의 삶 전반에 큰 위협이 되고 있다. 이러한 사이버 공격은 대부분 악성코드를 통해 발생하고 있으며, 그 숫자는 일평균 160만개를 넘어서고 있다. 이렇게 끊임없이 쏟아지는 대량의 악성코드를 대응하기 위해서는 자동화된 악성코드 분석기술이 매우 중요한 의미를 가지고 있다. 악성코드 분석기술의 결과물은 지능화된 형태로 진화하는 악성코드에 보조를 맞춰 보안솔루션들을 지속 갱신하고, 이를 통해 새로운 보안사고를 사전에 대응할 수 있어야 한다. 이에 대응하고자 다양한 형태의 악성코드 분석기술들이 출현하고 있다. 기존에 연구된 악성코드 정적 분석기술은 빠른속도와 코드에 대한 해석이 가능하지만, 암호화나 패킹 등 난독화 된 파일에 대한 해석에 어려움이 있고, 동적 분석기술은 결과적인 행위 기반으로 판단하기 때문에 난독화 이슈는 없지만, Logic bomb 등을 포함한 Anti-VM 기법에 해결책을 마련하기 쉽지 않다. 한 예로 Panaroid Fish는 현재의 VM 환경의 취약한 정보를 자동으로 check하여, 분석을 회피할 수 있는 악성코드 제작을 도와준다. 아래의 그림은 Panaroid Fish의 동작 화면 예시를 나타낸다.



〔그림 1〕 Anti-VM을 위한 Panaroid Fish 분석도구

이와 같이 정적, 동적 분석기술의 장단점이 존재하는 현실에서 머신러닝 기반 분석기술이 악성코드 분야에도 깊숙이 들어왔다. 이에, 본 논문에서는 머신러닝 기법을 이용한 악성코드 분석 기술 동향에 대해 기술할 예정이다. 본 논문의 목차는 다음과 같다. 2장에서는 머신러닝 기반 악성코드 분석의 목적을 7개로 분류하여 설명하였고, 3장에서는 머신러닝 분석에 쓰이는 다양한 악성코드 Feature들에 대해서 자세히 설명하였고, 4장에서는 머신러닝 기반 악성코드 분석의 연구흐름을 정리하였다.

* 호서대학교 컴퓨터정보공학부 (kinjecs0@gmail.com)

Ⅱ. 악성코드 분석 목적

악성코드 분석 목적은 지능화된 형태로 나타나는 신규 보안위협을 침해사고 발생하기 전에 대응하는데 있다. 이를 위한 악성코드 분석기술의 목적은 세부적으로 7개로 분류할 수 있다[15]. 가장 기본적인 악성여부 탐지기술부터 악성코드의 변종여부 분석 및 그룹 분류, 악성코드 주요행위별 유형분류, 악성코드에서의 추가/변경된 부분의 탐색, 신규 악성코드 특징 분석, 악성코드 공격그룹 분석, 대용량 악성코드 분류 등으로 하나씩 설명한다.

- Malware Detection

주어진 파일이 악성인지 아닌지에 대한 판단결과를 도출하는 것으로, 악성코드 분석의 가장 기본적이면서 중요한 목적이다. 악성코드의 다양한 생태계에서 다양한 요소기술들이 필요하지만, 크게 보면 악성여부 판단을 잘해서 보호하고자 하는 시스템을 하는 것으로 볼 수 있다.

- Malware Variants Detection

공격자 입장에서 기 제작된 코드를 활용하여 악성코드을 만드는 것이 비용이 가장 적게 들면서 효과적인 방법이다. 세부적으로는 2개로 분류할 수 있는데, 분석대상 파일이 기존에 알려진 악성코드의 변종임을 알고 있다면, 분석시간이 크게 단축되고 기존의 분석보고서를 활용할 수 있어 매우 유용하다.

• Variants Selection : 기존에 확보하고 있는 악성코드 Pool을 대상으로 분석대상 파일과 가장 가까운 파일을 선별하는 것을 목적으로 한다.

• Families Selection : 기존에 분류하고 있는 악성코드 그룹을 대상으로 분석대상 파일이 속해있는 그룹을 선별하는 것을 목적으로 한다.

- Malware Category Detection

앞서 기술한 악성코드 변종, 그룹분류의 관점과 달리 악성코드가 결과적으로 지향하고 있는 주요 핵심행위(Prominant Behavior)와 목적(Objectives)을 기준으로 분류하는 것이 필요하다. 예를 들어, 정보를 탈취하는 Spyware, 파일을 암호화하고 금전을 요구하는 Ransomware, 원격제어와 관련된 Remote access trojans 등으로 구분할 수 있는데, 아무리 복잡하게 설계된 악성코드라도 기 식별 및 분류된 Category에 대한 지식이 있다면, rough하긴 하지만 대단히 분석에 유용한 방향성을 제공한다. 비록, 보안업체별 Malware Category에 대한 분류가 표준화 되어있지 않지만, 여전히 유용한 것은 사실이다.

- Malware Novelty and Similarity Detection

앞서 악성코드 간의 변종식별, 그룹분류와 같은 파일의 유사성 분석에 관하여 분석결과를 제시하지만, 약간 다른 목적을 가지고 있다. 2개의 파일에서 무엇이 유사한 부분인지, 무엇이 다른 부분인지 코드레벨에서 자동으로 분석하는 것이 매우 중요하다.

• Similarity Detection : 기존에 분석된 지식과 연계하여, 분석대상 파일의 공통된 부분을 식별하고, 그대로 의미를 부여할 수 있는 부분, 분석할 필요가 없는 부분을 빠르게 분류 처리한다.

• Differences Detection : 기존 분석된 지식에 없던 부분을 모두 식별하고 의미를 부여하는 방식으로 Similarity Detection과 반대의 의미로 볼 수 있다. 결과적으로는 이 부분에 대한 식별 및 심도 깊은 분석이 기존에 없던 새로운 지식을 창출하게 된다.

- Malware Development Detection :

공격자들에게 유리한 점은 신규 제작한 악성코드가 얼마나 방어기술을 회피할 수 있는지 널리 쓰이는 Anti-Virus, Sandbox, On-line Scanning Service 등을 통해 사전에 시험할 수 있다. 따라서, VirusTotal 등과 같은 Online Service에 발생하는 다양한 Query들을 여러 유형의 meta-data와 연계하여 분석하면 공격자들의 의도, 개발방향을 파악할 수 있다. 실제로, Anubis Sandbox를 분석하여 주요 공격에 쓰인 악성코드를 분석한 사례도 있다.

- Malware Attribution

지금까지는 악성코드 자체의 분석에 집중했으나, 정작 중요한 것은 악성코드 공격자를 추적하는 일이다. 이에 대해, 3개 분류가 있다. Technical 분석은 악성코드가 사용한 언어, 악성코드에 내재된 IP나 URL, C&C와의 통신 메시지 등을 의미하며, Operational 분석은 다른 공격사례 분석에서 얻어진 Technical 분석결과와 연계하여 해석하는 것을 의미하며, Strategic 분석은

Threat Intelligence, 정치적 상황 등을 연계하여 공격자 그룹을 찾는 것을 의미한다.

- Malware Triage

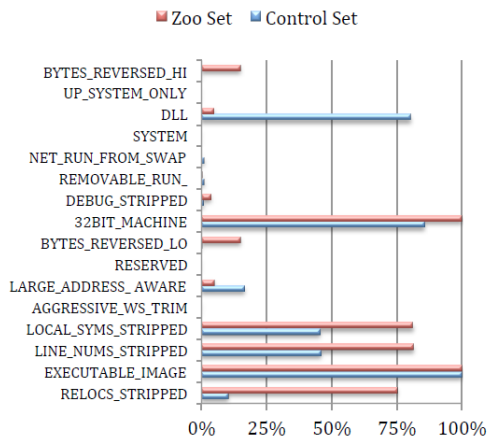
일평균 악성코드 출현량은 160만개를 넘어선 상황에서 악성코드 분석은 신속하고, 우선순위를 고려할 필요가 있다. 즉, 실 환경에서는 대량의 악성코드가 밀려들어오는 상황에서 Key Feature를 가지고 쉽게 처리하는 부분과 심도 있은 모델로 분석해야 하는 모델 등을 복합적으로 운용해야 한다.

Ⅲ. 주요 Features

여기서는 앞서 기술한 악성코드 분석 목적 달성을 위해 사용되는 주요 Feature들에 대해 설명한다. 악성코드 분석에 사용되는 Feature들은 PE header, Strings, Sequences, DLL/API, Entropy, Instructions, Visualization , Memory 정보, File 정보, Registry 정보, CPU Register, Network 접속정보, Anti-Virus 분석정보 등 다양하지만, 이 중 주요하다고 판단되는 주요 Feature 8개를 중심으로 기술하였다.

3.1. PE Header

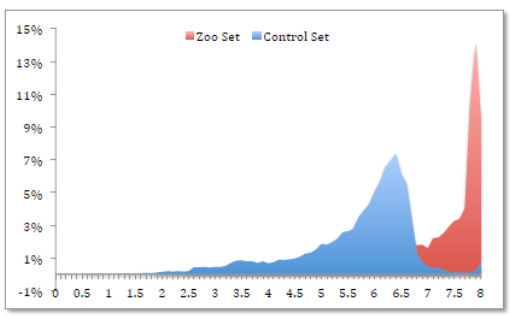
PE Header에는 파일분석에 대단히 유용한 정보들이 많이 포함되어 있다. Section, Import 정보, Compiler, Timestamp 등 많은 정보가 있으며 각각의 Section별로도 많은 정보들을 있으며, 이들은 앞서 기술한 악성코드의 모든 분석 목적에 활용 가능하다. Yonts는 이러한 PE Header 특징을 분석하기 위해서 악성코드와 정상파일 40만개에 포함된 모든 PE Header 정보를 통계적으로 분석하였다[2]. 아래 그림은 PE Header 중, Characteristics에 나타난 Flag 정보들을 토대로 분석한 결과를 나타낸다.



〔그림 2〕 PE Header Characteristics 분석예시

아래 그림은 PE Header 정보 중 Section 분포에 대한 통계적 분석결과를 나타낸다. 일반적으로 정상파일은 알려진 Section 이름으로 3~7개 정도로 많이 분포하나, 악성코드는 임의의 Section명이 사용되고 그 숫자도 상대적으로 많은 경향이 있다. 이러한 특징들을 주요 Feature로 활용할 수 있다.

이외에도 Mahew, Jinrong 등 다양한 연구에서 Resource icon's checksum, Section count, Section name, Section size, Pointer to Symbol table, PE timestamp, PE characteristics 등 많은 Feature들을 사용하여 의미있는 연구결과들을 도출하였다[16,17,18].



〔그림 3〕 PE Header Section 기반 분석 예시

3.2. Printable Strings

Binary 파일에서 Symbol, Strings 기반 연구들도 여러 가지 진행되고 있다. 악성코드의 핵심정보들은 Binary파일에서 특정의 String들로 추출이 될 수 있고, 이 정보가 결국 통계에 반영된다는 가정이며, 이는 Packing된 파일에서도 유사하게 적용 가능하다[5,12]. Binary 파일자체에서 String을 추출하면 일부분의 Code, 저자의 Signature, 파일이나 리소스 이름, 주석 등의 정보를 얻을 수 있는데 이들 중에 악성코드 분석의 Key feature가 포함될 수 있다. 특히, Printable ASCII String 들을 추출하고, 이들에 대해 통계적 분석 기반 Feature 추출, String들에 대한 다양한 유사도 비

〔표 1〕 Symbol 기반 악성코드와 정상파일의 분포분석 예시

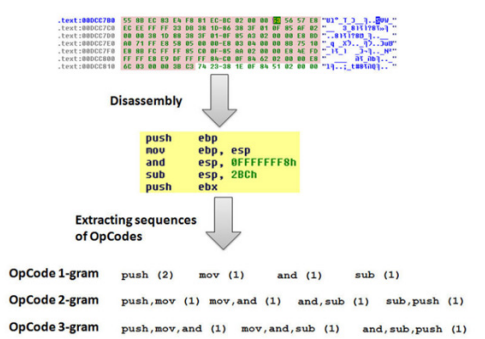
Symbol	Range	Malware Frequency	Benign Frequency
!	0	0.22%	3.20%
	1 ~ 30	10.56%	37.52%
	31 ~ 100	13.54%	20.44%
	101 ~ 800	50.62%	17.92%
	801 ~ 1000	4.64%	2.36%
	1000 ~	20.42%	18.56%
#	0	0.16%	7.52%
	1 ~ 30	6.34%	35.64%
	31 ~ 100	17.06%	19.04%
	101 ~ 800	51.42%	16.64%
	801 ~ 1000	5.14%	2.56%
	1000 ~	19.88%	18.60%
[	0	0.38%	2.28%
	1 ~ 30	5.02%	19.88%
	31 ~ 100	10.82%	21.64%
	101 ~ 800	54.34%	30.56%
	801 ~ 1000	5.18%	3.12%
	1000 ~	24.26%	22.52%

교 연산 등의 기법으로 악성여부 분석, 그룹분류 등의 결과를 산출할 수 있다. 아래 표는 String 분석의 일부분인 특정 Symbol의 분포에 관한 통계분석 결과를 나타낸다. 악성코드는 정상파일 대비 Symbol이 많이 나타나고 있음을 확인할 수 있다.

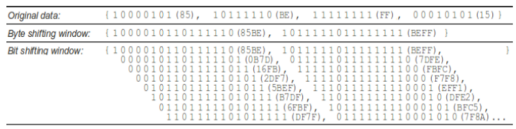
3.3. Sequences

Binary 파일은 결국 byte들에 대한 Sequence로 나타나는데, 이러한 Byte Sequence 분석을 위해 N-gram이 널리 사용된다. N-gram은 하나의 긴 String에 대해 n 크기의 sliding window 단위로 쪼개어 각각에 대한 나름의 방식으로 통계를 산출하고 이를 재료로 다양한 분석결과를 산출할 수 있다.

N-gram은 여기에 쓰이는 Sequence의 분석대상은 무엇이든 이용할 수 있다[19,20]. 일반적으로 Byte단위 기준으로 N-gram 기법이 널리 쓰여 왔으나, 난독화된 악성코드 분석에 효과가 떨어지는 측면이 있기 때문에 최근 Byte 단위가 아닌 Bit 단위의 N-gram 적용 연구가 있다. 아래 그림은 Byte 단위의 N-gram 산출방식과 Bit 단위의 N-gram 산출방식의 비교결과를 나타낸다.



〔그림 4〕 N-gram 기본개념



〔그림 5〕 Byte vs Bit 단위 Sequence 분석예시

3.4. DLL/API

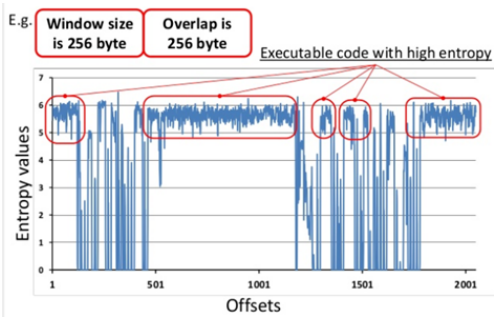
PE 파일의 헤더를 분석하면 Import 되어있는 DLL/API 정보를 확인할 수 있다. 이 정보들은 PE 파일이 실행 시 호출할 API들의 정보이므로 악성코드 분석에 대단히 중요한 정보를 제공한다. DLL Injection, Anti-Debugging 등 기본적인 악성행위를 구성하는 API 호출 목록을 특정할 수 있을 정도로 유용하다. 그러나, 난독화 기술이 적용된 파일들은 추출할 수 있는 DLL/API 들이 한정되어 있으므로 다양한 Feature들을 연계한 분석이 필요하다[21,22].

3.5. Entropy

Entropy는 무질서 정도를 나타내며, 일반적으로 text

〔표 2〕 주요 악성행위 API 분석결과

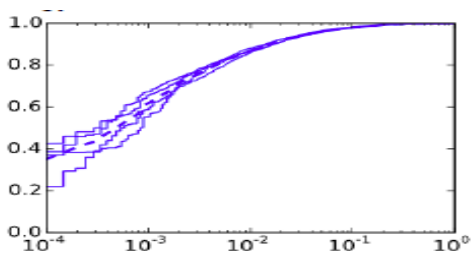
Malware Activity	API Patterns
Key Logger	FindWindowA, ShowWindow, GetAsyncKeyState, SetWindowsHookEx, RegisterHotKey, GetMessage, UnhookWindowsHookEx
Screen Capture	GetDC, GetWindowDC, CreateCompatibleDC, CreateCompatibleBitmap, SelectObject, BitBlt, WriteFile
Anti-debugging	IsDebuggerPresent, CheckRemoteDebuggerPresent, OutputDebugStringA, OutputDebugStringW
Downloader	URLDownloadToFile, WinExec, ShellExecute
DLL Injection	OpenProcess, VirtualAllocEx, WriteProcessMemory, CreateRemoteThread
Dropper	FindResource, LoadResource, SizeOfResource



〔그림 6〕 Offsets에 따른 Entropy값 예시

파일은 4이하, Packing된 파일은 4~6, Encryption된 파일은 6~8정도의 값을 가진다. 따라서, Binary 파일을 특정 크기 단위로 Entropy값을 분석하면 파일에 대한 특징을 파악할 수 있다. 아래 그림은 Offsets에 따른 Entropy 값 예시를 나타낸다.

VirusTotal 등에서는 이미 PE파일의 Section별 Entropy 정보를 제공하고 있으며[6], Ahmadi는 10,000byte 단위로 Entropy 값을 산출하고, 이에 대한 Histogram 분석을 통해 203개 feature를 대상으로 악성코드 그룹을 분류하였다[12]. Joshua는 1,024byte 단위로 Entropy값을 산출하되 1,024 byte block의 Hash값을 x축 16개로 매핑, Entropy값을 y축 16개로 매핑하여, 2차원 평면으로 변환 및 Deep Neural Network 기반으로 악성여부를 분석하였다[12,3]. 아래 그림은 Deep Neural Network 기반 Entropy 대상 ROC Curve 분석결과를 나타낸다.



〔그림 7〕 Entropy기반 악성코드 분석결과(ROC curve)

3.6. Sections

Binary 파일은 사전에 정의된 .text, .data, .bss, .rdata, .edata, .idata, .rsrc, .tls, .reloc 등의 Section 명이 있다. 각각의 Section에는 Code나 Data, Resource 등의 정보들이 들어가는데, 파일 규격상 강제화 되어있지 않다. 일반적으로 정상파일은 이러한 Section명의 규격을 따르는 특징이 있지만, 악성코드는 임의로 생성한 Section명을 사용하거나, 난독화 기법을 연계하여 전혀 다른 방식으로 이용한다. 따라서, 알려진 Section명, 알려지지 않은 Section명을 구분하는 관점에서 다양한 Feature들을 추출하여 분석을 진행할 수 있다. 아래 표는 이러한 Section 기반으로 Feature를 추출한 예시를 나타낸다[5].

Section별로 Binary파일의 컨텐츠가 분리되어 있으므로, Section별 Entropy, Instruction, Visualization 등의 Feature를 연계한 다양한 분석도 가능하다.

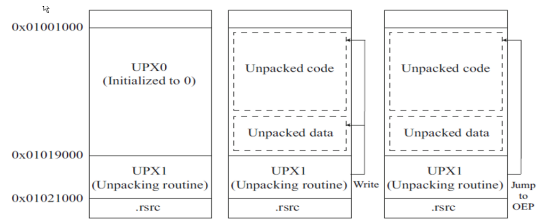
〔표 3〕 Section 기반 Feature 추출 예시

Name	Description
section_names_.bss	The total number of lines in .bss section
section_names_.data	The total number of lines in .data section
section_names_.edata	The total number of lines in .edata section
section_names_.idata	The total number of lines in .idata section
section_names_.rdata	The total number of lines in .rdata section
section_names_.rsrc	The total number of lines in .rsrc section
section_names_.text	The total number of lines in .text section
section_names_.tls	The total number of lines in .tls section
section_names_.reloc	The total number of lines in .reloc section
Num_Sections	The total number of sections
Unknown_Sections	The total number of unknown sections
Unknown_Sections_lines	The total number of lines in unknown sections
known_Sections_por	The proportion of known sections to the all section
Unknown_Sections_por	The proportion of unknown sections to the all sections
Unknown_Sections_lines_por	The proportion of the amount of unknown sections to the whole file
.text_por	The proportion of .text section to the whole file
.data_por	The proportion of .data section to the whole file
.bss_por	The proportion of .bss section to the whole file
.rdata_por	The proportion of .rdata section to the whole file
.edata_por	The proportion of .edata section to the whole file
.idata_por	The proportion of .idata section to the whole file
.rsrc_por	The proportion of .rsrc section to the whole file
.tls_por	The proportion of .tls section to the whole file
.reloc_por	The proportion of .reloc section to the whole file

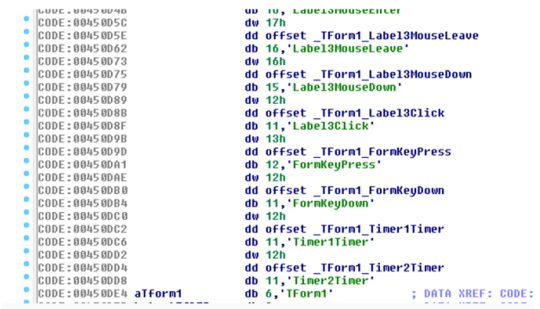
3.7. Instructions

Binary 파일자체에서의 분석은 사람이 코드를 이해하기 어렵기 때문에 특정의 Feature를 중심으로 분석하게 되어 한계가 있다. 이에, Assemble view를 통한 분석은 Semantic에 대한 이해가 가능하며, 좀더 효과적인 분석이 가능하다. 기본적으로 Instruction에 대한 Frequency 등의 통계적 분석을 하거나, Instruction Sequence를 대상으로 N-gram 기반 분석을 할 수 있다. 또한, 특정의 Instruction을 대상으로 다른 의미를 부여할 수도 있다. 예를 들면, Packing 파일은 실행되면 메모리에 Unpakcing된 코드가 기록되고, OEP로 이동하여 실행되게 된다. 아래 그림은 Packing된 파일을 실행했을때의 Unpacking 동작을 나타낸다.

이러한 과정에서 초기 과정은 Unpacking 코드를 메모리에 쓰는 것으로부터 시작된다. 따라서, 이와 관련된



〔그림 8〕 Packing된 파일의 Unpacking 동작 예시



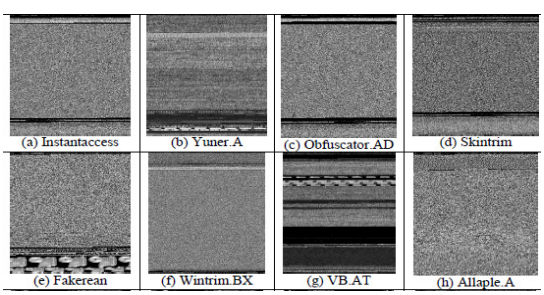
〔그림 9〕 Instruction 기반 Feature 추출 예시

Instruction인 db, dw, dd(bytes, words, double) 등이 실행되는데, 이러한 특징을 Feature로 반영할 수 있다[5]. 아래 그림은 Unpacking에 쓰인 코드들을 나타낸다.

3.8. Visualization

Vision 분야를 평정한 CNN(Convolutional Neural Network) 기반 분석방식을 악성코드에 적용하는 연구들이 진행되고 있다. 악성코드 변종이라면 Binary파일을 시각적으로 표시했을 때 비슷할 Image로 표현 될 것이라는 가정에서 출발한다. 아래 그림은 Natarj에서 분석한 악성코드 그룹별 Image예시를 나타낸다.

CNN기술은 Convolution layer, Polling layer를 통해 최적의 Feature를 선정/추출하지 않아도 되는 장점이 있으므로, Binary 파일 자체를 CNN을 통해 분석한 연구들이 많이 있다. Nataraj는 Binary 파일 각각의 Byte를 Gray-color 0~255값으로 하여 Image로 변환하고, 이렇게 변환된 Image를 K-NN, Euclidean Distance 기법으로 기존에 보유한 DB와 가장 가까운 악성코드, 악성코드 그룹을 분석하거나[4]. Kabanga는 128*128 크기로 변환된 악성코드 Image에 CNN 기법을 연계하여 25개로 구성된 악성코드 그룹분류 결과를 산출하였다.



〔그림 10〕 CNN 기반 악성코드 그룹별 Image 분석 예시

Ⅳ. 결 론

사이버 공격은 매년 큰 폭으로 증가할 뿐 아니라 전기, 가스 및 수도 등 사회 기반시설이 모두 연결되어 가면서 사이버 상의 피해를 넘어서 우리의 삶 전반에 큰 위협이 되고 있다. 이러한 사이버 침해공격의 대부분은 악성코드를 사용하고 있으며, 점차 지능화된 형태로 발전하고 있다. 이에 대응하고자 다양한 악성코드 분석기술이 출현해왔으며, 최근의 연구들은 대부분 머신러닝을 이용하여 기존에 진행했던 Pattern, Heuristic 기반의 한계들을 보완하려 노력하고 있다. 본 논문에서는 머신러닝을 이용한 악성코드 분석기술의 동향을 기술하였다. 특히, 머신러닝을 이용한 악성코드 분석 목적을 7개로 분류하였고, 악성코드 분석에 핵심이 되는 Key Feature들에 대해 소개하였다. 본 논문을 통해, 다양한 악성코드 분석 방법에 있어 새로운 Approach로 연결되는 계기가 되기를 기대한다.

참 고 문 헌

[1] Baset, Mohamad. "MACHINE LEARNING FOR MALWARE DETECTION." (2016).

[2] Yonts, Joel. "Attributes of malicious files." SANS Institute InfoSec Reading Room (2012).

[3] Kabanga, Espoir K., and Chang Hoon Kim. "Malware Images Classification Using Convolutional Neural Network." Journal of Computer and Communications 6.01 (2017): 153.

[4] Nataraj, Lakshmanan, et al. "Malware images: visualization and automatic classification."

Proceedings of the 8th international symposium on visualization for cyber security. ACM, 2011.

[5] Ahmadi, Mansour, et al. "Novel feature extraction, selection and fusion for effective malware family classification." Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy. ACM, 2016.

[6] Jacob, Grégoire, et al. "A static, packer-agnostic filter to detect similar malware samples." International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment. Springer, Berlin, Heidelberg, 2012.

[7] Li, Yuping, et al. "Experimental study of fuzzy hashing in malware clustering analysis." 8th workshop on cyber security experimentation and test (cset 15). Vol. 5. No. 1. 2015.

[8] You, Ilsun, and Kangbin Yim. "Malware obfuscation techniques: A brief survey." Broadband, Wireless Computing, Communication and Applications (BWCCA), 2010 International Conference on. IEEE, 2010.

[9] Liu, Liu, and Baosheng Wang. "Malware classification using gray-scale images and ensemble learning." Systems and Informatics (ICSAI), 2016 3rd International Conference on. IEEE, 2016.

[10] Dahl, George E., et al. "Large-scale malware classification using random projections and neural networks." Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013.

[11] Souri, Alireza, and Rahil Hosseini. "A state-of-the-art survey of malware detection approaches using data mining techniques." Human-centric Computing and Information Sciences 8.1 (2018): 3.

[12] Saxe, Joshua, and Konstantin Berlin. "Deep neural network based malware detection using two dimensional binary program features." Malicious and Unwanted Software (MALWARE), 2015 10th International Conference on. IEEE, 2015.

[13] Madry, Aleksander, et al. "Towards deep learning models resistant to adversarial attacks." arXiv preprint arXiv:1706.06083 (2017).

[14] Lin, Chih-Ta, et al. "Feature Selection and Extraction for Malware Classification." J. Inf. Sci. Eng. 31.3 (2015): 965-992.

[15] Ucci, Daniele, Leonardo Aniello, and Roberto Baldoni. "Survey on the Usage of Machine Learning Techniques for Malware Analysis." arXiv preprint arXiv:1710.08189 (2017).

[16] Ma hew Asquith. 2015. Extremely scalable storage and clustering of malware metadata. Journal of Computer Virology and Hacking Techniques (2015), 1 – 10.

[17] Jinrong Bai, JunfengWang, and Guozhong Zou. 2014. A malware detection scheme based on mining format information. e Scienti c World Journal 2014 (2014

[18] Mansour Ahmadi, Giorgio Giacinto, Dmitry Ulyanov, Stanislav Semenov, and Mikhail Tro mov. 2015. Novel feature extraction, selection and fusion for e ective malware family classi cation. CoRR abs/1511.04317 (2015).

[19] Blake Anderson, Daniel ist, Joshua Neil, Curtis Storlie, and Terran Lane. 2011. Graph-based malware detection using dynamic analysis. Journal in Computer Virology 7, 4 (2011), 247 – 258.

[20] Blake Anderson, Curtis Storlie, and Terran Lane. 2012. Improving malware classi cation: bridging the static/dynamic gap. In Proceedings of the 5th ACM workshop on Security and arti cial intelligence. ACM, 3 – 14.

[21] Ra qul Islam, Ronghua Tian, Lynn M Ba en, and Steve Versteeg. 2013. Classi cation of malware based on integrated static and dynamic features. Journal of Network and Computer Applications 36, 2 (2013), 646 – 656

[22] Ki, Youngjoon, Eunjin Kim, and Huy Kang Kim. "A novel approach to detect malware based on API call sequence analysis."

International Journal of Distributed Sensor Networks 11.6 (2015): 659101.

〈저 자 소 개〉

이 태 진 (Taejin Lee)
정회원

2003년 2월 : POSTECH 컴퓨터공학과 학사
2008년 2월 : 연세대학교 컴퓨터공학과 석사
2017년 2월 : 아주대학교 컴퓨터공학과 박사
2003년~2017년 : KISA 팀장
2017년~현재 : 호서대학교 정보보호학 교수
관심분야 : 시스템보안, 악성코드 분석, 사이버보안