동적 분석과 심층 신경망 모델을 이용한 악성코드 분류 연구

한채연 [O1] 김영재 [1] 허준녕 [1] 명준우 [1]

[1] 국민대학교 컴퓨터공학부

cyhan1004@naver.com, korea_kyj@naver.com, hurjn96@naver.com, mjw95@kookmin.ac.kr

Malware Classification Using Dynamic Analysis and Deep Learning Model

ChaeYeon Han[O1] YoungJae Kim[1] JunNyung Heo[1] JoonWoo Myung[1]

[1]School of Computer Science, Kookmin University

요 약

현재 악성코드 분석에서 많이 사용되는 정적 분석(Static Analysis) 방식은 패킹(Packing) 및 암호화 등의 난독화기술이 적용되었을 때 분류 정확도가 크게 낮아진다. 본 연구에서는 악성코드를 실행시켜 행위를 분석하는 동적 분석(Dynamic Analysis) 기법에 의해서 얻어진 데이터를 심층 신경망의 학습 데이터로 활용함으로써 정적 분석 기법의 한계가 극복될 수 있음을 보인다. 특히, 피처 해싱(feature hashing) 기법을 이용한 전처리 과정을 결합시켜서 악성코드 다중 분류 모델을 학습시킴으로써 정적 분석 방식으로만 학습시킨 분류 모델의 약점을 보완한다. 실제 악성코드 데이터로 실험한 결과, 패킹이 적용된 상황에서 2배 이상 정확도가 향상된 것을 확인할 수 있었다.

1. 서 론

AV-TEST의 집계에 따르면 해마다 발견되는 악성코드의 숫자가 크게 증가하고 있으며, 매일 대략 100만 개 정도의 악성코드가 발견되고 있다[1]. 기존 탐지 솔루션은 지능화되는 악성코드에 대해 효과적으로 대응하지 못하고 있다[2]. 지능화된 악성코드는 실행 압축, 난독화, Anti-VM 등 분석을 회피하는 기술과도 결합되고 있다[2].

악성코드 분석 기법은 크게 정적 분석과 동적 분석으로 나뉜다. 정적 분석은 빠르게 실행되는 장점이 있지만 패킹이 적용된 상황에서는 분석 정확도가 현저히 낮아진다. 동적 분석은 악성코드를 실행시킨 후 악성코드의 행위를 분석하는 방법이다. 분리된 가상 머신 환경에서 해당 파일을 실행한 후 어떤 행위가 일어나는지 관찰하여 구체적인 정보를 획득할 수 있다. API 콜 시퀀스는 기존의 연구에서 동적 분석 행위 지표 중 대표적으로 사용되며, 정적 분석의 한계를 극복할 수 있다.

본 연구에서는 동적 분석을 적용하여 악성코드 분류의 척도가 될 수 있는 API 콜 시퀀스와 API 카테고리 시퀀스를 추출하여 심층 신경망 모델의 학습 데이터로 사용한다. 심층 신경망 모델에 적용하기 전 동적 분석에서 추출한 피처를 바탕으로 빈도수에 기반한 피처 벡터를 만드는 전처리 과정을 도입한다. 이를 통하여 정적 분석 결과만을 사용하는 모델에서 패킹된 악성코드 분류를 잘 하지 못하는 한계점을 극복할 수 있다는 것을 보인다.

본 논문의 구성은 다음과 같다. 2장에서는 기존 악성코드 탐지 및 분석 관련 연구에 대해 소개한다. 3장에서는 본 논문에서 사용한 전체적인 모델을 설명 한다. 동적 분석을 정의하고 추출된 결과 데이터를 심층 신경망 학습 데이터로 사용하기 위한 가공 과정 및 심층 신경망 모델을 소개한다. 4장에서는 제안하는 기법에 대해 실험 결과를 보인다. 5장에서는 결론을 맺고 향후 연구에 관하여 언급한다.

2. 관련 연구

악성코드 탐지 기법 중 시그니처 기반 탐지 기법은 상용화된 안티바이러스에서 가장 널리 쓰이는 기법 중 하나이다. 하지만 상당수의 악성코드는 암호화 기법이나 은닉 기술이 적용되는데, 이 경우 변종 악성코드를 탐지하는 데에 한계가 발생한다.

악성코드 분석 기법 중 정적 분석은 파일을 실행 하지 않고 악성 코드 파일을 분석하는 방법이다. 정적 분석을 통해 얻을 수 있는 정보로는 제어 흐름 그래프, 디스어셈블 된 명령어 시퀀스 등이 있다. 배가영 외 3인은 opcode의 빈도수를 이용한 악성코드 탐지 기법에 기계 학습을 적용한 방법을 제안하였다[3]. 산토스 외 3인은 정적 분석에서 opcode 시퀀스 출현 빈도를 기반으로 새로운 악성코드를 탐지할 수 있는 방법을 제안하였다[5]. 정적 분석을 기반으로 악성코드를 탐지하게 될 경우 처리가 쉽고 빠르다는 장점이 있지만, 패킹이나 난독화가 되어있을 때 탐지하기 어려운 문제점이 발생하게 된다.

고동우 외 1인은 동적 분석 결과인 API 콜 시퀀스에 LSH(Locality Sensitive Hashing) 기법을 적용하여 각 분석 대상들 간의 유사도를 계산하고 유사한 유형끼리 클러스터링을 형성하는 방법을 제안 하였다[6]. 권일택 외 1인은 API 콜 시퀀스를 추출하여 순환 신경망 학습의 과정을 통해 악성코드 패밀리의 API 호출 패턴을 추출하는 방법을 제안하였다[7]. 본 논문에서 제안하는 방식은 동적 분석에서 나오는 정보 중 API 콜 시퀀스와, API의 카테고리 시퀀스를 이용하고, 나아가 이를 딥 러닝 모델에 적용하기 전 가공하는 작업을 포함한다.

3. 동적 분석 기반 딥 러닝 모델

본 연구에서 사용한 악성코드 분석 방법, 데이터 가공 방법 및 심층 신경망 모델은 다음과 같다.

3.1. 동적 분석

본 논문에서는 악성코드 탐지 기법 중 파일을 직접 실행시켜 행위를 분석하는 기법인 동적 분석을 사용한다. 분석 도구로는 오픈소스인 Cuckoo Sandbox를 사용하였다. Cuckoo Sandbox를 통해 나오는 분석 결과는 JSON 형태로 저장되며, 분석 결과 정보는 Process memory, Network, Static, Behavior 등이 있다[4]. 악성 행위를 나타내는 Behavior 중 API 콜 시퀀스를 사용한다. 수많은 API 중 Cuckoo Sandbox에서는 323개의 API 함수 호출을 기록한다. 또한, 2017년 기준으로 Cuckoo Sandbox는 API 함수들을 17가지의 카테고리로 분류하였고[6], 현재 실험 결과 18가지의 카테고리로 분류되어 있다.

3.2. 데이터 가공

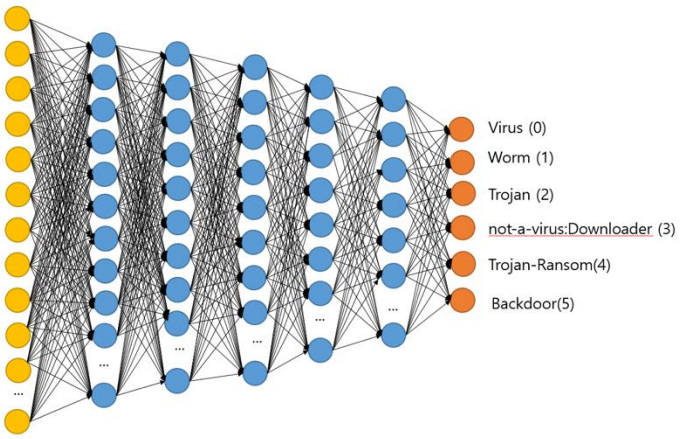
동적 분석을 바탕으로 추출된 정보들을 딥 러닝 모델에 적용하기 전 전처리 과정을 거친다. API 콜 시퀀스와 카테고리 시퀀스는 각각 3, 4, 5-gram 기법을 거친 후, 피처 해싱 기법을 적용한다. 본 논문에서는 암호학적 해시 함수를 사용하여 인덱스를 구하고 피처 벡터를 생성한다.

기존의 피처 해싱 방법은 특정 인덱스에서 많은 충돌 발생 시 값이 비정상적으로 커져서 딥 러닝 모델의 정확도를 떨어뜨릴 수 있다. 본 연구는 [그림 1]과 같이 n-gram 기법을 거친 API 콜 시퀀스 셋에 암호학적 해시 함수를 적용하고, 이를 통해 나온 256비트의 값을 피처 벡터의 최대 크기인 4,096으로 나눈 나머지로 인덱스를 구한다. 그리고 인덱스에 해당하는 배열 값을 1을 증가시킴으로써 서로 다른 두 가지의 암호학적 해시 함수를 사용하는 결과와 동일하게 작용하도록 한다. 마지막으로 피처 벡터의 특정 원소가 너무 작아지거나 커지는 현상을 방지하기 위하여 정규화 과정을 거친다. 피처 벡터의 max 값과 min 값을 추출한 후 두 값이 같지 않은 경우 아래와 같은 수식을 거쳐 0과 1 사이의 값으로 정규화를 시킨다.

$$\frac{x - \min(vector)}{\max(vector) - \min(vector)} \quad (if\ \ min! = max, for\ \ x\ \ in\ \ v)$$

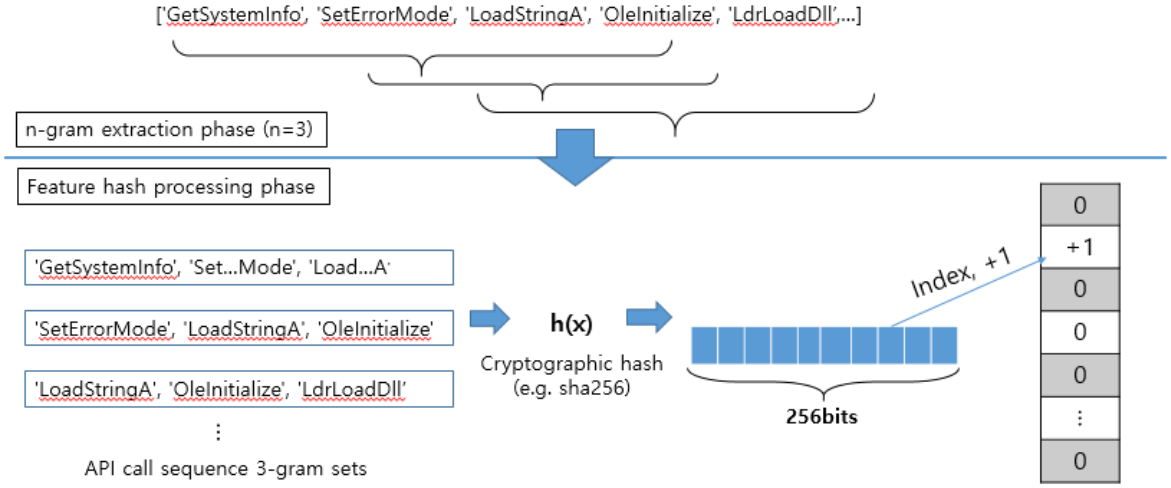
3.3. 딥 러닝 모델

본 논문에서 사용한 악성코드 탐지를 위한 딥 러닝 모델의 구조는 [그림 2]와 같다.



[그림 2] 딥 러닝 모델

심층 신경망의 입력층 노드 개수는 3-gram, 4-gram, 5-gram 기반으로 생성된 4,096 크기를 가지는 피처 벡터를 세 개 연속 연결하여 구성한다(총 12,288개)[8]. 피처 해싱 기법



[그림 1] API 콜 시퀀스에 대한 3-gram과 피처 벡터 생성 과정

을 이용해 가공한 피처 벡터(feature vector)를 입력으로 사용한다. 심층 신경망의 은닉층 개수는 총 5개이며 각각 4,096, 1,024, 256, 64, 16개의 노드를 가지고, 활성화 함수로 ReLU를 사용하였다. 또한 오버 피팅을 방지하기 위해서 드랍 아웃(dropout)을 사용하였다[9]. 이 모델은 30%의 정보를 잊도록 하였다. 소프트맥스(Softmax) 함수를 이용해서 마지막 은닉층을 거쳐 출력층에 도달할 때 어떤 유형의 악성코드에 속할 지의 확률로 변환한 뒤, 가장 높은 확률을 선택하여 파일을 분류하였다[9]. 본 논문에서는 총 6개의 그룹을 정의하였다.

4. 실험

실험에 사용된 데이터 셋은 다음과 같다. 학습에 사용된 악성코드 데이터는 유료 구매 사이트에서 입수한 2017년 8~9월 악성코드 42,000개 이다. 테스트에 사용한 데이터는 2017년 10월 1일~7일 데이터 중 446개를 선정했다. 정적 분석 결과로는 opcode sequence를 사용했으며, 동적 분석을 통해 나오는 결과는 API 콜 시퀀스와 API 카테고리 시퀀스를 실험에 사용했다. 세 가지 경우에 대해 피처를 가공한 후, 딥 러닝 모델에 적용하여 실험을 진행했다.

분류 라벨은 IT 보안 테스트 및 컨설팅 서비스 제공 업체인 AV-TEST에서 우수한 평가를 받는 안티바이러스인 카스퍼스키의 라벨을 기반으로 하여 자체적으로 6 종류의 라벨을 정의하여 사용했다(Virus, Worm, Trojan, not-a-virus:Downloader, Trojan, Trojan-Ransom, Backdoor).

IDA Pro를 이용해 정적 분석 결과인 opcode 시퀀스를 추출하였고. 테스트 데이터 42,000개 중 유의미한 결과가 얻어진 34,350개를 학습 데이터로 사용했다. Cuckoo Sandbox를 이용해 동적 분석 결과인 API 콜 시퀀스를 추출하여 42,000개 중 유의미한 결과를 갖는 40,827개를 학습하였다. 5개 이상의 시퀀스가 얻어지는 경우를 유의미한 결과의 조건으로 사용했다.

다음 실험은 테스트 데이터 셋을 UPX를 이용해 패킹을 하여 진행하였다. 패킹하지 않은 데이터 셋에 대해서는 정확도 차이가 적지만, 패킹한 테스트 셋에 대해서는 정적 분석과 동적 분석이 현저한 차이를 보인다.

실험 결과는 [표 1,2]과 같이 요약된다. 정적 분석의 경우, 패킹한 데이터 셋에 대해서는 정확도가 82.7%에서 35.43%로 확연히 낮아지지만, 동적 분석의 경우는 70%대로 양호하게 분류 정확도가 유지됨을 확인할 수 있다.

[표 1] 패킹 하지 않은 테스트 셋을 이용한 실험 결과

피처(Feature)		정확도
정적	opcode 시퀀스	82.70%
동적	API 콜 시퀀스	80.51%
	API 카테고리 시퀀스	78.67%

[표 2] 패킹한 테스트 셋을 이용한 실험 결과

피처(Feature)		정확도
정적	opcode 시퀀스	35.43%
동적	API 콜 시퀀스	70.59%
	API 카테고리 시퀀스	70.81%

5. 결론

본 논문에서는 동적 분석을 이용하여 악성코드의 행위 지표가 되는 피처를 추출하고, 이를 가공하여 딥 러닝 모델에 적용함으로써 악성코드를 분류하는 기법을 제안하였다. 실제 악성코드를 수집해서 패킹을 적용하여 실험을 진행한 결과, 정적 분석만 사용할 경우 80%대에서 30%대로 정확도가 낮아졌지만 동적 분석의 경우는 70%대로 비교적 높게 정확도를 유지할 수 있음을 확인할 수 있었다. 즉, 패킹이나 난독화 등 지능화된 악성코드를 정적 분석이 잡지 못한다는 한계점을 동적 분석을 이용하여 극복할 수 있고, 신종 및 변종 악성코드에 대해 분류해낼 수 있음을 보였다. 향후 연구에서는 동적 분석 정보 중 다양한 피처를 추출하여 실험을 확장해 볼 계획이다. 또한, 다양한 방법으로 피처를 가공해 봄으로써 악성코드 분류 및 탐지 정확도를 높일 계획이다.

ACKNOWLEDGEMENT

본 연구는 과학기술정보통신부 및 정보통신기술진행센터의 SW중심대학지원사업의 연구결과로 수해되었음(2016-0-00021). This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2016R1A2B4009083).

참 고 문 헌

[1] https://www.avtest.org/en/statistics/malware/

[2] 조호묵, 윤관식, 최상용, 김용민, “지능형 악성코드 분석을 위한 리얼머신 기반의 바이너리 자동실행 환경”, 정보과학회 컴퓨팅의 실제 논문지 제22권 제3호, pp. 139-144, 03. 2016.

[3] 배가영, 우상욱, 김동희, 정태명, “Opcode 를 사용한 악성코드 탐지 기법에 기계학습 알고리즘 적용”, 한국통신학회 학술대회논문집, 1327-1328, 06. 2016

[4] https://cuckoosandbox.org/

[5] I. Santos, F. Brezo, X. Ugarte-Pedrero, and P. G. Bringas. “Opcode sequences as representation of executables for data-mining-based unknown malware detection.” Information Sciences, 231, pp.64-82, 2013.

[6] 고동우, 김휘강, “API 콜 시퀀스와 Locality Sensitive Hashing을 이용한 악성코드 클러스터링 기법에 관한 연구”, 정보보호학회 논문지 27(1), 91-101, 02. 2017.

[7] 권일택, 임을규, “악성코드 API 호출 패턴 추출에 대한 순환 신경망 적용 연구”, 한국정보과학회 학술발표논문집, 1081-1083(3 pages), 06. 2017.

[8] 정성민, 이식, 정지만, 윤명근, “심층학습과 악성코드 분석 연구”, 정보과학회지 제36권 제2호 37-42, 02. 2018.

[9] 정지만, “4차 산업혁명을 대비한 딥러닝 기술의 금융보안 적용 연구”, 학위논문, 국민대학교, 2017