



국민대학교
전자정보통신대학
컴퓨터공학부


캡스톤 디자인 I

종합설계 프로젝트

프로젝트 명	asi(a security insight)
팀 명	<i>assist(A Security Safety Important Special Team)</i>
문서 제목	수행계획서

Version	1.4
Date	2020-MAR-25

팀원	손 현기 (조장)
	김 주환
	김 호준
	오 예린
	이 동윤
	RUSLAN

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25


CONFIDENTIALITY/SECURITY WARNING

이 문서에 포함되어 있는 정보는 국민대학교 전자정보통신대학 컴퓨터공학부 및 컴퓨터공학부 개설 교과목 캡스톤 디자인I 수강 학생 중 프로젝트 “asi(a security insight)”를 수행하는 팀 “물약”의 팀원들의 자산입니다. 국민대학교 컴퓨터공학부 및 팀 “assist(A Security Safety Important Special Team)”의 팀원들의 서면 허락없이 사용되거나, 재가공 될 수 없습니다.

문서 정보 / 수정 내역

Filename	수행계획서-asi(a security insight)
원안작성자	김주환
수정작성자	김주환

수정날짜	대표수정자	Revision	추가/수정 항목	내 용
2020-03-13	김주환	1.0	최초 작성	전문에 대한 초안 작성
2020-03-15	김주환	1.1	내용 수정	역할 수정
2020-03-18	김주환	1.2	내용 수정	오타 수정
2020-03-23	김주환	1.3	내용 추가	국내외 경쟁기관 추가
2020-03-25	김주환	1.4	내용 추가	현실적 제한요소 추가

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

목 차

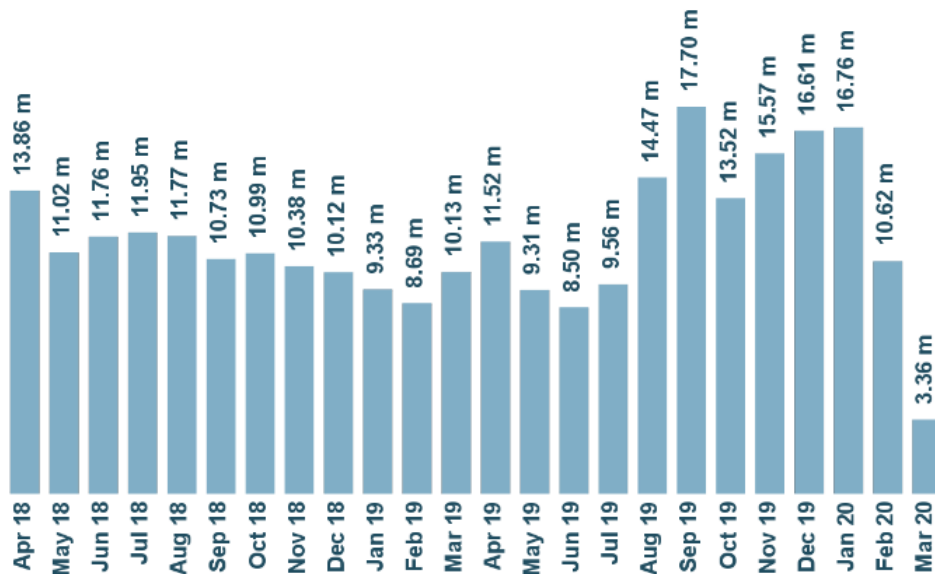
1 개요	4
1.1 프로젝트 개요	4
1.2 추진 배경 및 필요성	7
1.2.1 국내외 악성코드 분석 연구 동향	7
1.2.2 국내외 시장 현황	13
1.2.3 국내외 경쟁기관 현황	14
1.2.4 딥러닝 기반 악성코드 분석 보조도구의 필요성	19
2 개발 목표 및 내용	21
2.1 목표	21
2.2 연구/개발 내용	21
2.2.1 데이터셋 확보 및 opcode 추출	23
2.2.2 딥러닝 기반 opcode 특징 벡터 추출 도구 개발	23
2.2.3 딥러닝 기반 자동 악성행위 코드 추출 도구 개발	24
2.3 개발 결과	27
2.3.1 시스템 기능 요구사항	27
2.3.2 시스템 비기능(품질) 요구사항	28
2.3.3 시스템 구조	28
2.3.4 결과물 목록 및 상세 사양	29
2.4 기대효과 및 활용방안	29
3 배경 기술	30
3.1 기술적 요구사항	30
3.2 현실적 제한 요소 및 그 해결 방안	31
3.2.1 실행파일의 목적 시스템	31
3.2.1 패킹(packing)	31
3.2.2 하드웨어	31
4 프로젝트 팀 구성 및 역할 분담	32
5 프로젝트 비용	33
6 개발 일정 및 자원 관리	34
6.1 개발 일정	34
6.2 일정 별 주요 산출물	35
6.3 인력자원 투입계획	36
6.4 비 인적자원 투입계획	37
7 참고 문헌	38



1 개요


1.1 프로젝트 개요

정보보안 전문 기업 AV-Test의 통계 조사에 따르면 매일 약 350,000 개의 악성코드가 새로 생성되고 있으며, 2019년에는 매달 평균적으로 약 12,075,800 개의 악성코드가 새로 생성되었다 [1]. 매일 생성되는 수많은 악성코드를 전문가가 분석하는 것은 현실적으로 어려우므로 일반적으로 자동 분석 도구를 통해 악성코드를 분석하고, 대응하는 방법을 채택하고 있다. 전통적인 자동 분석 방안으로는 악성코드의 행위 규칙이나 해시 값을 이용하는 서명 탐지(signature detection)와 정상코드의 API(application programming interface) 규칙 기반의 이상 탐지(anomaly detection)가 있다. 그러나, 전통적인 방법은 변종 악성코드를 탐지하지 못하는 문제가 존재한다 [2, 7].

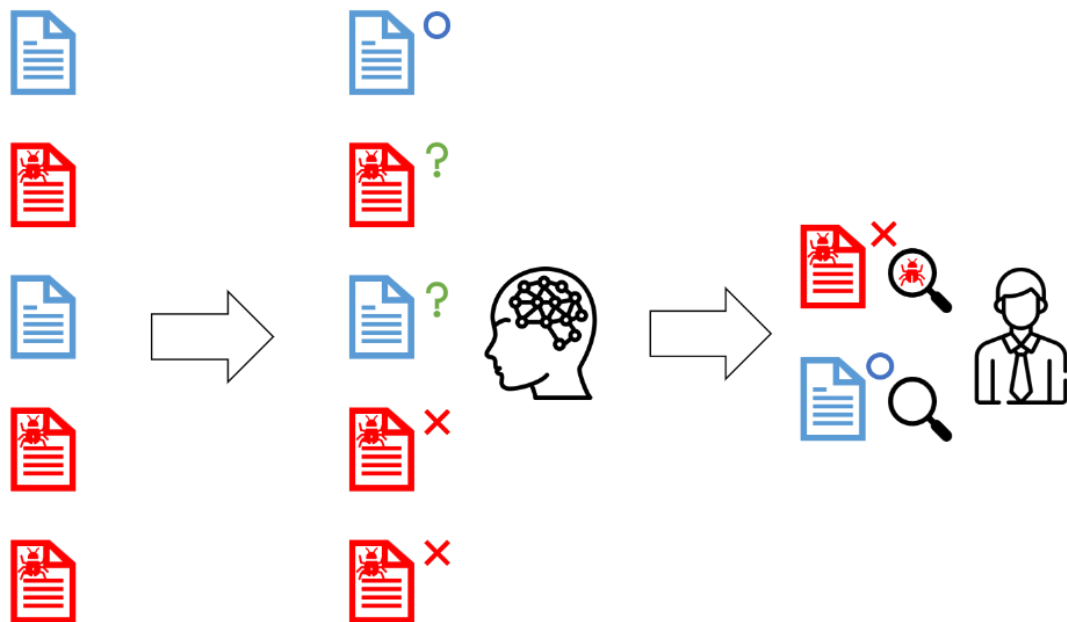


[그림 1] AV-Test의 신종 악성코드 발생량 통계조사 결과 [1]

최근에는 기계학습(machine learning) 및 딥러닝(deep learning)을 기반으로 하는 악성코드 자동 분석기 연구가 활발히 진행되고 있다 [3-6, 8-10]. 딥러닝 기반의 악성코드 분석은 데이터로부터 특징을 학습하여 분류를 수행하므로 악성코드의 변형을 감내할 수 있으며, 전통적인 분석 방법에 비해 월등히 높은 성능을 갖는다. 그러나, 전통적인 암호 분석 도구 뿐만 아니라 딥러닝 기반의 악성코드 분석기 역시 100%의 정확도로 분석할 수 없으므로, 일부 악성코드에 대해서는 여전히 전문가의 분석이 필요하다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

[그림 2]는 일반적인 악성코드 분석 방법을 도식화한 것이다. 악성파일(붉은색)과 정상파일(파란색)이 입력으로 들어왔을 때, 먼저 자동 분석기가 파일의 악성/정상 여부를 판별한다. 대부분의 파일은 이 단계에서 분류되나, 악성/정상 여부가 불분명한 일부 파일에 대해서는 분석기가 판단하지 못하고 전문가에게 분석을 요청한다. 최근 딥러닝 기반 자동 분석기의 성능은 크게 향상되고 있으나, 악성코드의 수 역시 증가하고 있는 추세이므로 여전히 전문가가 많은 악성코드를 직접 분석해야 하는 실정이다.

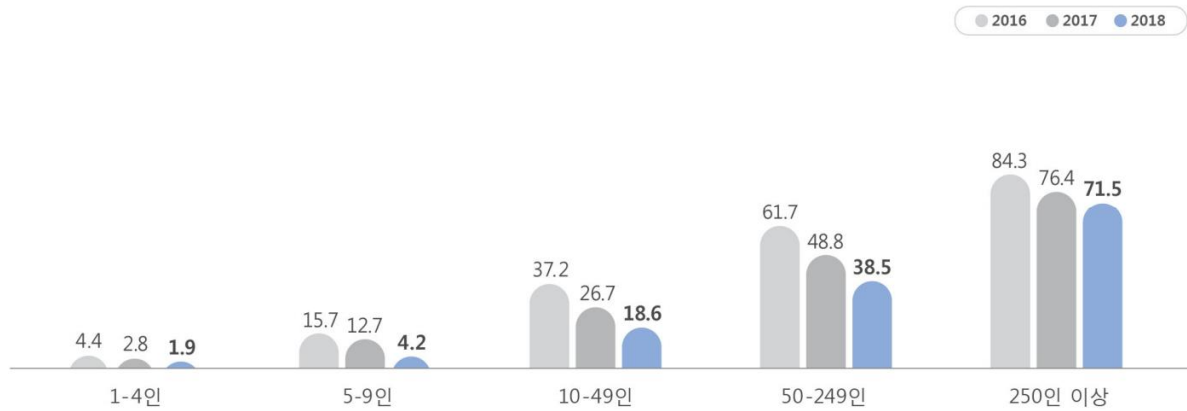


[그림 2] 일반적인 악성코드 분석 방법

악성코드 분석 전문가의 수요는 증가하는 반면, 국내의 정보보호에 대한 투자는 매우 빈약한 상황이다. 과학기술정보통신부의 2018년 정보보호실태조사에 따르면 국내 기업 중 정보보호 조직을 보유하지 않은 기업의 비율은 94.5%이고, 정보보호 또는 개인정보보호에 투자하지 않는 기업의 비율은 63.8%, IT 예산 중 1% 미만을 투자하는 비율은 89.1%에 달한다 [12].



계획서		
프로젝트 명	asi(a security insight)	
팀 명	assist(A Security Safety Important Special Team)	
Confidential Restricted	Version 1.4	2020-MAR-25




[그림 3] 규모별 정보보호(개인정보보호) 조직 보유율 [12]

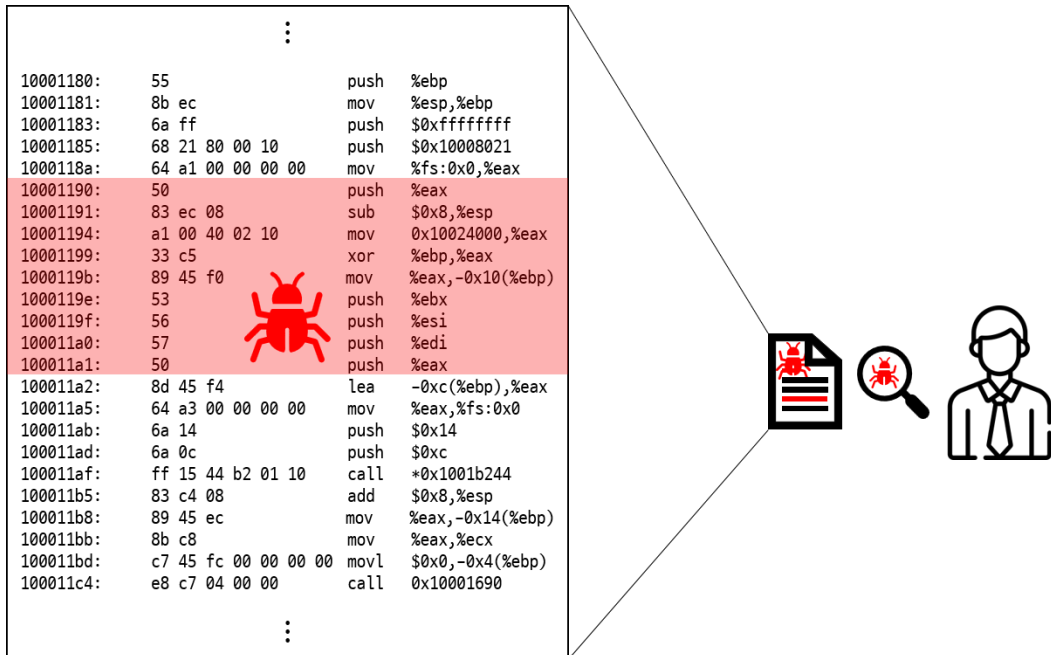


[그림 4] IT 예산 중 정보보호(개인정보보호) 예산 비중 (투자하지 않는 기업은 미표기) [2]

정보보안 전문 기업 AhnLab에 따르면, 전문가가 하나의 악성코드를 분석하는데 최소 몇 시간에서 최대 몇 주가 소요된다 [11]. 최근 악성코드의 크기가 커지면서, 전문가가 확인해야 하는 코드의 수가 증가하는 추세이므로 악성코드 분석에 소요되는 시간은 더 길어질 것으로 예상된다. 정보보안에 대한 투자가 적고 생성되는 악성코드는 많아지며, 분석에 소요되는 시간이 길어지는 현상 상황에서 안전한 시스템을 구축하기 위해서는 전문가의 악성코드 분석시간을 줄일 수 있는 소프트웨어의 개발이 시급하다.

우리의 목표는 [그림 5]와 같이 악성코드 중 전문가가 중점적으로 검사해야 하는 부분을 요약하는 딥러닝 기반 소프트웨어를 개발함으로써 분석시간을 줄이는 것이다. 우리의 소프트웨어가 성공적으로 개발된다면, 동일한 인력으로 더 많은 악성코드를 효율적으로 분석할 수 있으므로 더욱 안전한 시스템을 개발하는데 도움이 될 것으로 기대된다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25



[그림 5] 제안하는 소프트웨어의 목표

1.2 추진 배경 및 필요성

1.2.1 국내외 악성코드 분석 연구 동향


악성코드 분석 연구는 [그림 2]의 첫 번째 단계인 자동 분석기의 성능 향상에 치중되어 있고, 두 번째 단계인 전문가의 악성코드 분석 보조도구에 대한 연구는 전무한 상황이다. 본 항에서는 악성코드 분석 연구 동향을 살펴봄으로써 2장에서 기술할 우리가 제안하는 연구의 방법론과 필요성에 대한 기반 지식을 제시한다.

악성코드 분석 연구의 목적은 크게 악성/비악성 분류 [3-5], 악성코드의 패밀리 분류 [6-10]로 나눌 수 있다. 악성/비악성 분류기는 가장 기본적인 연구 목표로 입력 파일의 악성 행위 여부를 판단한다. 악성코드 패밀리 분류는 악성코드의 실행 행위에 따라 패밀리를 분류하는 것으로 적절한 방어기법 제공을 위해 필요하다. 아래의 두 목에서 각 연구 동향을 상세히 기술한다.

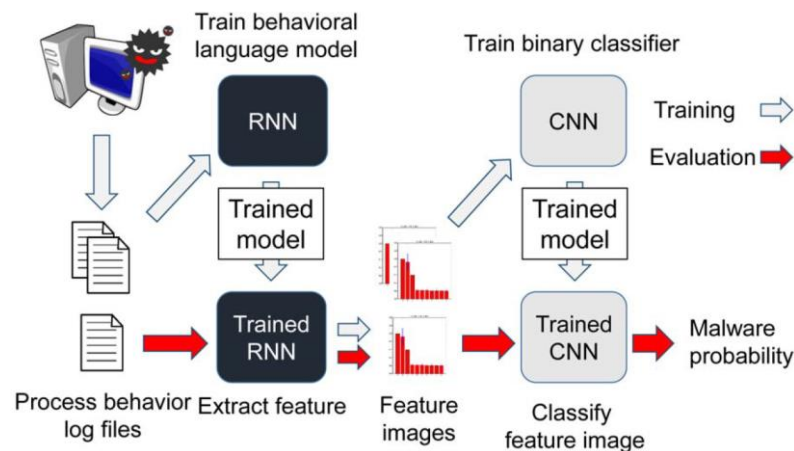
1.2.1.1. 악성코드 분류 연구 동향

● Malware Detection with Deep Neural Network Using Process Behavior [3]

Tobiyama, 등은 RNN(recurrent neural network, 순환신경망), CNN(convolution neural network, 합성곱 신경망)을 기반으로 프로그램의 트래픽을 분석하여 악성파일과 정상파일을 최대 96%의 정확도로

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

분류하는 기법을 제안했다. 분류기는 [그림 6]과 같이 세 단계를 거쳐 분류를 수행한다. 먼저 프로그램의 트래픽을 Microsoft 사의 Process Monitor를 이용하여 트래픽 데이터를 수집한다. 다음으로 RNN의 일종인 LSTM(long short-term memory, 장단기메모리)을 이용해 트래픽으로부터 특징 이미지를 생성한다. 마지막으로 특징 이미지를 CNN으로 분석하여 파일의 정상/악성 여부를 판별한다.




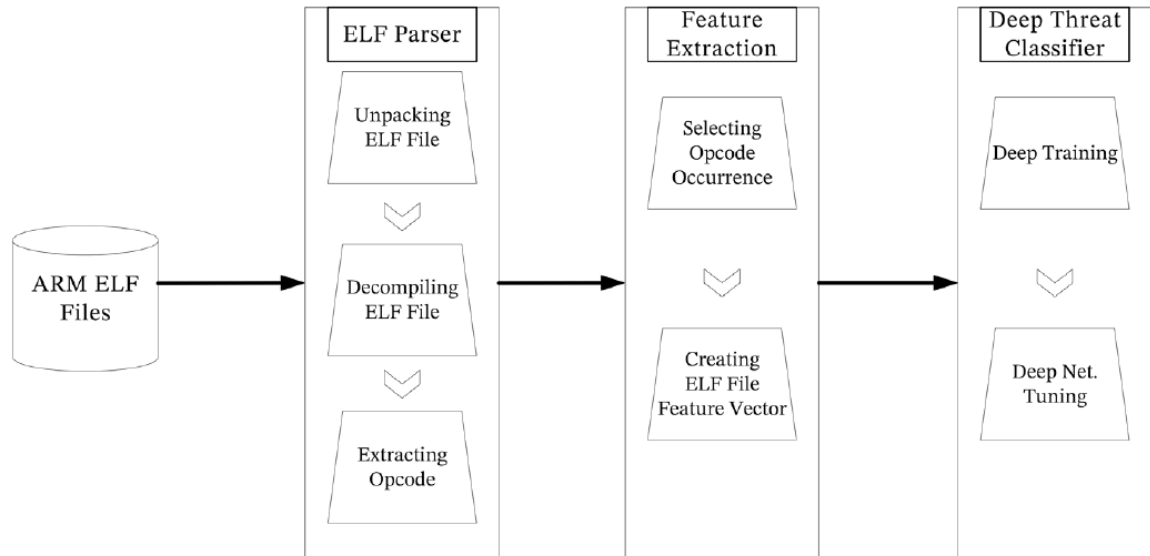
[그림 6] [3]의 악성코드 분류 방법의 개요

이 논문에서 주목해야 할 점은 두 번째 단계인 특징 추출(feature extraction) 단계이다. 저자들은 LSTM이 언어 모델(language model)을 학습할 수 있도록 현재 명령어를 입력으로 받아 다음 명령어를 예측하도록 학습시켰다. 이는 정상코드와 악성코드의 명령어의 분포가 다를 것을 시사하므로, 정상코드에 대한 명령어의 분포를 학습시킨 뒤 악성코드의 명령어의 분포를 검사한다면, 파일 중 악성 행위를 수행하는 코드를 탐지할 수 있을 것으로 기대된다.

● A Deep Recurrent Neural Network based approach for Internet of Things malware threat hunting [4]

HaddadPajouh, 등은 양방향 RNN(bidirectional RNN, BRNN)을 기반으로 프로그램의 opcode를 분석하여 악성파일과 정상파일을 최대 98.18%의 정확도로 분류하는 기법을 제안했다. 저자들은 ELF(executable and linking format) 파일에서 GNU 바이너리 유틸리티 중 disassembler인 objdump를 이용해 opcode를 추출했다. 특징의 개수를 줄이기 위해 정보이득(information gain, IG) 값을 계산하여 0.3 초과인 opcode에 대한 분석을 수행했다.

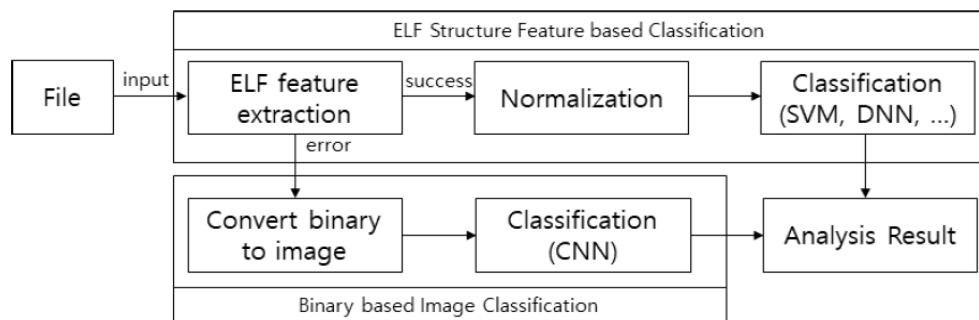
 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25



[그림 7] [4]의 악성코드 분류 방법의 개요

● 정적 분석과 앙상블 기반의 리눅스 악성코드 분류 연구 [5]


황준호, 등은 SVM(support vector machine), Random Forest, naïve Bayes, k-NN(k-nearest neighbor), MLP를 기반으로 ELF 파일을 최대 99.68%의 정확도로 분석하는 기법과, CNN을 기반으로 이미지화된 바이너리 데이터를 95.05%로 분석하는 기법을 제안했다. 정상코드와 악성코드의 ELF 파일의 통계적 특성(ELF header의 크기, program header의 수, 등)이 차이가 있음을 보임으로써 ELF 파일 기반 악성코드 분석의 타당성을 보였다. 또한, ELF 파일이 존재하지 않는 파일을 CNN 기반으로 분석하는 방법을 제안했다.



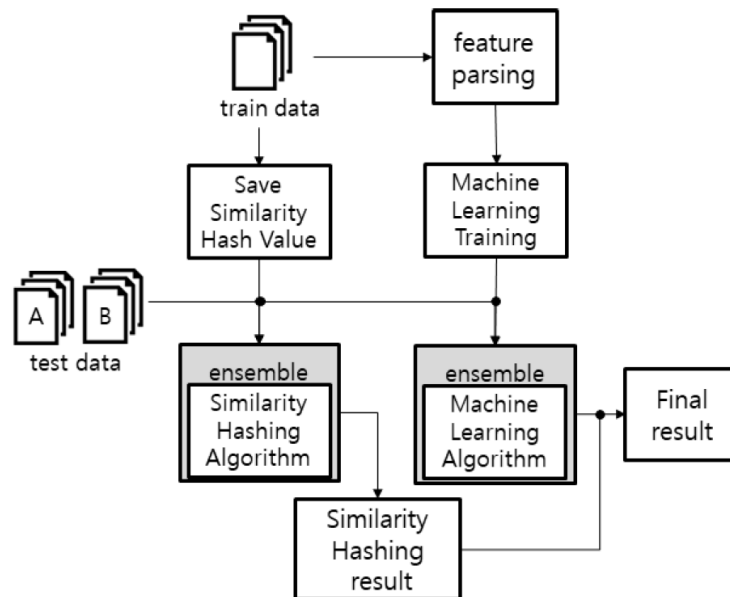
[그림 8] [5]의 악성코드 분류 방법의 개요

● 정적 분석 기반 기계학습 기법을 활용한 악성코드 식별 시스템 연구 [6]

김수정, 등은 유사성 해시 알고리즘과 기계학습 알고리즘 앙상블을 기반으로 PE 파일을 분석하여 악성파일과 정상파일을 최대 95.9%의 정확도로 분류하는 기법을 제안했다. 이 연구에서는

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

ssdeep, DHASH(difference hash), TLSH(trend micro locality sensitive hash) 세 유사성 해시 알고리즘을 앙상블하고, k-NN, Decision Tree, SVM, MLP 네 기계학습 알고리즘을 앙상블하여 정확도를 향상시켰다.




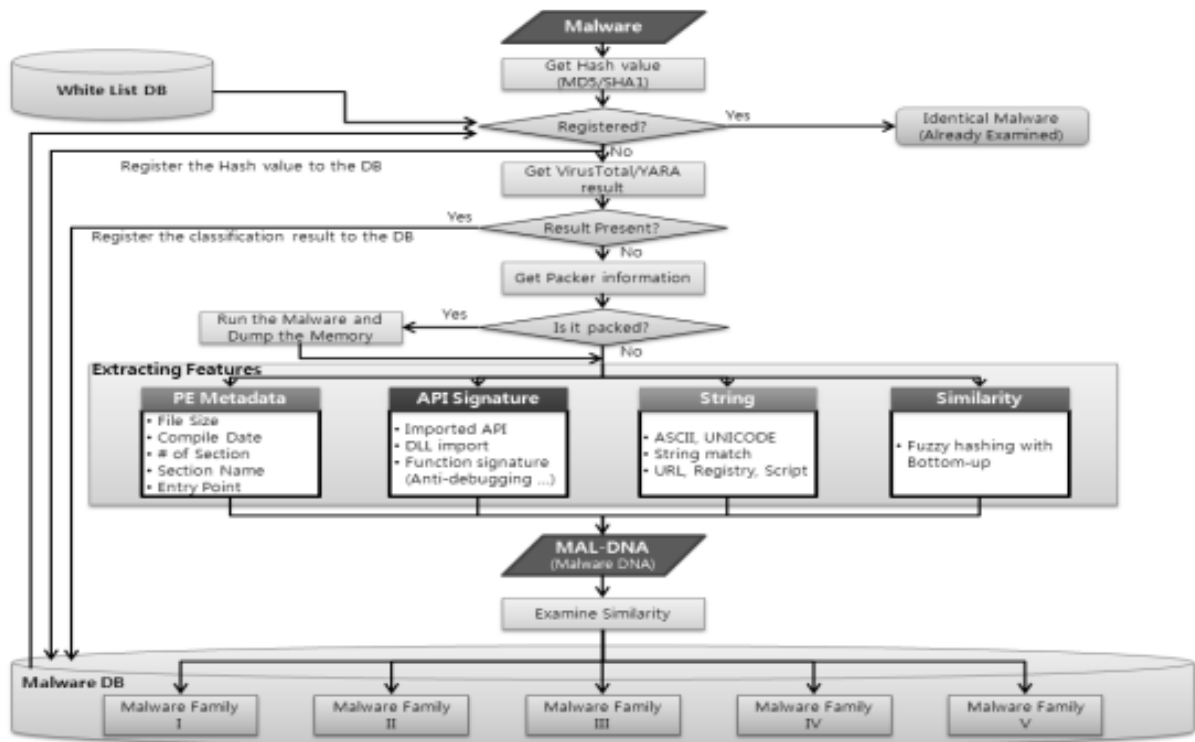
[그림 9] [6]의 악성코드 분류 방법의 개요

1.2.1.2. 악성코드 패밀리 분류 연구 동향

● 악성코드 DNA 생성을 통한 유사 악성코드 분류기법 [7]

한병진, 등은 API 호출, PE 파일의 구조 등을 규칙 기반으로 분석하여 4 개의 악성코드 패밀리를 약 83.9%의 정확도로 분류하는 기법을 제안했다. 이전의 연구에서는 악성코드의 일부 특징만을 이용하여 분류를 수행했다면, 이 연구에서는 API뿐만 아니라 다양한 특성인자를 활용하여 유사도를 검증함으로써 정확도를 높였다. 저자들은 PE(portable executable) 파일의 메타데이터, API 서명, 스트링, 등을 이용해 입력된 파일별로 악성코드 DNA (malware DNA)를 계산할 수 있는 방법론을 제안했다.

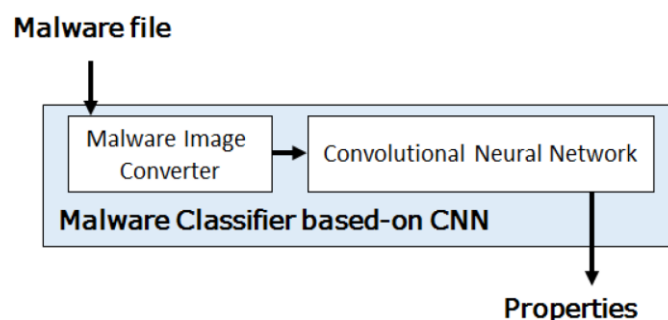
 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25




[그림 10] [7]의 악성코드 DNA(Malware DNA) 생성 및 분류 방법의 개요

● Convolutional Neural Network 기반의 악성코드 이미지화를 통한 패밀리 분류 [8]

석선희, 등은 CNN을 기반으로 이미지화한 바이너리 데이터를 분석하여 악성코드 패밀리의 수가 9인 경우 96.2%의 정확도, 패밀리 수가 27인 경우 82.9%의 정확도로 분류하는 기법을 제안했다. 제안된 방법은 특징 추출과 같은 별도의 전처리 없이 CNN만으로 종단간 기계학습(end-to-end machine learning)을 수행하므로 하나의 악성코드를 분류하는데 4ms의 빠른 시간 내에 분류가 가능하다는데 의의가 있다.

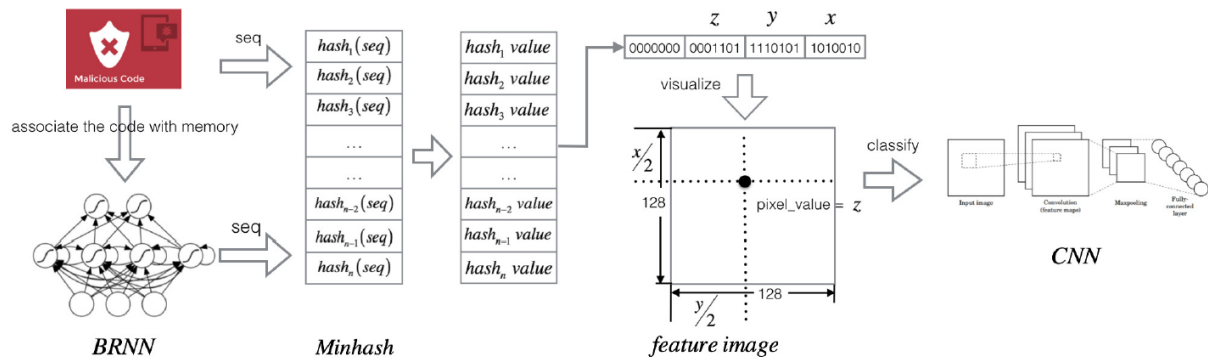


[그림 11] [8]의 악성코드 분류 방법의 개요

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

● Deep Learning and Visualization for Identifying Malware Families [9]


Sun, 등은 BRNN과 CNN을 기반으로 opcode를 분석하여 6 개의 악성코드 패밀리를 99.5%의 정확도로 분류하는 기법을 제안했다. 제안된 기법은 BRNN을 이용해 추출한 특징과 바이너리 코드 각각에 대해 MinHash 알고리즘을 적용한 것을 결합한 후 이미지화한다. 생성된 이미지를 CNN으로 분석해 악성코드 패밀리를 분류한다. 이 연구에서는 opcode의 수를 줄이기 위해 가장 많이 나온 255개의 opcode를 남기고, 나머지 opcode는 하나의 새로운 opcode로 치환했다. LSTM의 그레디언트 소멸(gradient extinction) 문제와 그레디언트 폭발(gradient explosion) 문제를 막기 위하여 GRU(gated recurrent unit, 게이트 순환 유닛)를 이용해 BRNN을 구현했다.

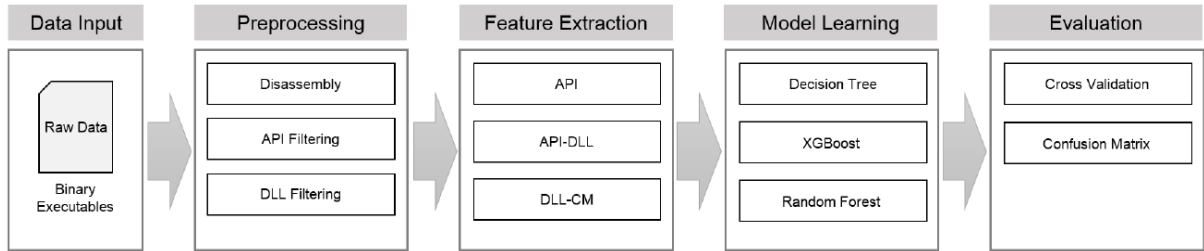


[그림 12] [9]의 악성코드 분류 방법의 개요

● 악성코드 패밀리 분류를 위한 API 특징 기반 앙상블 모델 학습 [10]

이현중, 등은 XGBoost, Random Forest를 기반으로 API와 DLL(dynamic link library)를 분석하여 패밀리의 수가 2인 경우 93.6%의 정확도, 패밀리의 수가 5인 경우 93.5%의 정확도로 분류하는 기법을 제안했다. 제안된 기법은 API, API-DLL, DLL-CM 세 가지 특징을 추출하고, 각각에 대해 신경망을 학습시켰다. API는 API 함수로부터 추출한 특징이며, 차원 감소를 위해 사용 빈도가 0.10% 이상인 API만을 학습에 사용했다. API-DLL은 사용 빈도가 0.25% DLL으로부터 사용된 API를 추출했으며, DLL-CM은 DLL의 호출 관계를 CM(call matrix)로 표현한 것으로 DLL 호출 순서로부터 2-gram 구조를 2차원 행렬 형태로 변형하여 생성한다.

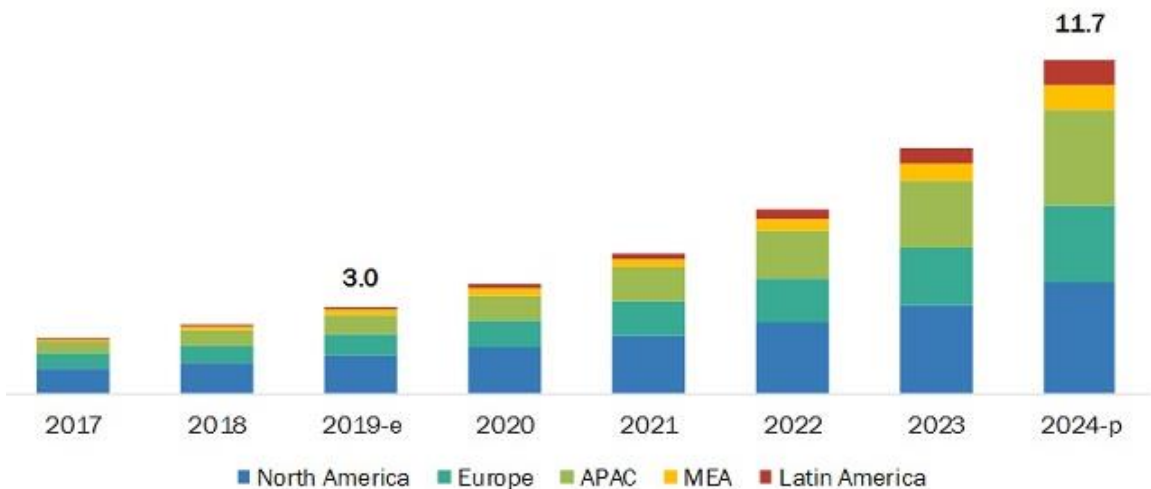
 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25



[그림 13] [10]의 악성코드 분류 방법의 개요

1.2.2 국내외 시장 현황


세계적인 시장예측 기관인 Markets and Markets의 조사에 따르면 2019년 세계 악성코드 분석 시장 규모는 약 30억 달러(한화 3조 6,540억 원) 규모이고, 연평균 성장률(compound annual growth rate, CAGR)은 31.0%로 2024년에는 117억 달러(한화 14조 2,506억 원) 규모로 성장할 것으로 예측했다 [13].



[그림 14] 악성코드 분석 시장규모 및 예측치 (단위: USD Billion) [13]

악성코드 분석과 관련된 국내외 시장은 빠르게 성장하고 있고, 딥러닝 기반 악성코드 분석기에 대한 연구가 활발하게 진행되고 있는 한편, 1.2.1 항에서 살펴본 것처럼 우리의 목표인 딥러닝 기반 악성코드 분석 보조도구에 대한 연구는 전무한 상황이다.

[표 1]은 국내외 주요 수요처를 나열한 것이다. 본 기술은 정보보안 분석도구로 제공되기 때문에 정확한 수요량을 파악하기 어려우므로 기재하지 않았다.


 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

[표 1] 국내외 주요 수요처 현황

수요처	국명
국가정보원	대한민국
군사안보지원사령부(구 국군기무사령부)	대한민국
한국정보통신기술협회(TTA)	대한민국
한국기계전기전자시험연구원(KTC)	대한민국
국가보안기술연구소(NSR)	대한민국
한국인터넷진흥원(KISA)	대한민국
AhnLab	대한민국
SKinfosec	대한민국
EST Security	대한민국
WINS	대한민국
Penta Security	대한민국
IGLOO Security	대한민국
root9B	미국
CheckPoint	이스라엘
Sophos	영국
DarkTrace	영국
Qihoo 360	중국

1.2.3 국내외 경쟁기관 현황

정보보안기업은 새로운 보안위협에 대응하기 위해 신종 악성코드 및 스파이웨어를 분석하고 악성코드 분석 엔진을 업데이트하기 위해 보안 전문가로 이루어진 대응 조직을 운영하고 있다. 국내에서 바이러스 분석 및 대응 소프트웨어로 유명한 V3, 알약을 각각 개발한 AhnLab과 EST security는 이러한 대응 조직으로 각각 안랩 시큐리티대응센터(AhnLab Security Emergency response Center, ASEC) [14], 이스트시큐리티 시큐리티대응센터(ESTsecurity Security Response Center, ESRC) [15]를 운영하고 있다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25



[그림 15] 안랩 시큐리티대응센터의 대응 프로세스 [14]


각 대응센터에서는 악성코드 분석 보조도구가 없거나, 공개하지 않는 상황이다. 따라서 보안에 대한 투자가 적은 기업이나 기술력이 충분하지 않은 기업은 빠르고 정확한 악성코드 분석이 어렵다.


한편, 이후에 살펴볼 바이러스 분석 프로그램 개발기업 현황에서 우리가 제안하는 악성코드 분석 보조도구 프로그램은 전무하고, 악성코드 자동 분석기의 개발만 활발히 진행되고 있는 상황으로 우리가 제안하는 기술의 개발이 시급하다.

● VirusTotal [19]

VirusTotal 은 여러 바이러스 검사 소프트웨어 엔진을 이용해 업로드된 파일 또는 URL(uniform resource locator)의 악성여부를 검사하는 웹사이트이다. 악성코드 검사 엔진으로는 안랩, 알약, Kaspersky, McAfee, 등 72 개의 악성코드 분석 엔진을 사용하며, domain 검사 엔진으로는 Alexa, Google Safebrowsing, 등 68 개의 분석 엔진 및 데이터 셋을 사용한다.

[그림 16]은 C 언어로 작성한 소스코드를 Visual Studio 2019 로 컴파일한 예제 실행 파일을 VirusTotal 에 업로드한 결과이다. 파일 타입 등의 문제로 검사하지 못한 일부 엔진을 제외한 총 69 개의 분석 엔진으로 검사한 결과가 표시되어 있다. 악성 행위를 수행하지 않는 코드임에도 불구하고 SecureAge APEX 엔진에서는 악성으로 오탐한 것을 볼 수 있다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25



① One engine detected this file

c5dac27fa8a61c5459c2afe55a6c63b53468ea6ceb3c654da28d6de5bdf730b

test.exe

21.50 KB
Size

2020-03-23 09:48:27 UTC
1 minute ago

64bits assembly peexe runtime-modules

DETECTION	DETAILS	BEHAVIOR	COMMUNITY
SecureAge APEX	① Malicious	Acronis	Undetected
Ad-Aware	Undetected	AegisLab	Undetected
AhnLab-V3	Undetected	Alibaba	Undetected
ALYac	Undetected	Antiy-AVL	Undetected
Arcabit	Undetected	Avast	Undetected
Avast-Mobile	Undetected	AVG	Undetected
Baidu	Undetected	BitDefender	Undetected
BitDefenderTheta	Undetected	Bkav	Undetected
CAT-QuickHeal	Undetected	ClamAV	Undetected
CMC	Undetected	Comodo	Undetected
CrowdStrike Falcon	Undetected	Cybereason	Undetected
Cylance	Undetected	Cyren	Undetected
DrWeb	Undetected	eGambit	Undetected

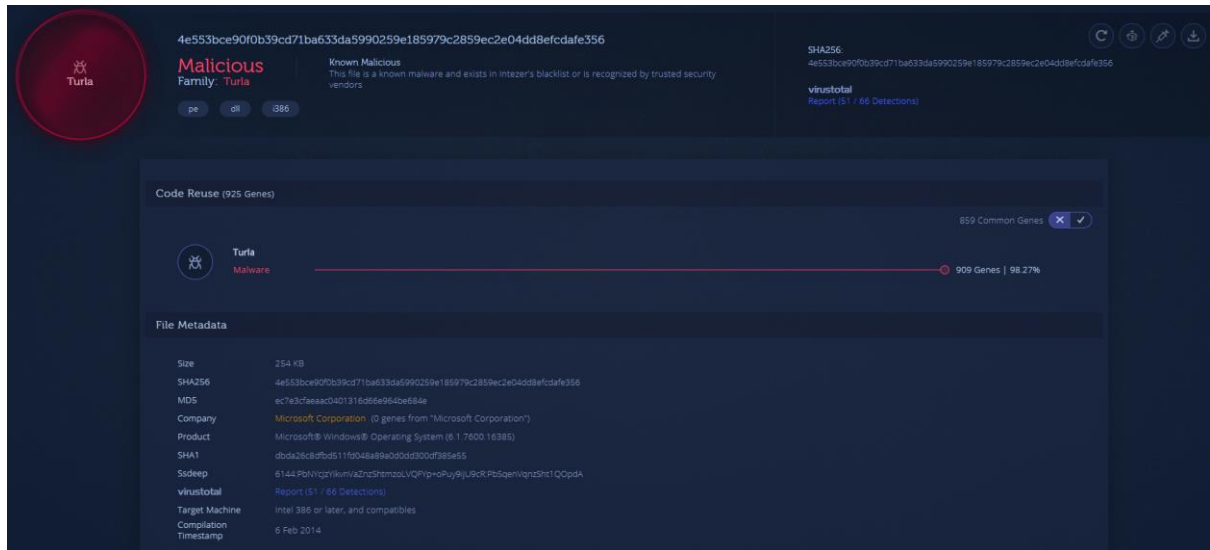
[그림 16] VirusTotal 정상파일 분석 결과

● Intezer [20]

Intezer 는 악성코드 유전자 분석 기법(genetic malware analysis)을 적용해 악성코드 패밀리를 분류하는 회사이다. 악성코드를 작은 조각(유전자)으로 나눈 뒤, 각 작은 조각을 데이터베이스의 코드 조각들과 비교하여 코드의 악성여부를 판별한다.

[그림 17]은 Turla 악성코드를 분석한 결과이다. 분석 결과 총 925 개의 코드 조각 중 909 개가 Turla 와 유사한 것을 볼 수 있으며, 이는 VirusTotal 의 분석 결과 66 개의 악성코드 분석 엔진 중 51 개의 엔진에서 악성여부를 탐지한 것과 일치한다. 분석 결과에서 추가로 파일의 메타데이터(metadata)와 스트링 정보 등을 추가로 확인할 수 있다.

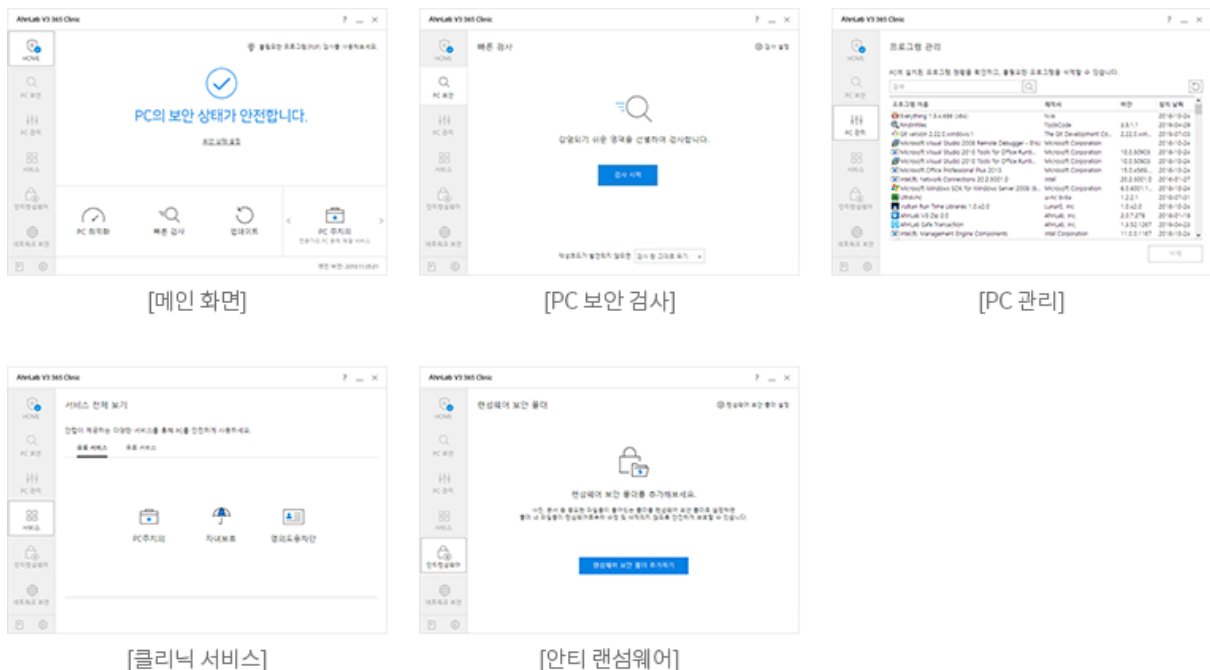
 <div> <p>국민대학교 컴퓨터공학부 캡스톤 디자인 I</p> </div>	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25




[그림 17] Intezer의 Turla 악성코드 분석 결과

● V3 [21]

V3는 AhnLab에서 개발한 보안 프로그램이다. 시스템에 저장되어 있는 파일을 행위 기반, 평판 기반으로 분석 및 대응하는 것이 주 목적이며, 정상 또는 악성 여부가 불분명한 파일은 클라우드 기반으로 분석하여 변종 악성코드에 대응하는 기능을 제공한다.

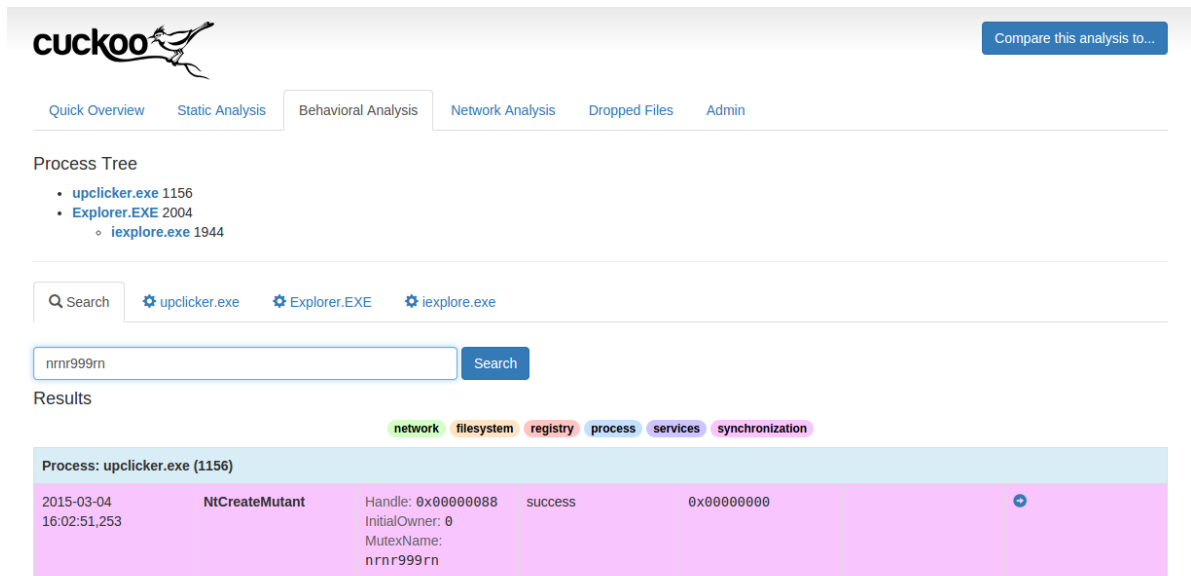


[그림 18] V3 365의 주요 기능

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

● Cuckoo Sandbox [22]

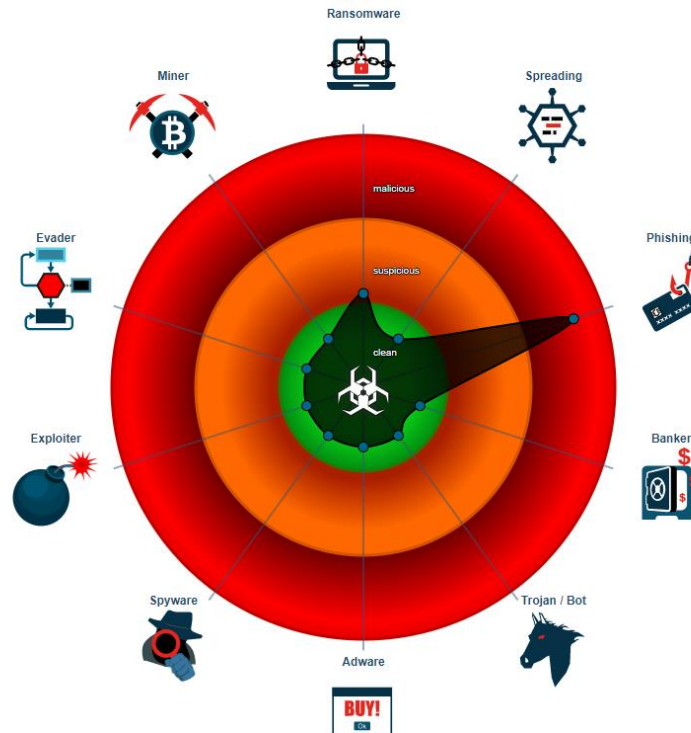
Cuckoo Sandbox 는 악성코드를 자동으로 동적분석하는 시스템으로, 악성코드를 가상환경에서 실행시켜 악성코드의 프로세스, 생성한 파일, 네트워크 트래픽, 메모리, 등을 분석한다. 악성코드를 시스템에서 수행시키면 악영향을 미칠 수 있으므로 독립된 가상환경에서 악성코드를 수행시키고 동작 정보를 수집한다. 분석 가능한 파일 형식은 윈도우 실행파일, DLL 파일, PDF 파일, 등이다.



[그림 19] Cuckoo sandbox의 동적 분석 결과

● Joe Sandbox [23]

Joe Sandbox 는 악성코드를 자동으로 동적분석하는 시스템으로, 악성코드를 가상환경에서 실행시켰을 때의 동작 정보를 분석한 뒤, [그림 20]과 같이 결과를 시각화하여 제시한다. 이 프로그램은 Windows, macOS, Linux, Android, iOS 와 같은 다양한 플랫폼에서 악성파일, 악성문서가 동작할 때의 분석 결과를 제공하는 장점이 있다.



[그림 20] Joe Sandbox의 동적 분석 결과

1.2.4 딥러닝 기반 악성코드 분석 보조도구의 필요성


딥러닝 기반의 분류기는 다음과 같이 두 가지 문제점이 있으므로 일부 악성코드에 대해서는 전문가의 추가 분석이 필요하다.

- **정확도 문제**

1.2.1 에서 살펴본 어떠한 연구 결과에서도 100%의 정확도를 갖는 악성코드 자동 분석기를 개발하지 못했다. 그러나, 분석기가 하나의 악성코드라도 정상이라 판정해 적절한 대응을 하지 못한다면, 시스템의 보안에 큰 위협이 될 수 있다. 따라서 자동 분석 결과가 불명확한 경우, 즉 파일의 정상/악성 여부를 높은 신뢰도로 판정하지 못하는 경우, 전문가의 추가 분석이 필요하다.


- **Zero-day attack**

딥러닝의 성능은 데이터셋의 영향을 크게 받는다. 딥러닝 기반의 악성코드 자동 분석기 역시 학습에 사용한 데이터셋에 있는 악성코드나, 일부 변형된 악성코드는 높은 정확도로 판별할 수 있지만, 새롭게 개발된 악성코드를 제대로 분류하지 못하는 문제가 있다. 또한, 악성코드의 변형이 반복되는 경우 이전 악성코드와의 유사도가 떨어져 오래된 데이터셋으로

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

학습된 분석기가 제대로 분류를 하지 못하는 시간붕괴 문제가 있다. 이 경우 전문가의 추가 분석이 필요하다.

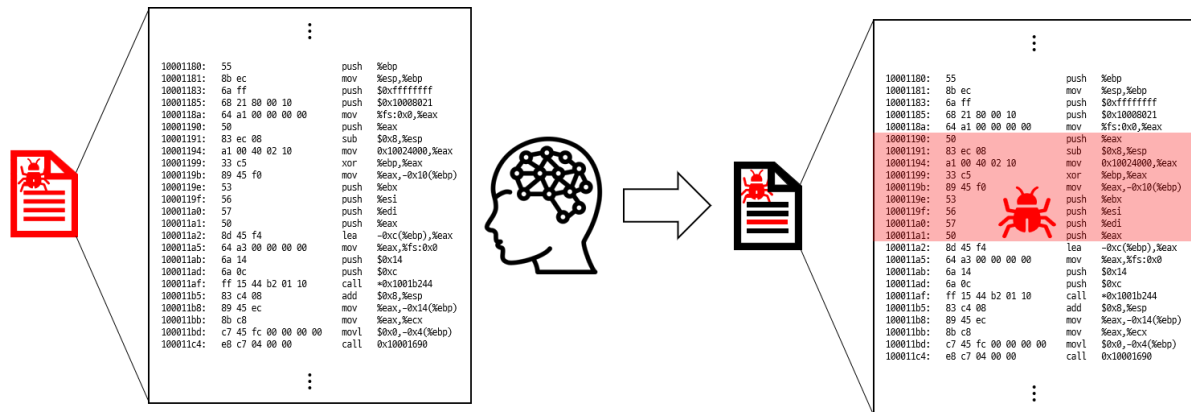
현재 악성코드는 매년 크게 증가하고 있는 반면, 악성코드 분석 인력은 매우 적은 상황이다. 적은 인력으로 높은 분석 효율을 달성해야 하는 현상황에서, 악성코드 분석에 대한 연구는 1.2.1 에서 살펴보았듯 분류기의 성능 향상에 치중되어 있고, 전문가의 분석을 돕는 딥러닝 기반의 보조 도구 연구는 전무하며, 1.2.3 에서 살펴보았듯 악성코드 분석도구 역시 악성행위 판정과 패밀리 분류를 위한 자동분석 도구가 주로 개발되고 있는 상황이다.. 따라서 전문가의 분석 시간을 줄이기 위한 기술 연구가 시급하다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

2 개발 목표 및 내용

2.1 목표


본 프로젝트의 목표는 딥러닝 기반 악성코드 분석 보조도구를 개발하는 것이다. 파일을 입력으로 받아 역어셈블(disassemble)을 수행한 뒤, 추출된 opcode 중 악성 행위를 수행하는 부분을 요약하는 소프트웨어를 개발함으로써 적은 인력으로도 악성코드를 효과적으로 분석할 수 있는 소프트웨어를 개발한다.

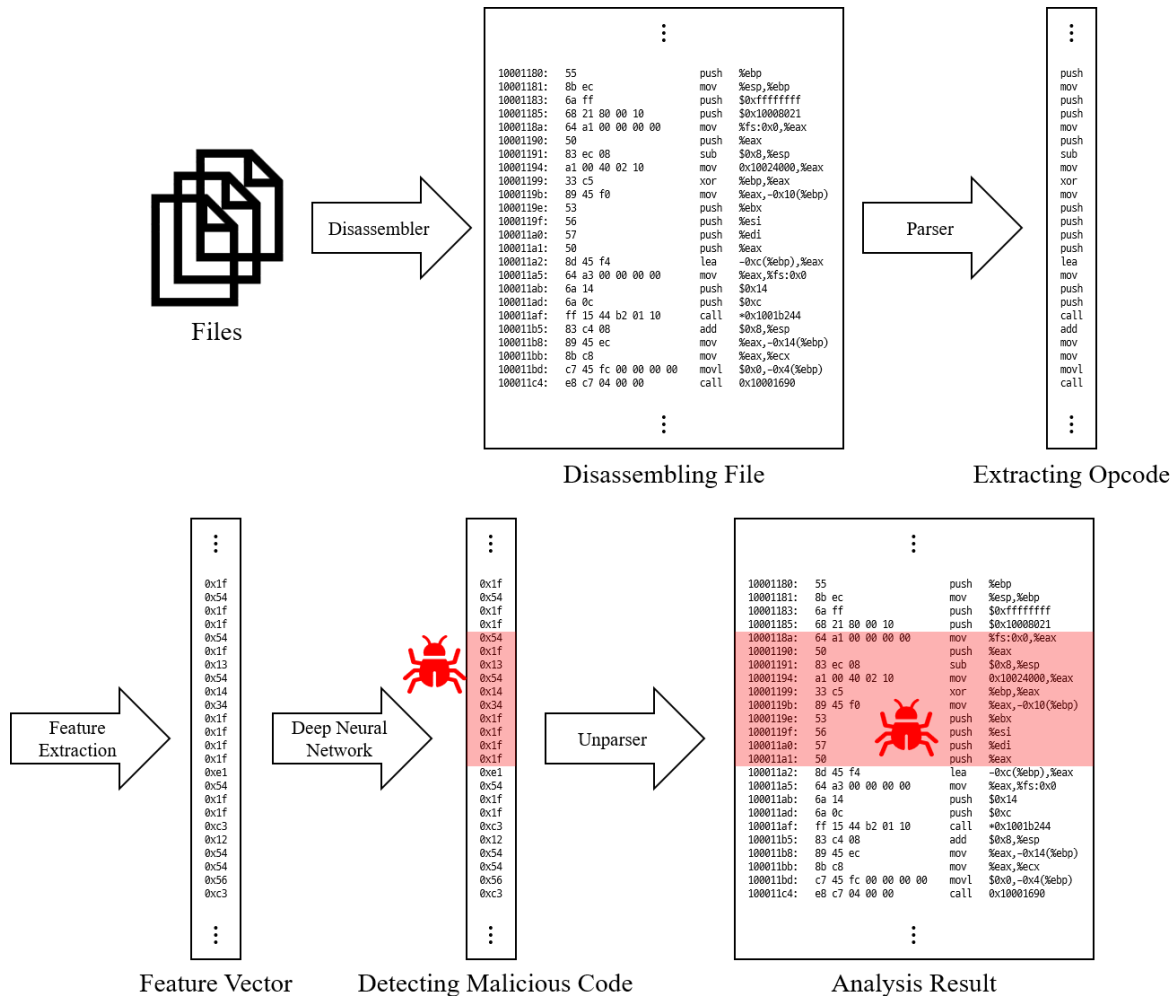


[그림 21] 기술개발의 최종목표

2.2 연구/개발 내용

제안된 소프트웨어의 개요는 [그림 22]과 같고, 소프트웨어 개발의 세부목표는 [표 2]와 같이 크게 세 단계로 나눌 수 있다. 첫 번째 단계는 정상코드 및 악성코드 파일 데이터셋을 확보하고, opcode를 추출하는 것이다. 두 번째 단계는 추출된 opcode를 신경망이 학습할 수 있도록 적절한 특징 벡터 공간으로 변환하는 것이다. 세 번째 단계는 특징 벡터를 적절하게 학습할 수 있는 신경망을 학습 및 튜닝하고, 본 서비스를 제공할 수 있는 서비스를 구축하는 것이다. 본 절에서는 각 세부 단계의 실험 방안과, 단계별 결과물을 제시한다.


 <div> 국민대학교 컴퓨터공학부 캡스톤 디자인 I </div>	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25



[그림 22] 제안된 소프트웨어의 분석 방법 개요

[표 2] 기술개발의 세부목표

1 단계 연구 목표	
- 정상코드 및 악성코드 파일 데이터셋 확보	
- Opcode 추출 알고리즘 개발	
2 단계 연구 목표	
- 딥러닝 기반 opcode 특징 벡터 추출 도구 개발	
3 단계 연구 목표	
- 딥러닝 기반 이상탐지 알고리즘 확보	
- 딥러닝 기반 자동 악성행위 코드 추출 도구 개발	
- 악성코드 분석 보조도구 제공을 위한 서버 구축	

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

2.2.1 데이터셋 확보 및 opcode 추출

신경망의 성능을 높이기 위해서는 양질의 데이터셋을 확보하는 것이 중요하다. 우리는 정상코드 및 악성코드 파일을 얻기 위해 Microsoft의 Kaggle 프로젝트 malware prediction의 데이터셋 [16], 한국인터넷진흥원의 정보보호 R&D 데이터 챌린지의 데이터셋 [17], 등을 이용한다. 우리의 연구에서는 양질의 정상코드 파일을 충분히 많이 확보하는 것이 중요하므로 웹 크롤러(web crawler)를 개발하여 웹으로부터 정상코드 파일을 얻을 수 있어야 한다.

본 연구의 목표는 opcode 중 악성행위를 수행하는 부분을 추출하는 것이므로 정상파일과 악성파일로부터 opcode를 추출하는 역어셈블러와 역어셈블된 결과 중 opcode를 추출하는 파서가 필요하다. 우리는 역어셈블러로 IDA Pro를 이용하고, 파서는 직접 개발하여 사용한다.


[표 3] 1 단계 결과물 목록

1 단계 결과물
<ul style="list-style-type: none"> - 정상파일을 수집하는 웹 크롤러 - Assembly code로부터 opcode를 추출하는 파서 - 정상코드 및 악성코드 파일 데이터셋

2.2.2 딥러닝 기반 opcode 특징 벡터 추출 도구 개발

신경망의 학습이 적절히 이루어지기 위해서는 로우 데이터(raw data)를 적절한 특징 벡터 공간으로 변환시키는 것이 필요하다. Opcode는 자연어와 유사한 시계열적 특징을 가지고 있으므로 자연어 처리에 사용되는 변환 방법을 사용하면 신경망의 성능을 높일 수 있다. 이때 특징 벡터 공간의 크기가 너무 크면 신경망 학습이 어려운 차원의 저주(curse of dimension) 문제가 발생하므로 선행 연구에서는 특징 벡터 공간의 크기를 줄이기 위해 다양한 방법을 사용했다. [3]에서는 파일에서 10번 이상 나오는 명령어는 one-hot code로 변환하였고, 그렇지 않은 명령어는 unknown에 대응되는 벡터로 변환했다. [4]에서는 정보이득(information gain, IG)이 0.3 초과인 opcode만을 학습에 사용했으며, 차원의 수를 줄이기 위해 [18]에서 사용되는 단어 임베딩(word embedding) 방법을 적용했다. [9]에서는 각 opcode를 가장 많이 사용된 순으로 정렬했을 때, 256번째 이후 명령어들은 하나의 opcode로 취급해 one-hot code로 변환하였다.

우리는 이 단계에서 [9]의 방법, word2vec, FastText 세 가지 방법에 대한 특징벡터 추출 방법을

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

개발한다. 이후 3 단계 진행 후 실제 데이터셋과 동일 신경망에 세 가지 데이터셋을 학습시켰을 때의 결과를 비교해 가장 적합한 추출 방법을 결정한다.

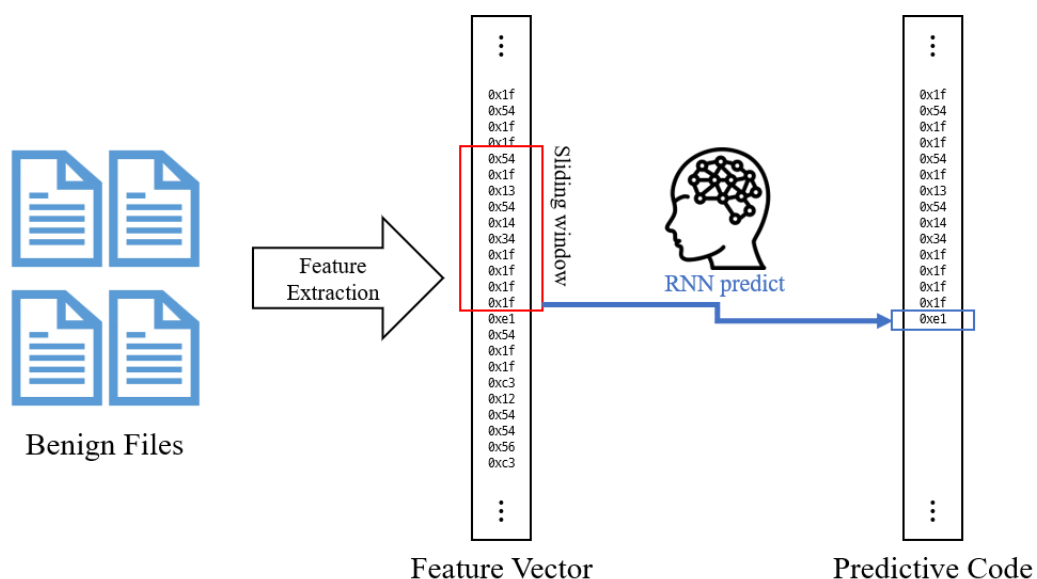
[표 4] 2 단계 결과물 목록

2 단계 결과물
<ul style="list-style-type: none"> - 단어 임베딩 소프트웨어 (word2vec, FastText, [9]의 방법) - 1 단계의 데이터셋에 대한 특징 벡터


2.2.3 딥러닝 기반 자동 악성행위 코드 추출 도구 개발

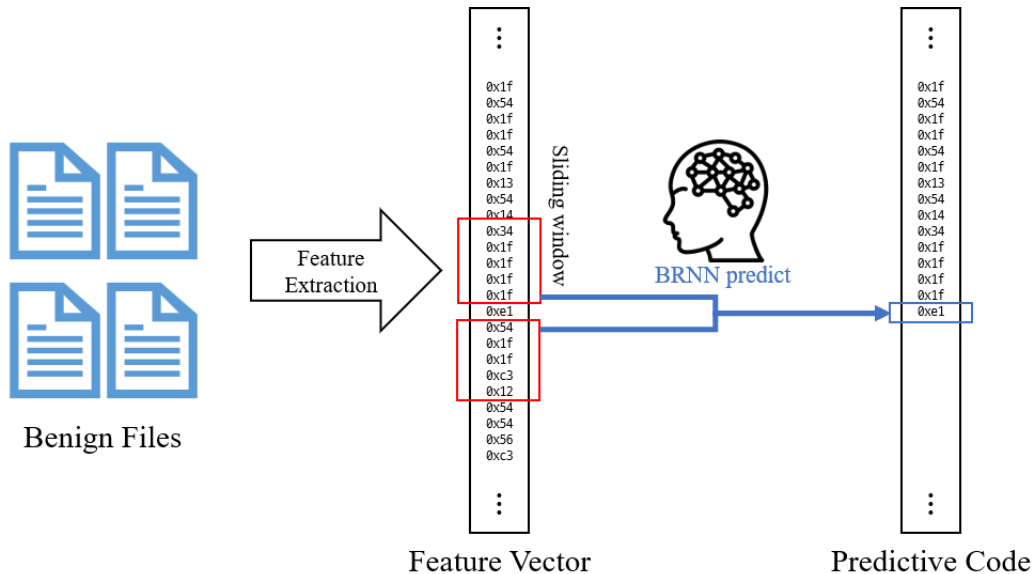
자연어 처리 연구에서 RNN을 기반으로 언어모델을 학습하여 문장의 문법 오류를 높은 정확도로 분류할 수 있었다. 우리는 이를 기반으로 정상코드의 명령어의 언어모델을 학습하면, 악성코드 중 악성행위를 수행하는 부분을 추출할 수 있음을 밝힌다. [3]에서 정상코드의 명령어 분포와 악성코드의 명령어 분포가 다를 것을 보였으므로, 본 프로젝트의 수행 결과 악성행위 코드를 높은 정확도로 추측할 수 있을 것으로 기대된다.

우리는 정상코드의 opcode에 대해 이전 opcode들로 다음 opcode를 추측하는 방식과([그림 23]), 주변 opcode들로 중심 opcode를 추측하는 방식([그림 24]), 두 가지 방식에 대해 실험을 수행하여 최적의 신경망 학습 방안을 찾는다.



[그림 23] 이전 명령어들로 다음 명령어를 예측하는 RNN 학습 방안


 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

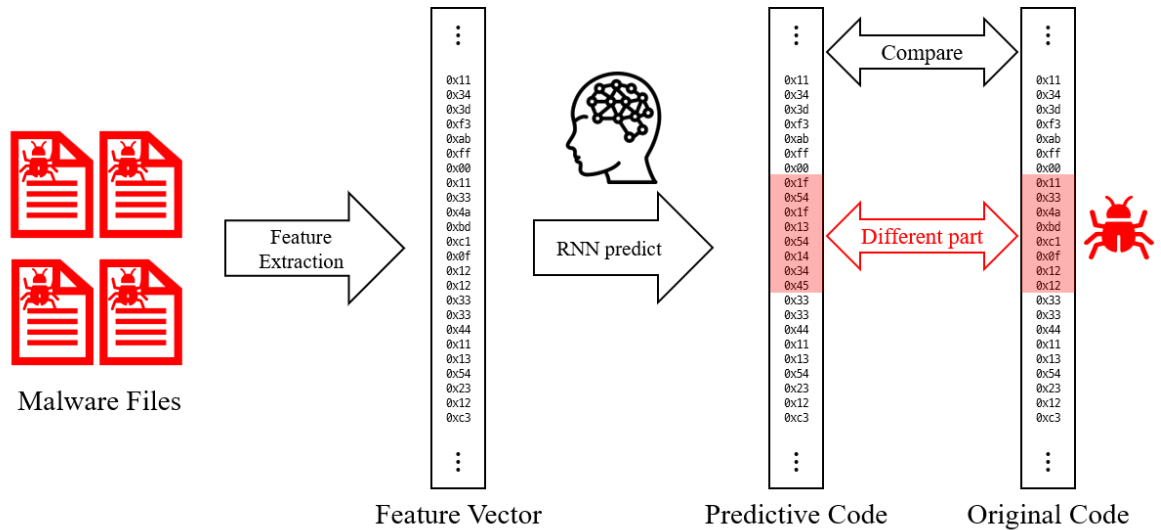


[그림 24] 주변 명령어들로 중심 명령어를 예측하는 RNN 학습 방안

이때 고려해야 할 하이퍼 파라미터로는 크게 1) 윈도우 크기, 2) 중심 명령어의 위치 (방안 2의 경우), 3) 신경망의 종류가 있다. 신경망은 vanilla RNN, LSTM, GRU 세 가지 종류의 RNN을 학습한 결과를 비교하고, 앞의 두 파라미터 중 최적의 파라미터를 실험적으로 찾는다.

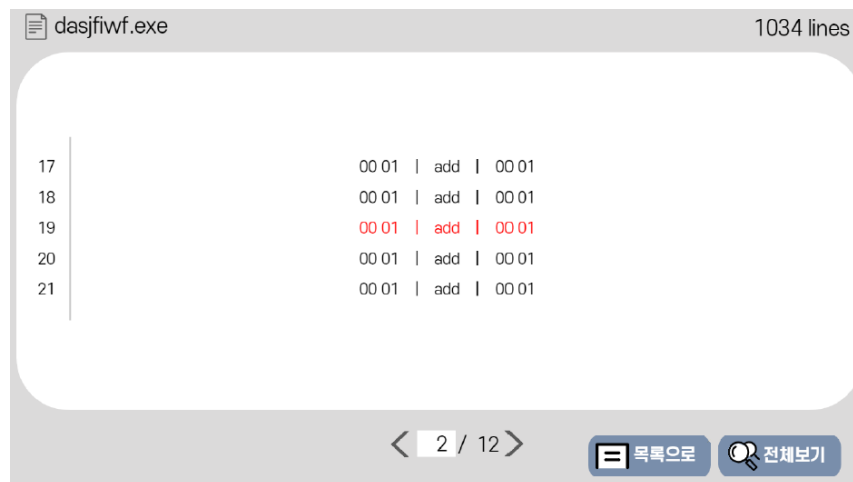
신경망의 학습이 완료되었다면, [그림 25]와 같이 악성코드의 opcode를 신경망에 입력하여 예측 코드(predictive code)를 추출한다. 신경망의 출력과 원래 코드를 비교했을 때, 신경망이 예측을 제대로 하지 못한 코드는 정상파일의 언어모델과 다른 분포를 갖는 코드이므로 악성행위를 수행하는 코드라 추측할 수 있다. 이때 우리의 목표는 악성행위를 수행하는 코드를 포함하는 의심영역을 찾는 것이므로 2종 오류(false negative)의 발생이 적어야 하며, 1종 오류(false positive)의 발생률은 상대적으로 클 수 있다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25




[그림 25] 악성행위 코드 추출 방안

우리가 제안하는 시스템을 많은 사용자가 쉽게 접근하고, 결과를 직관적으로 확인할 수 있도록 만들기 위하여 웹 서버를 구축한다. 사용자가 악성으로 의심되는 파일을 업로드하면, 신경망이 [그림 22]과 같은 방법으로 코드를 분석한 뒤, [그림 26]과 같이 악성으로 의심되는 부분을 요약하여 시각화한다.



[그림 26] 웹 기반 분석 결과 시각화 안

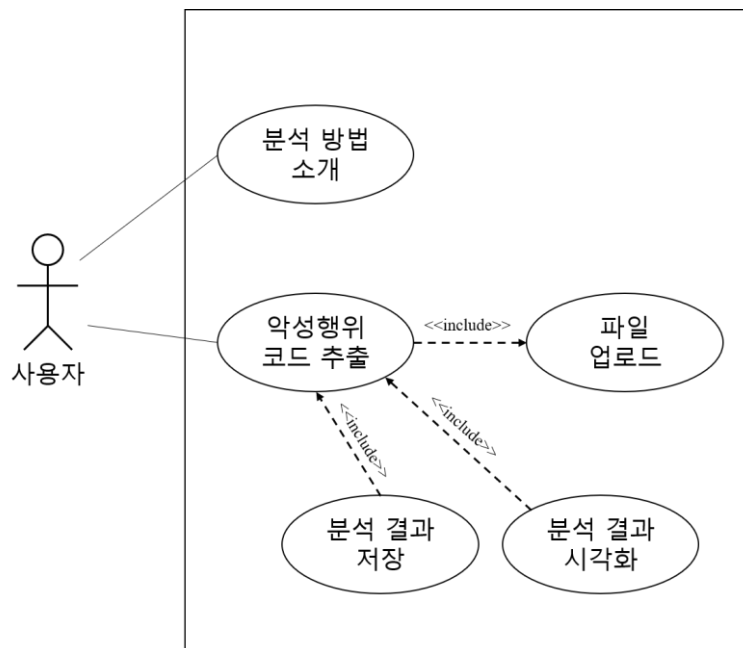
 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

[표 5] 3 단계 결과물 목록

3 단계 결과물
<ul style="list-style-type: none"> - 각 하이퍼 파라미터별 실험 결과 - 최적화된 신경망 모델 - 자동 악성행위 코드 추출 도구 - 분석 파일 업로드 및 결과 시각화를 위한 웹


2.3 개발 결과

2.3.1 시스템 기능 요구사항



[그림 27] 제안된 소프트웨어의 use case diagram

개발된 소프트웨어는 파일 중 사용자에게 악성행위 코드를 추출하는 방법론을 설명하는 기능이 있어야 한다. 사용자가 파일을 분석하기 위해 파일을 직관적인 UI(user interface)로 업로드 할 수 있어야 하며, 업로드된 파일을 분석한 결과를 웹에서 확인할 수 있어야 한다. 이때 악성행위를 수행하는 코드가 파편화되어 있는 경우를 대비하여 탐지된 악성행위 코드를 목록화하여 사용자가 쉽게 결과를 확인할 수 있어야 한다. 분석된 결과를 사용자가 파일로 저장할 수 있어야 한다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

2.3.2 시스템 비기능(품질) 요구사항

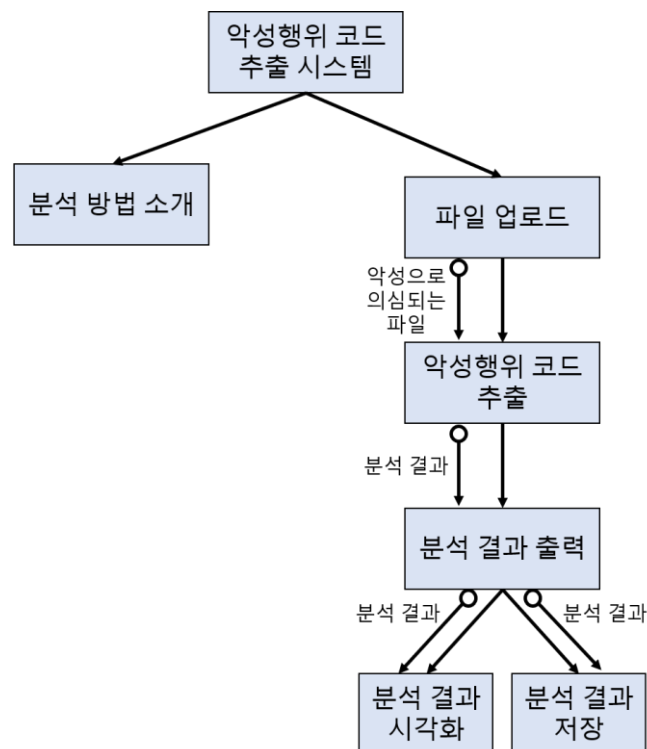
1. 보안

악의적인 사용자가 대용량 파일을 반복하여 업로드하면 서비스가 지연되거나 서버가 다운되는 문제가 발생할 수 있다. 따라서 동일한 IP 에서 1 시간 내에 업로드 할 수 있는 파일의 수는 10 개, 파일의 크기는 300MB 로 제한한다.

2. 성능


입력된 파일의 각 opcode별 악성행위 여부를 빠르고 정확하게 판단할 수 있어야 한다. 제안된 소프트웨어는 1초에 2,000개 이상의 opcode를 분석할 수 있어야 하며, 분석 결과는 전문가가 악성 행위라 판정한 코드의 70% 이상을 포함하고 있어야 한다.

2.3.3 시스템 구조



[그림 28] 제안된 소프트웨어의 시스템 구조도

제안된 소프트웨어는 웹 서비스로 제공된다. 분석 방법 소개는 우리가 제안한 소프트웨어의 방법론과 사용방법 기술한 문서를 보인다. 사용자가 웹에 파일을 업로드한 뒤, 분석을 요청하면

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

[그림 22]의 방법으로 악성행위를 수행할 것으로 의심되는 코드를 추출한다. 추출된 결과는 웹으로 전송되어 분석 결과를 시각화 하거나 저장할 수 있다.


2.3.4 결과물 목록 및 상세 사양

[표 6] 결과물 목록 및 상세 사양

대분류	소분류	기능	형식	비고
웹 서비스	<i>레이아웃</i>	직관적인 UI 를 제공한다.		
	<i>파일 업로드</i>	사용자가 악성이라 의심되는 파일을 업로드한다.	DLL/함수	
	<i>결과 페이지</i>	분석한 결과를 시각화하여 사용자가 악성행위 코드를 쉽게 확인하고, 결과를 저장할 수 있도록 한다.		
악성행위 코드 추출기	<i>단어 임베더</i>	입력된 opcode 를 적절한 특징 벡터로 변환한다.	함수	
	<i>RNN</i>	입력된 특징벡터를 분석하여 악성행위를 수행할 것으로 의심되는 opcode 를 추출한다.	모듈	
데이터 수집	<i>크롤러</i>	정상파일을 수집하여 데이터셋을 구축한다.	함수	

2.4 기대효과 및 활용방안


제안된 소프트웨어가 성공적으로 개발된다면, 전문가가 더 빠르게 악성코드를 분석할 수 있으므로 1) 더 적은 인력으로 더 많은 악성 코드를 분석할 수 있고, 2) 새로운 보안위협에 대해 빠르게 대응할 수 있다. 특히, 시큐리티대응센터와 같이 신속하게 신종 악성코드를 분석해야 하는 경우 큰 도움이 될 것으로 기대된다. 악성코드 분석 입문자는 제안된 소프트웨어를 활용해 악성코드의 패턴과 분석 노하우를 학습할 수 있으므로 보안 전문가의 양성에 도움이 될 것으로 기대된다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

3 배경 기술

3.1 기술적 요구사항

- 개발 환경
 - CPU: Intel Xeon E3-1230 v3
 - RAM: 24 GB
 - GPU: NVIDIA GeForce GTX 750
 - OS: Windows 10
- 개발 언어 및 라이브러리
 - Python 3.8.2
 - Numpy 1.17.3
 - Tensorflow 2.0.0
 - Pandas 1.0.2
 - Keras 2.3.0
 - Flask 1.1.1
 - React 16.13.0
- 프로젝트 결과물 확인 환경
 - CPU: AMD Ryzen 7 3700U
 - RAM: 8 GB
 - OS: Windows 10
- 오픈소스 소프트웨어
 - GNU objdump 2.28

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

3.2 현실적 제한 요소 및 그 해결 방안

3.2.1 실행파일의 목적 시스템


입력 차원의 크기가 크면 특징공간의 차원이 증가하므로 차원의 저주 문제가 발생할 수 있어 학습에 악영향을 미칠 수 있다. 다양한 목적기계 및 운영체제를 대상으로 컴파일된 파일을 학습 데이터 집합으로 사용하는 경우, 가능한 어셈블리어 집합의 크기가 증가하므로 학습 성능이 하락할 수 있다. 우리는 이를 해소하기 위해 하나의 목적기계 및 운영체제를 대상으로 컴파일된 파일을 분류한 뒤, 해당 파일에 대해서만 학습을 수행한다. 우리의 신경망이 효과적으로 학습된다면, 각 목적기계별 신경망을 학습시키거나, 통합된 신경망을 개발하는 등으로 추가 연구의 가능성을 제시할 수 있다.

3.2.1 패킹(packing)

실행 파일(executable file)로부터 원본 코드를 복구하는 역공학(reverse engineering)을 방지하거나, 파일의 크기를 줄이기 위하여 실행 파일을 압축하는 패킹(packing) 기법이 개발되고 있다. 특히, 악성코드의 경우 자동 분석기의 분석을 피하거나 전문가의 분석을 어렵게 만들기 위하여 패킹을 적용하는 경우가 많다. 패킹이 적용된 경우 IDA와 같은 역어셈블링 도구를 이용해 opcode를 복구하기 어려우므로 우리의 기법을 적용할 수 없다. 패킹된 파일을 역어셈블링하는 것은 우리의 연구범위를 벗어나므로, 우리는 패킹되지 않은 파일에 대해서만 학습 및 검증을 수행한다.

3.2.2 하드웨어


RNN은 다른 신경망 구조에 비해 학습에 많은 시간이 소요된다. 이를 완화하기 위해서는 고사양 GPU(graphics processing unit), CPU(central processing unit)와 고용량 RAM(random access memory)가 탑재된 시스템에서 학습을 수행해야 한다.

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

4 프로젝트 팀 구성 및 역할 분담

[표 7] 팀원별 역할


이름	역할
손현기	<ul style="list-style-type: none"> - Software Project Leader - 정상파일 크롤러 개발 - Opcode 파서 개발 - 신경망 구현 - 웹 프론트엔드 개발 - 웹 백엔드 개발
김주환	<ul style="list-style-type: none"> - 논문 동향조사 - 제안서 및 보고서 작성 - 신경망 구현 및 튜닝
김호준	<ul style="list-style-type: none"> - 자료조사 - 문서작업 보조 - 웹 프론트엔드 개발
오예린	<ul style="list-style-type: none"> - 디자인 - 웹 UI(user interface)/UX(user experience) 기획 - 웹 프론트엔드 개발
이동윤	<ul style="list-style-type: none"> - 정상파일 크롤러 개발 - 신경망 구현
Ruslan	<ul style="list-style-type: none"> - Opcode 파서 개발 - 웹 프론트엔드 개발

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

5 프로젝트 비용

[표 8] 프로젝트 비용

항목	예상치 (MD)
주제 구상 및 동향조사	45
서류 작성 (제안서, 중간보고서, 결과보고서)	50
개발 환경 구축	12
특징 추출기 및 신경망 구현, 최적화	100
웹 프론트엔드 개발 및 서버 구축	60
프로젝트 테스트	40
합	307


 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

6 개발 일정 및 자원 관리

6.1 개발 일정

[표 9] 개발 일정


항목	세부내용	1 월	2 월	3 월	4 월	5 월	6 월	비고
요구사항분석	요구 분석							
	SRS 작성							
관련분야연구	답러닝 기술 연구							
	관련 논문 동향조사							
설계	시스템 설계							
구현	코딩 및 모듈 테스트							
테스트	시스템 테스트							

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

6.2 일정 별 주요 산출물

[표 10] 일정 별 주요 산출물


마일스톤	개요	시작일	종료일
계획서 발표	개발 환경 구축 (Jupyter lab 및 딥러닝 라이브러리 설치) 산출물 : <ol style="list-style-type: none"> 1. 프로젝트 수행 계획서 2. 계획서 발표 자료 	2020-03-05	2020-03-26
설계 완료	시스템 설계 완료 산출물 : <ol style="list-style-type: none"> 1. 시스템 설계 사양서 	2020-03-27	2020-04-03
중간 보고	프로그램의 기본 기능 구현 완료 (크롤러, 파서, 단어 임베딩, 신경망) 산출물 : <ol style="list-style-type: none"> 1. 프로젝트 중간 보고서 2. 프로젝트 진도 점검표 3. 1 차분 구현 소스 코드 	2020-04-04	2020-04-23
구현 완료	시스템 구현 완료 산출물: 악성행위 코드 추출 프로그램	2020-04-24	2020-05-22
테스트	시스템 통합 테스트 산출물: 악성행위 코드 추출 프로그램 및 웹	2020-05-23	2020-06-04
최종 보고서	최종 보고 산출물: 최종 보고서, 프로젝트 소개 책자, 포스터	2020-06-04	2020-06-11

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

6.3 인력자원 투입계획

[표 11] 인력자원 투입계획


이름	개발항목	시작일	종료일	총개발일(MD)
전원	프로젝트 주제 구상	2019-12-14	2020-02-03	30
김주환	악성코드 논문 동향 조사	2020-01-28	2020-02-15	15
김주환, 오예린	제안서 작성	2020-03-01	2020-03-18	15
손현기, 이동윤	웹 크롤러 제작 및 데이터셋 수집	2020-03-01	2020-03-10	7
손현기, Ruslan	Opcode 파서 제작	2020-03-10	2020-03-15	5
김주환, 손현기, 이동윤	특징 추출기 및 신경망 구현, 하이퍼 파라미터에 따른 신경망 성능 비교	2020-03-15	2020-05-22	100
손현기	웹서버 구축	2020-03-15	2020-05-22	30
오예린, Ruslan, 김호준	웹 UI 기획 및 웹 프론트엔드 개발	2020-03-15	2020-05-22	30
전원	프로젝트 테스트	2020-05-16	2020-06-04	40
김주환, 오예린	중간보고서 작성	2020-04-10	2020-04-22	15
김주환, 오예린	결과보고서 작성	2020-05-23	2020-06-09	20

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

6.4 비 인적자원 투입계획


[표 12] 비 인적자원 투입계획

항목	Provider	시작일	종료일	Required Options
머신러닝용 PC 2 대	조립 PC	2020-03-27	2020-05-22	GPU
개발용 노트북 4 대	미정	2020-03-27	2020-05-22	
웹 서버용 PC 1 대	조립 PC	2020-03-27	2020-06-30	

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

7 참고 문헌

번호	종류	제목	출처	발행년도	저자	기타
[1]	기사	Malware Statistics	https://www.av-test.org/en/statistics/malware/	2020	AV Test	
[2]	논문	An overview of anomaly detection techniques: Existing solutions and latest technological trends	Computer networks. Vol. 51. No. 12.	2007	Animesh Patcha and Jung-Min Park	
[3]	논문	Malware Detection with Deep Neural Network Using Process Behavior	IEEE 40th Annual Computer Software and Applications Conference	2016	Shun Tobiyama, Yukiko Yamaguchi, Hajime Shimada, Tomonori Ikuse, Takeshi Yagi	
[4]	논문	A deep Recurrent Neural Network based approach for Internet of Things malware threat hunting	Future Generation Computer Systems 85	2018	Hamed HaddadPajouh, Ali Dehghantanha, Raouf Khayami, Kim-Kwang Raymond Choo	
[5]	논문	정적 분석 기반 기계학습 기법을 활용한 악성코드 식별 시스템 연구	한국정보보호학회 Vol. 29. No. 4	2019	김수정, 하지희, 오수현, 이태진	
[6]	논문	정적 분석과 양상불 기반의 리눅스 악성코드 분류 연구	한국정보보호학회 Vol. 29. No. 6	2019	황준호, 이태진	
[7]	논문	악성코드 DNA 생성을 통한 유사 악성코드 분류기법	한국정보보호학회 Vol. 23. No. 4.	2013	한병진, 최영한, 배병철	
[8]	논문	Convolutional Neural Network 기반의 악성코드 이미지화를 통한 패밀리 분류	한국정보보호학회 Vol. 26. No. 1.	2016	석선희, 김호원	
[9]	논문	Deep Learning and Visualization for Identifying Malware Families	IEEE Transactions and Dependable and Secure Computing	2018	Guosong Sun, Quan Qian	
[10]	논문	악성코드 패밀리 분류를 위한 API 특징 기반 양상불 모델 학습	한국정보보호학회 Vol. 29. No. 3.	2019	이현종, 어성율, 황두성	
[11]	기사	드라마 ‘유령’ 속 악성코드, 실제로는?	https://www.ahnlab.com/kr/site/securityinfo/secuNews/secuNewsView.do?cmd=print&seq=19768&menu_dist=3	2012	AhnLab	
[12]	보고서	2018년 정보보호 실태조사	https://www.msit.go.kr/web/msipContents/contentsView.do?cateId=_status&artId=1513388	2018	과학기술정보통신부, 한국정보보호산업협회	
[13]	보고서	Malware Analysis Market by Component (Solution (Static Analysis and Dynamic Analysis) and Services), Organization Size (SMEs and Large	https://www.marketsandmarkets.com/Market-Reports/malware-analysis-market-	2019	Markets and Markets	

 국민대학교 컴퓨터공학부 캡스톤 디자인 I	계획서		
	프로젝트 명	asi(a security insight)	
	팀 명	assist(A Security Safety Important Special Team)	
	Confidential Restricted	Version 1.4	2020-MAR-25

		Enterprises), Deployment (Cloud and On-premises), Vertical, and Region - Global Forecast to 2024	108766513.html			
[14]	사이트	ASEC 소개	https://www.ahnlab.com/kr/site/securityinfo/asec/asecIntro.do?	2020	AhnLab	
[15]	사이트	ESRC 소개	https://www.estsecurity.com/	2020	EST Security	
[16]	사이트	Microsoft Malware Prediction	https://www.kaggle.com/microsoft-malware-prediction	2018	Kaggle	
[17]	사이트	정보보호 R&D 데이터 챌린지 2019	http://datachallenge.kr/challenge19/rd-datachallenge/malware/introduction/	2019	한국인터넷진흥원	
[18]	논문	A Neural Probabilistic Language Model	Journal of Machine Learning Research 3	2003	Yoshua Bengio, Rejean Ducharme, Pascal Vincent, Christian Jauvin	
[19]	사이트	VirusTotal homepage	https://www.virustotal.com/gui/home	2020	VirusTotal	
[20]	사이트	Intezer homepage	https://intezer.com/	2020	Intezer	
[21]	사이트	V3 소개	https://www.ahnlab.com/kr/site/product/productView.do?prodSeq=15	2020	AhnLab	
[22]	사이트	What is Cuckoo	https://cuckoo.readthedocs.io/en/latest/introduction/what/	2020	Cuckoo Sandbox	
[23]	사이트	Joe Sandbox Cloud	https://www.joesecurity.org/joe-sandbox-cloud#overview	2020	Joe Security	