<별첨6>

4차산업혁명을 선도하는 CODE형 SW 인재 양성

Capstone Design(종합설계) 중간보고서

<table>
<tr><td colspan="2">교과목명</td><td colspan="2">소프트웨어캡스톤디자인</td><td>학부(과)명</td><td colspan="2">콘텐츠IT</td></tr>
<tr><td colspan="2">작 품 명</td><td colspan="5">인공지능 기반 Windows 악성코드 탐지 및 PE 파일분석 서비스</td></tr>
<tr><td colspan="2">팀 명</td><td colspan="5">아메바</td></tr>
<tr><td colspan="2">지도교수</td><td colspan="5">곽병일</td></tr>
<tr><td rowspan="7">참여
학생</td><td rowspan="2">대표</td><td>소속학과</td><td>학번</td><td>성명</td><td colspan="2">연락처</td></tr>
<tr><td>콘텐츠IT</td><td>20157148</td><td>우건희</td><td colspan="2">010-5767-9457</td></tr>
<tr><td rowspan="5">팀원</td><td>소속학과</td><td>성명</td><td>소속학과</td><td colspan="2">성명</td></tr>
<tr><td>빅데이터</td><td>김동영</td><td></td><td colspan="2"></td></tr>
<tr><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td></td><td></td><td></td><td colspan="2"></td></tr>
<tr><td colspan="2">참여 분야</td><td colspan="5">√ SW융합대학 □ SW융합/연계 전공</td></tr>
<tr><td colspan="2">참여기업</td><td colspan="5">㈜인반트</td></tr>
</table>

과제 목적

코로나19 Pandemic으로 인해, 일상 전반의 인프라가 디지털화, 비대면화되고 있습니다. 그러나 충분히 준비되지 못한 상황에서 온라인을 활용한 활동이 증가함에 따라, 상대적으로 보안이 취약한 가정용 장비 등에 대한 공격 및 악성코드에 대한 보고가 증가하고 있으나, 증가하는 악성코드를 분석가가 분석하기에는 한계가 있습니다.

기존에 인공지능 기반 악성코드 관련 연구는 많이 이루어져 왔습니다. 그러나, 이러한 상황에서 직접적으로 활용되고 있지 못하고 있습니다. 저희는 이러한 상황에 문제를 느끼고, 분석가를 도움을 줄 수 있는 보조 도구 개발을 결정하게 되었습니다. 그중에서, 비교적 구현이 쉬운 Windows 악성코드를 중심으로 개발할 예정입니다.

이 프로젝트에는 크게, 악성코드를 처리하는 부분과 웹 서비스를 담당하는 부분으로 나뉘게 됩니다. 자체 개발한 인공지능 모델을 통해 악성코드 여부를 확인하고, 분석가들이 분석을 용이하게 할 수 있도록 각종 지표와 값을 제시합니다. 이는 웹 페이지 서버 - 악성코드 분석 서버 - DB로 이어지는 인프라를 기반으로 작동하며, 웹을 통해 언제 어디서든 접근할 수 있습니다.

이를 통해, 악성코드 분석가가 분석할 대상을 명확히 하고, 그들이 원하는 값을 적시에 제공하는 것을 목표

로 합니다. 본 프로젝트를 통해, 악성코드 분석의 효율과 질은 높이고, 분석에 들어가는 시간은 낮추는 효과가 있을 것이라 기대합니다.

과제 추진 내용

- 진행 중 발생하는 기술/제품/서비스 문제점 및 해결 방안
- 과제 예상 결과물 및 구체적인 과제 수행 진행 상황 기술

저희 프로젝트의 개발 구현은, 악성코드 분석 인공지능 부분과 웹 개발 부분으로 나눠집니다.

인공지능 개발 부분의 경우, 각 부분을 Jupyter Notebook으로 개발 및 Test를 진행 중입니다. 개발을 완료한 코드는 다음과 같습니다.

첫번째로, das-malwerk와 vx-underground 두 사이트에서 악성코드를 크롤링하는 코드를 개발했습니다. 구체적으로, urllib를 통해 페이지에 대한 HTML 코드를 불러옵니다. BeautifulSoup를 통해 HTML 코드를 Parsing 하여 악성코드가 있는 링크를 수집하고, 각 링크에 접속하여 다운로드를 진행합니다. 그 결과, 총 38,000여개의 Windows PE 악성코드를 수집할 수 있었습니다.

두 번째로, 다운받은 악성코드의 이름을 SHA-256 해시함수의 해시값 (이미지 값)으로 바꾸는 코드를 개발했습니다. 해시함수는 입력값이 1비트만 달라져도 해시값 (이미지 값)이 달라지므로 무결성을 보장할 수 있게 됩니다. 이런 이유로, 악성코드의 이름을 해시값 (이미지 값)을 이용해 많이 부릅니다. 저희도 악성코드를 원래의 이름이 아닌, 해시값으로 바꿔 파일 이름으로 바꾸는 코드를 개발하게 되었습니다. 구체적으로, 해당 코드는 open()을 바이너리 모드로 불러와서, 해당 값을 hashlib의 sha256()의 hexdigest()를 이용해 해시값을 추출한 뒤, os의 rename을 통해 새로운 파일 이름으로 바꾸게 됩니다.

세 번째로, 저희는 다운받은 악성코드 중 PE 파일만 골라내는 코드를 개발했습니다. 구체적으로, 파일을 open()을 통해 연 뒤, 파일의 맨 앞 2바이트만 불러와서 'MZ' 값이 있는지 판단하고, 없을 경우 os 라이브러리의 remove()를 통해 해당 파일을 제거합니다. PE 구조에는 맨 앞 2바이트가 'MZ'라는 Magic 값이 있는데, PE 파일 포맷이라는 것을 알려주는 값이며, 없을 경우 해당 파일은 실행할 수 없게 되기 때문에 해당 값으로 PE 파일 여부를 판단하였습니다.

네 번째로, PE 값을 추출하는 코드를 개발했습니다. PE 값은 실행 압축파일로써, 프로그램이 실행하기 위해 필요한 값들이 저장되어 있습니다. 그중에서 악성코드 분석에 많이 쓰이는 값 (NT Header 중에서 File Headers 내 일부 값, NT Header 중에서 Optional Headers 내 일부 값, 각 Section Header 내 일부 값, Import API 목록, Export API 목록)들을 중심으로, pefile 라이브러리를 이용하여 추출하게 되었습니다.

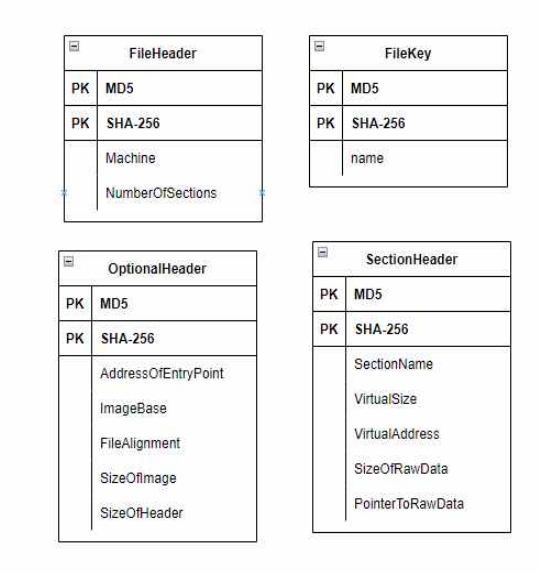
다섯 번째로, CNN 분석을 위해 다운받은 악성코드 전체 데이터를 이미지로 변환하는 코드를 개발했습니다. 구체적으로, 악성코드 파일의 데이터를 불러들인 뒤, 이미지 구현을 위한 길이 (파일 길이의 제곱근)을 구하고, array 라이브러리의 배열에 파일의 내용을 저장한 뒤, 이 배열의 값을 imageio 라이브러리를 통해 이미지에 데이터를 입력하고, Image의 PIL 라이브러리를 통해 입력된 값을 이미지 파일로 만들어줍니다.

여섯 번째이자, 현재 개발 중인 CNN 모델 코드입니다. 구체적으로, 학습 데이터는 다섯번째 코드에서 악성코드를 변환한 이미지가 됩니다. 이 이미지를 불러와서, Numpy 배열에 데이터를 대입하고, 테스트셋과 학습셋을 나누고, CNN 모델을 K-Fold Validation을 이용해 학습합니다. 현재는 학습이 잘 진행되지 않고 있습니다. 손실함수를 binary_cross_entropy()로 설정하였는데, 손실값이 e^-n 형태로 나오고 있고, 학습데이터-검증데이터의 손실값 그래프도 비정상적인 형태를 띠고 있습니다. 이 문제를 해결하기 위해, K-Fold Validation, 채널 크기 변경, 학습 데이터 이미지 크기 변경 등을 시도하였습니다. 앞으로는, 악성 / 정상 파일의 수를 줄여, 비슷한

비율에서 학습이 진행될 수 있도록 하거나, Confusion Matrix를 통해 학습이 제대로 진행되고 있는지 파악하는 등, 문제점을 해결하기 위해 노력해 나가겠습니다.

웹 및 데이터베이스 개발의 경우, Amazone AWS를 사용하여 개발 진행 중에 있습니다.

웹 및 데이터베이스 서버는 AWS에서 instance를 만들어서 사용 중에 있습니다. 먼저 데이터베이스는 AWS의 RDS를 사용하여 MariaDB 엔진을 이용해 데이터베이스를 만들고 로컬에 설치되어 있는 Mysql workbench를 통하여 DB를 관리하고 있습니다.



현재 만들어진 DB는 FileHeader, FileKey, OptionalHeader, SectionHeader의 테이블로 구성되어 있으며 FileKey에는 MD5, SHA-256 값을 가지고 있어 나머지 테이블에 두 값으로 접근이 가능하게 만들었습니다.

그리고 웹서버는 EC2 인스턴스에 Ubuntu 운영체제를 설치하여 nodejs, React를 설치하였습니다.

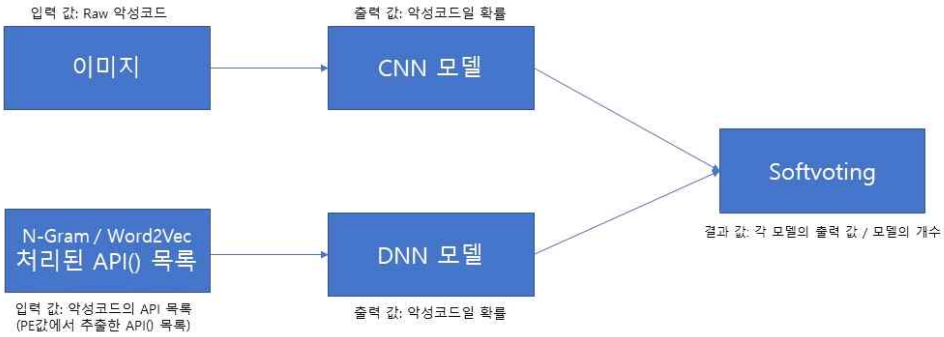
웹 개발은 로컬에서 React, Nodejs, VSCODE를 사용하여 개발하고 있습니다. 페이지는 MainPage, DetailsPage, AboutPage로 나눠집니다. 메인페이지는 Navbar, MainSection, Footer로 나뉘어 있으며 리액트의 Router를 이용하여 Detail, About으로 가는 메뉴가 있습니다. 왼쪽 페이지는 메인 페이지이며 현재로선 프론트엔드 부분만 완성된 상태입니다.

과제 추진 계획

- 앞으로 추진할 내용
- 5페이지 내외로 작성

프로젝트 종료 전까지는 다음과 같은 계획을 가지고 있습니다.

인공지능 모델의 개요도



인공지능 모델의 개요도처럼, CNN과 DNN 모델을 개발한 뒤, SoftVoting이라는 방식으로 각 모델의 예측값 평균을 계산하여, 모델의 신뢰성을 높일 예정입니다. 이때, DNN 모델은 PE값에서 추출한 API() 목록을 기반으로 N-Gram이나 Word2Vec을 적용하여 처리된 데이터를 학습데이터로써 사용하며, Classification 모델로써 개발할 계획입니다.
또한, 패킹 파일에 대한 공개된 Yara rule set을 활용하여, 패킹 여부 판단 및 어떤 종류의 패커을 사용했는지 판단하는 코드를 작성할 예정입니다. 패킹이 된 악성코드는 PE 값 등을 볼 수 없기 때문에 이를 해제할 필요가 있습니다. 이때 어떤 패커를 사용했는지 알 수 있다면, 패킹을 해제하는 데 많은 도움이 됩니다. 이는 패커마다 다른 방법으로 패킹을 진행하기 때문입니다. 구체적인 원리는, 악성코드/분석 파일의 일정 바이트를 가져온 뒤, YARA Rule set에 등록된 바이트와 일치하는지 여부를 검사하는 것입니다.
DB와 연동하는 코드를 작성할 예정입니다. 이 코드는 PE 값 등을 사전에 등록된 DB에 저장하거나, 새로운 파일이 들어왔을 때 기존에 저장된 값과 비교할 수 있도록 만들 예정입니다. 구체적으로, mysql.connector라는 라이브러리를 활용하여, DB를 연동하는 코드를 작성할 예정이며, 해당 라이브러리를 이용한 접속시험 등은 완료하였습니다.
악성코드 분석 서버 코드 작성(통합화)을 진행할 예정입니다. 지금까지 작성한 코드를 하나의 클래스 혹은 몇몇개의 파일로 모아 통합을 진행할 예정입니다. 이 코드는 악성코드 분석 서버에서 작동할 코드가 될 것입니다.
마지막으로 통합 테스트를 진행할 예정입니다. 이를 위해, 각 서버가 잘 연동이 되는지 확인할 예정이며, 각종 예외 처리가 잘 되는지 등도 살펴볼 예정입니다.

웹 파트의 경우 프론트엔드가 70% 정도 완성된 상태입니다. 리액트를 처음 사용해보아서 리액트를 공부하면서 프론트엔드를 만들다 보니 예정보다 늦어졌습니다. 따라서 5월 둘째 주 안에는 프론트엔드를 완성할 예정입니다. 프론트엔드의 남은 부분은 AboutPage 구성, Detail Page 구성, Footer 링크 연동입니다.

백엔드를 개발할 예정입니다. 프론트엔드가 완성된 후 백엔드에서 Main Page의 Upload 버튼을 통하여 Hash 값을 찾아주는 프로그램과 연결하여 SHA-256을 통하여 DB에 있는 SHA-256 값과 비교합니다. AWS DB에서 일치하는 정보가 있으면 DB에 있는 정보를 Table 형식으로 가져오게 Detail Page를 수정할 예정입니다. Upload 버튼은 로컬에 있는 파일을 선택하여 서버에 보내주는 역할을 합니다. 백엔드의 경우 Nodejs를 바탕으로 할 예정입니다.