딥러닝 기반 파일리스 악성코드 탐지 기법의 연구

채승언, 김봉현, 이차규, 최선오
호남대학교 컴퓨터공학과
e-mail : eirinlove@gmail.com
kibohy4853@gmail.com
jnvr200409@naver.com
suno@honam.ac.kr

A Study on Detecting Fileless Malware Using Deep Learning

Seung-Un Chae, Bong-Hyun Kim, Cha-Gyu Lee, Sunoh Choi
Dept. of Computer Engineering, Ho-Nam University

요 약

기존 악성코드 탐지 방법의 한계점과 심층 학습기술의 적용을 통한 악성코드의 탐지 및 분류방법을 기술하고 탐지에서의 각 학습모델에 대한 테스트 성능 과정 정확도를 비교하여 파일리스 악성코드 탐지에서의 심층 학습기술의 유용성과 발전 가능성을 판단하려 한다.

1. 서론

기존 PE (윈도우 실행 파일 [dll, exe] 등 사용자가 직접 파일을 실행하거나 추출하여 시스템에 침투하였던 악성코드와는 달리 근래에는 공격자가 Fileless 형태로 PowerShell 스크립트를 사용해 사용자의 단말의 조작권한을 획득할 수 있다.
사용자가 개방한 포트간 통신 혹은 인가한 프로그램(어플리케이션)을 경로로하여 Fileless Malware 코드가 침투하며, 이는 보안 취약점이 발견된 FlashPlayer, Active X, JavaScript 등에서 발견되는 취약점과 유사하다.
모든 백신 소프트웨어는 보고된 것 혹은 파일 내 문자열 패턴 등을 기준으로 검사하며 Fileless 악성코드 공격은 최종적으로 PowerShell 을 통해 사용자가 실행시키게 되는데 PowerShell 은 실행흔적을 거의 남기지 않고, PowerShell 코드를 난독화 하거나 정상적인 스크립트로 위장하여 사용될 경우 판단이 쉽지 않기 때문에 이에 대한 대처가 어렵다.
이에 대해 본 논문에서는 파일리스 악성코드 탐지의 기법에 대해 기술하고 심층학습을 통한 악성코드 탐지와 가용성 있는 모델을 나열하여 분석해보고자 한다.

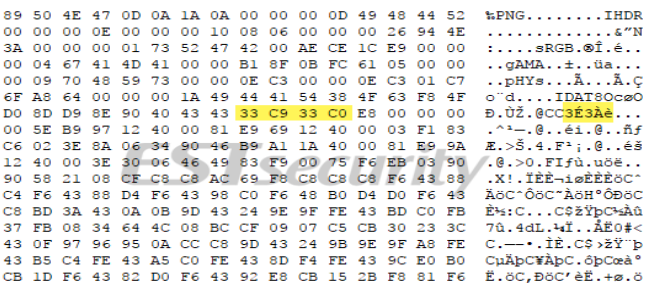
2. 본론

2.1 PE (Portable Executable)

PE 포맷은 운영체제에서 사용할 수 있는 DLL, Object 코드, EXE, SYS 등의 운영체제 Loader 가 실행 가능한 코드를 실행하는 실행파일을 말하며, PE 포맷을 통한 공격으로 File Header 혹은 Optional Header 등을 속이거나 PE 의 실행을 통해 파일의 연결 주소를 알아내어 사용자의 단말에 악성코드를 삽입해 시스템에 이상을 일으키는 방법이 있다.

2.2 파일리스 악성코드(Fileless Malware)

파일리스 악성코드는 PE 를 이용한 악성코드 삽입공격과 달리 사용자의 디스크 내에 악성코드 파일을 설치하지 않고 바로 메모리 내에 적재시킨다.
파일리스 악성코드는 PowerShell 과 WMI 등에 접근하여 실행되는데 악성코드는 커널 메모리 취약점, 악성 스크립트 실행, WMI 바인딩 등을 통해 사용자의 시스템 프로세스에 코드가 삽입될 수 있다.
근래에는 기존 PE 공격에 대한 대처가 분명해지면서 파일리스 악성코드 형태의 공격이 늘어나고 있다.



(그림 0) PNG 포맷 내 존재하는 ShellCode
출처 - 이스트 소프트

2.3 악성코드 탐지방법

악성코드의 탐지방법으로서 백신 프로그램에서 사용되는 대표적인 탐지 방법과 심층 학습에서 사용되는 방법 두 가지로 분류한다.

① 시그니처 기반 탐지

수집된 악성코드의 특징 및 패턴을 분석하여 악성코드를 탐지하는 시그니처를 생성하며 시그니처 데이터 베이스에 등록하여 입력 패턴에 대해 대조 및 검출한다.

② 시퀀스 기반 탐지

시그니처 기반탐지가 행위기반에서 비롯하여 시스템에 가하는 영향을 탐지하는 기법이라면 시퀀스 기반 탐지는 파일에서 연결되는 단어나 문법, 길이 등의 연속적인 데이터의 연관 관계를 설명하며, 딥러닝 기반의 악성코드 탐지에서 정상 시퀀스를 가진 파일의 데이터와 악성 시퀀스를 가진 파일의 차를 구해 악성코드의 유무를 탐지할 수 있다.

2.4 심층망에서의 악성코드 분류

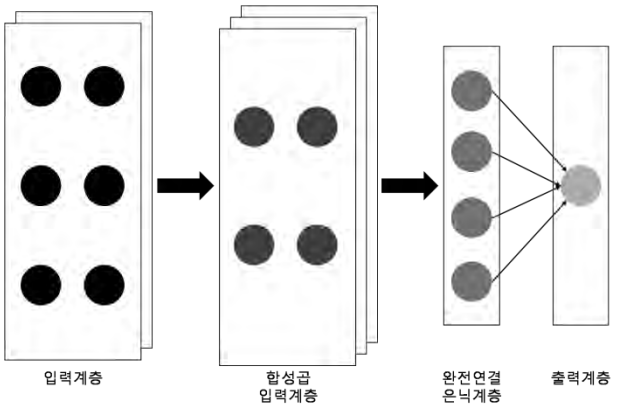
심층망에서의 악성코드 분류를 위해 쉘로부터 추출된 4000여개의 정상 파일과 악성파일의 길이 그리고 둘의 시퀀스 데이터를 비교하는 방법을 적용하였다. 이 때, 지도학습에 있어 추출할 테스트 데이터를 만들기 위해 데이터 배열 4000에 관해 테스트 데이터를 0.1로 설정하였다.

	악성여부	길이	시퀀스			
normal1001	0	30	4062	2025	8002	2026
normal1002	0	8000	8013	8002	2035	8000
normal1003	0	8002	2052	33	4080	8005
normal1005	0	8016	8016	8016	8016	8016
normal1006	0	8000	8005	8006	8010	8002
normal1007	0	21	2147	8002	2148	8000
normal1008	0	8002	2150	8000	8005	90
normal1009	0	48	4631	2153	4632	2084

(그림 1) Noraml_data_export.csv

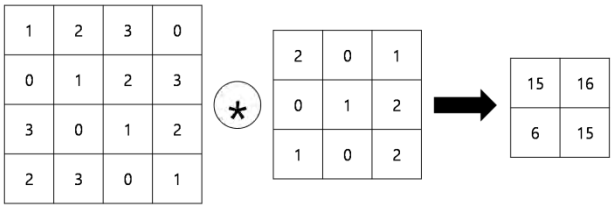
2.5 Convolution Neural Network (CNN)

CNN에서는 합성곱 필터를 이용하여 특징점을 추출한다. Dense 수에 해당하는 필터는 추출된 특징을 통해 차원을 축소하여 완전연결계층으로 처리결과를 통신하게 된다.



(그림 2) CNN 모델의 레이어 구성

ConvLayer 와 OutputLayer 사이에 속하는 PoolLayer 는 추출한 이미지의 특징을 모으고 강화하게 되며 활성화 함수인 ReLU는 경사값의 조정을 하게 된다.

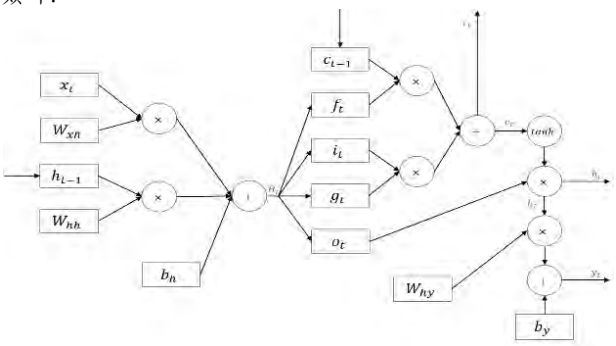


(그림 3) 합성곱 필터의 동작

합성곱 필터에서는 입력 데이터에 필터를 곱하여 특징을 추출하게 되고 입력 패딩은 합성 곱 연산을 수행하기 전 입력 데이터 주변을 0과 1로 채워 출력의 크기를 조절하여 합성 곱 연산을 지속하여 수행할 수 있도록 한다.

2.6 Long Short-Term Memory (LSTM)

RNN 에서는 신호를 순환해 시계열 신호와 같이 상호 관계가 있는 신호를 처리하게 된다. 이 때 출력된 신호에 대해 순환 횟수에 따라 활성화 함수가 중첩 적용되어 경사가 사라질 수 있는데 LSTM에서는 RNN의 이런 의존기간에 관련한 문제를 해결하기 위해 Call state에 Gate 구조를 부가하여 컴포넌트의 정보 전달량을 조절할 수 있다.



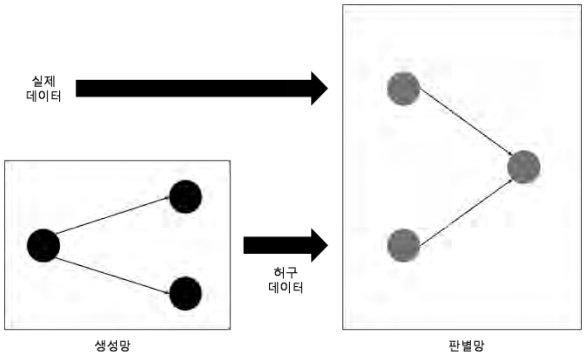
(그림 4) LSTM 모델에서의 순전파

LSTM 모델의 순전파 과정은 H_t를 n등분 하고 각각에 해당하는 활성함수를 적용하게 된다. 이 때의 역전파를 구할 때 각각의 Gradient를 병합하게 되면 dH_t가 되는데 이는 다시 역전파로 활용된다.

2.7 Generative Adversarial Networks(GANs)

GAN은 CNN과 LSTM과 달리 분류에서, 정답 레이블에 의존하지 않는 비지도형 학습방법이다.

생성자(Generator)와 판별자(Discriminator) 레이어로 나뉘게 되며, 생성자가 받은 훈련 데이터를 통해 가상의 데이터를 생성하게 되고 판별자는 앞서 말한 방법을 사용하여 생성자의 가상 데이터를 실제데이터의 판별 근거로 사용하게 된다.



(그림 5) GAN 모델의 동작원리

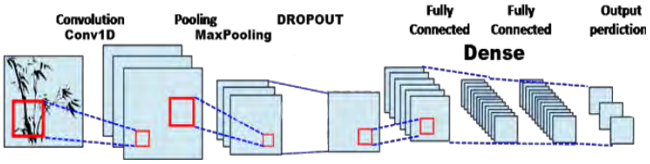
이와 같이 GAN 모델은 생성자가 가상의 데이터를 생성한다는 점에서, Malware Data 에 Noise 등을 부가하여 역으로 판별자가 판별하지 못하는 악성코드를 생산해내는 MalGAN 구조가 제안된 바가 있다.[1]

3. 실험 결과

4000 여 개의 훈련 데이터를 바탕으로 400 여 개의 테스트 데이터를 대입한 결과로 각 심층 학습 모델을 구현 및 적용하면서 도출되는 결과와 방법을 기술하려고 한다.

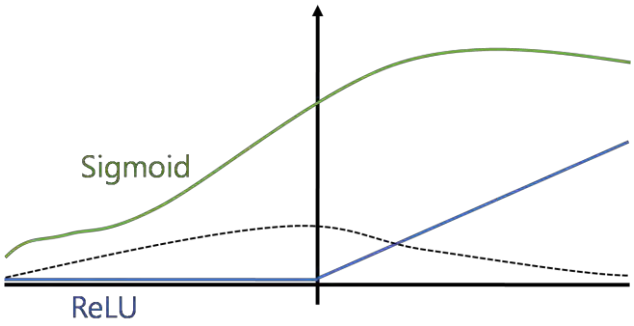
3.1 CNN 에서의 악성코드 탐지

악성코드의 탐지에 CNN 모델을 적용하는 경우 선형 데이터를 매핑 시키기 위해 벡터화(Embeding) 하여 표현하고 지역적 특징 추출을 위해 합성곱 레이어, 그리고 특성에서 가장 큰 벡터 정보를 반환하기 위해 MaxPooling 을 적용하였다. 다차원 차원공간에서의 간소화에 Dropout 그리고 최종적으로 추출된 레이어를 전 연결 시키는 것이 CNN 의 과정이다.



(그림 6)정의된 CNN 의 계층구조

또한 합성곱 레이어 적용시, 경사 소실문제를 완화하기 위해 활성화 함수를 ReLU 로 설정한다. 여기서 ReLU 는 $\max(0,x)$ 즉, 0 미만의 값에 0, 0 초과의 값에 1 을 반환시켜 경사를 조정하기 때문에 경사 소실문제를 극복한다. Sigmoid 는 0 보다 큰 값의 경우 무조건 1 을 반환하므로 DROPOUT 계층을 거친 후 전 연결 계층과 출력 계층에서 Sigmoid 활성화 함수를 사용하여 더 정확도가 높아지도록 유도할 수 있었다.



(그림 7)

시그모이드 함수의 경우 0 보다 큰 값에 대하여 경사가 상승한다.

여기서 CNN 모델을 적용한 탐지결과는 다음과 같다.

```
TP= 201.0
FP= 7.0
FN= 1.0
TN= 191.0
precision= 0.9663461538461539
recall(detection rate)= 0.995049504950495
false positive rate= 0.03535353535353535
f1_score= 0.9804878048780488
```

(그림 8)CNN 적용 결과

4000 여개의 테스트 데이터를 사용한 결과로 정확도와 재현율(탐지율)을 구하게 될 경우 다음과 같다.

Precision = TP / (TP+FP) = 201/208 = 0.9663

Recall = TP / (TP+FN) = 0.9950..

정확도의 수치가 탐지율에 비해 상대적으로 낮게 나온 것으로 확인되었다.

3.2 LSTM 에서의 악성코드 탐지

LSTM 레이어를 악성코드 탐지에 적용하는 경우 LSTM Output 의 크기를 Cell 과 같이 설정하여야 한다. 이 때 각 크기를 동치 시켜 셀에 적당량의 정보가 들어갈 수 있도록 한다는 점은 RNN 과 비슷하다.

```
TP= 197.0
FP= 6.0
FN= 5.0
TN= 192.0
precision= 0.9704433497536946
recall(detection rate)= 0.9752475247524752
false positive rate= 0.030303030303030304
f1_score= 0.9728395061728395
```

(그림 9)LSTM 적용 결과

Precision= 0.9704

Recall = 0.9752

이는 정확도의 수치가 탐지율과 비슷함을 보여주지만 전반적으로 조화평균(f1_score)의 값이 CNN 모델에 비해 떨어짐을 알 수 있다.

3.3 CNN + LSTM

CNN 모델의 CONV 계층과 LSTM 모델의 LSTM 계층을 같이 적용하여 탐지율이 오를 수 있는가를 확인하기 위함으로 두 개의 Dense 계층은 각각 활성화 함수 Sigmoid 와 ReLU 두 가지를 사용한다.

```
TP= 201.0
FP= 8.0
FN= 1.0
TN= 190.0
precision= 0.9617224880382775
recall(detection rate)= 0.995049504950495
false positive rate= 0.04040404040404041
f1_score= 0.9781021897810219
```

(그림 10) CNN+LSTM 적용 결과

이는 LSTM 에 비해 높은 탐지율과 조화 평균을 보이고 CNN 과 거의 비슷한 양상을 보인다.

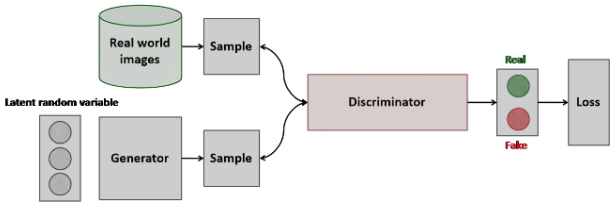
3.4 GAN 에서의 악성코드 공격 가능성

GAN 레이어에서는 생성자와 판별자 모델이 사용되기 때문에 마찬가지 모델도 2 개로 나누어야 한다. 이 때 생성자 모델과 판별자 모델 사이에서의 Loss 함수는 다음과 같은 관계를 갖는다.

$$\min_G \max_D V(D,G) = E_{x\sim p_{data}(x)}[log\ D(x)] + E_{z\sim p_z(z)}[log(1-D(G(z))]$$

이는 가상 데이터의 형성에 있어, 두 번째 항을 최소화 시켜 각 모델이 갖는 구분 값을 0 에 근사하도록 한다.

각 모델에는 경사를 조정하기 위해 LeakyReLU 사용을 통한 경사 조정을 하였으며, 판별자 레이어의 배치 정규화를 통해 가중치의 변동폭을 증가시켰다.



(그림 11)GAN 모델의 적용구조

이제 생성자가 가상의 데이터를 생성하여 판별자가 판별할 경우에 대한 loss 함수의 수치가 서로 반비례하며 변동하게 되는데 이 때 생성자 모델은 사전에 충분한 훈련 데이터가 적재되어야 한다.

```
47 [D loss: 0.606489, acc.: 68.75%] [G loss: 1.282632]
48 [D loss: 0.718132, acc.: 65.62%] [G loss: 1.603984]
49 [D loss: 1.156854, acc.: 42.19%] [G loss: 1.494775]

TP= 189
FP= 4.0
FN= 4.0
TN= 174
precision= 0.9749
recall(detection rate)= 0.9792
false positive rate= 0.0228
```

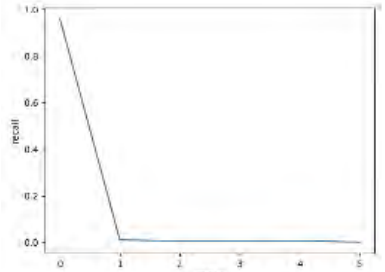
(그림 12)판별자 모델의 학습결과와 모델생성

생성자가 충분한 학습을 거치지 않을 경우 적은 조합의 생성 데이터만을 내놓으므로 과적합 문제를 야기할 수 있다.

```
TP= 2.0
FP= 3.0
FN= 191.0
TN= 179.0
precision= 0.4
recall(detection rate)= 0.010362694300518135
false positive rate= 0.016483516483516484
f1_score= 0.020202020202020204
```

(그림 13)생성된 GAN 모델의 CNN 레이어 적용

GAN의 생성자 모델혹은 무작위로 변조된 시퀀스 등을 기존 레이블 기반 악성코드 탐지에 적용할 경우 충분한 Epoch 에서 정확도가 떨어지거나, 시작 Epoch 에서의 재현률이 0 에 수렴해가는 것을 확인할 수 있다. 이는 악성 시퀀스 데이터를 공격자가 변조하여 검출해내지 못하게 하고 기존 악성코드 검사 방법들을 우회할 수 있는 방법이기도 하다.



(그림 14) 난수 입력 데이터로 변조된 학습모델의 분류

3.5 최적화

각 모델을 적용 및 탐지 테스트를 거치면서 지도형 심층학습 모델의 경우 평균적으로 Epoch 와 테스트 데이터양과 탐지율이 비례하는 편이었으며, 비지도형 학습 모델의 경우 사전 훈련 데이터가 탐지율에 큰 영향을 주었다.

적용모델	Normal	Mal	Test	Epoch	TP	FP	FN	TN	정확도	탐지율	오탐률
CNN	1838	1872	371	10	191	4	2	174	0.97948	0.98963	0.02247
CNN	1838	1872	371	20	191	5	2	173	0.97448	0.98963	0.02808
CNN	1838	1872	371	40	191	5	2	173	0.97448	0.98963	0.0808
LSTM	1838	1872	371	10	191	3	2	175	0.94856	0.98963	0.01685
LSTM	1838	1872	371	20	198	1	4	177	0.99473.	0.97927	0.00561
LSTM	1838	1872	371	40	188	1	5	177	0.99470	.097400	0.00561
CNN+LSTM	1838	1872	371	10	189	4	4	174	0.97927	.097927	0.02247
CNN+LSTM	1838	1872	371	20	189	5	4	173	0.97422	0.97927	0.02808
CNN+LSTM	1838	1872	371	40	190	5	3	173	0.97435	0.98445	0.02808
DCGAN	1838	1872	371	50	189	4	4	174	0.97490	0.97920	0.02280

(그림 15)각 모델 별 테스트 결과값

또한 이는 파일의 시퀀스 데이터의 값에 의해서도 바뀔 수 있었는데, 음수의 시퀀스에 대해 예외처리를 하지 않을 경우 시퀀스를 0 으로 반영해 탐지율이 현저히 떨어지는 현상을 초래할 수 있다.

normal10(	0	8023	8000	8019	-1
normal10(	0	8019	-1	140	-1
normal10(	0	8005	8005	8005	8005
normal10(	0	8023	8021	8009	8028
normal10(	0	286	-1	8019	-1
normal10(	0	8019	-1	8023	8021

(그림 16)

4. 결론

파일리스 악성코드의 탐지에 있어 인공신경망의 구현은 상당히 높은 탐지율을 보여주었고 몇몇 모델에 한해서는 기대치 이상의 결과를 보여주어 악성코드 탐지 및 분류 기술로서의 인공신경망 기술의 유용성을 보일 수 있었으며, 공격자 입장에서 이 기술을 역이용하여 무력화할 수 있는 방법 등에 대한 대처를 고려해야 한다는 점에서 인공신경망 기술의 적용과 개발 필요성 촉구는 더욱 가속화될 전망이다.

참고문헌

[1] Danny Hendler, Detecting Malicious PowerShell Commands using Deep Neural Networks, 2018

[2] Bingcai Chen, Adversarial Examples for CNN-Based Malware Detectors, 2019

[3] WeiWei Hu, Modbus Gernerating Adversarial Malware Examples for Black-Box Attacks Based on GAN, 2017

[4] Lantao Yu, SeqGAN: Sequence Generative Adversarial Nets With Policy Gradient, 2016