최종보고서

머신러닝 최신기술 특론 term project

M2018170김동훈, M2018175이식

2018

1. 프로젝트 동기

Packing 이란? 파일의 크기를 줄이는 목적 또는 리버스엔지니어링으로 부터 파일을 보호하는 것이다. 정상적인 소프트웨어뿐만 아니라 악성코드에도 packing이 적용 되 있는 경우가 있기 때문에 악성코드를 정적 분석을 해야 되는 경우에는 packing을 여부를 판단하는 것이 정적 분석에 도움을 줄 수 있다.

1. 프로젝트 목적

packer들의 signature를 탐지해서 packing 여부를 확인하는 프로그램들은 이전부터 존재는 하지만 signature 기반의 탐지의 한계로 등록되어 있지 않은 packer들이나 signature를 뽑을 수 없는 packer들은 탐지를 하지 못한다.

그렇기 때문에 새로운 packer의 패킹에도 잘 작동하는 packing 탐지 프로그램을 만들기 위해서 딥러닝 알고리즘을 사용하여 특정 파일의 packing여부를 판단하려 한다.

1. 개발 내용
2. 데이터 수집

Opcode 기반으로 딥러닝에 학습을 시키기 위해서 PE파일중에서도 .exe 형식의 파일만을 수집을 하였다. packing 할 것 이기 때문에 패킹된 데이터 약50000개와 패킹이 되지않은 데이터 50000개를 수집을 하였다.

패킹이 되지 않은 데이터 셋을 수집을 하기 위해서 기존의 연구실에서 가지고 있는 악성코드 데이터 중 upxpacker로 수집이 되 있는 악성코드를 unpacking 시켜주었고 이것을 패킹이 안 돼 있는 데이터로써 약50000개를 수집하였다.

패킹이 돼 있는 데이터 셋은 전 단계에서 얻은 패킹이 안되 있는 데이터 셋에서 각종 패커들로 다시 패킹을 하여 50000개의 데이터를 수집하였다.

1. 데이터 가공

DNN를 사용하기 위해서는 고정된 크기(512bit)의 벡터로 변환을 해주어야 한다. 이를 위하여 파일들의 feature를 추출해 주는 작업이 필요하다.

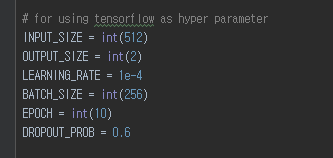
우선 ida pro 라는 disassemble 툴을 이용하여 파일들의 opcode중 operator만 추출하고 특정 크기의 문자열로 저장한다. 이 때 BasicBlock 단위로 opcode들을 추출하였다.

한 파일에서 여러개의 BasicBlock 단위의 opcode들이 추출되는데 각각을 세개의 해쉬함수 (MD5, SHA256, SHA512)를 적용시킨다. 각각의 해쉬의 해당 인덱스로 간 후 그 값이 가장 작은 곳을 찾아 +1을 해준다. 단 +1을 해주기 전에 그 값이 255 이하라면 +1을 해준다. 각 인덱스의 최대값은 256이며 벡터의 크기는 512이다. 모든 BasicBlock 단위의 opcode들에 대해 이러한 행위를 다 했다면 마지막으로 벡터의 각각의 값에 대해 0~1 값 사이를 취하기 위해 256으로 나누어 준다.

1. 학습

TensorFlow(동훈), Keras(식) 두가지를 이용하여 학습을 진행하였다.

TenserFlow에서는 512bit의 크기의 벡터를 입력으로 받아 5개의 층으로 구성된 ANN을 사용한다. 512개의 input을 받아 5개의 히든레이어를 거쳐 2개의 output이 나온다. 파라메터 설정은 LearningRate는 1e-4로 설정하였으며 Epoch 10, BatchSize 256, Dropout 0.6으로 설정하였다.

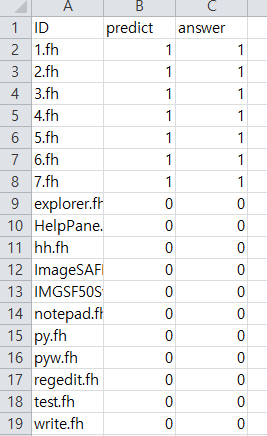


Keras에서는 512bit 크기의 벡터를 입력으로 받아서 3개의 층으로 구성된 DNN를 사용한다.

1. 결과

TensorFlow에서의 정확도 99.98% (약 4만개의 TestDataSet으로 검증)

다른 흥미로운 결과로는 더미다 패커로는 학습을 시키지 않았다. 학습을 시키지 않은 패커에 대해 결과를 확인해 보았더니 모두 패킹을 했다고 탐지를 하였다.



1.fh ~ 7.fh는 더미다로 패킹을 한 파일이며 그 아래는 Windows10에서의 exe파일, VisualStudio로 컴파일하여 생성한 패킹을 하지 않은 exe파일들이다. 모두 정확하게 패킹 유무를 탐지해 내고 있다.

1. 기타

GitHub Commit Graph

