JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 19, No. 2, pp. 363-370, Feb. 2018

다이나믹 API 호출 흐름 그래프를 이용한 오프라인 기반 랜섬웨어 탐지 및 분석 기술 개발

강 호 석[1] · 김 성 열[2*]
[1]건국대학교 유비쿼터스정보기술연구원
[2]건국대학교 소프트웨어학과

Offline Based Ransomware Detection and Analysis Method using Dynamic API Calls Flow Graph

Ho-Seok Kang[1] · Sung-Ryul Kim[2*]
[1]Institute of Ubiquitous Information Technology and Application (UBITA), Konkuk University, Seoul 05029, Korea
[2*]Department of Software, Konkuk University, Seoul 05029, Korea

[요 약]

최근 랜섬웨어 탐지는 디지털 콘텐츠 보호를 위한 컴퓨터 보안 분야에서 중요한 주요한 이슈가 되고 있다. 그러나 불행하게도 현재 시그니쳐 기반이나 정적 탐지 모델의 경우 압축 및 암호화 등의 기법을 이용하여 탐지를 피해갈 수 있다. 이를 극복하기 위해 본 논문에서는 RF, SVM, SL, NB 알고리즘 같은 데이터 마이닝 기법을 이용한 다이나믹 랜섬웨어 탐지 시스템을 제안하였다. 이 기법은 실제 소프트웨어를 구동 시켜 동작 행위를 추출해 API 호출 흐름 그래프를 만들고 그 특징을 분석에 이용하였다. 그 후 데이터 정규화, 특징 선택 작업을 진행하였다. 우리는 이러한 분석과정을 더욱더 개선 시켰다. 마지막으로 데이터 마이닝 알고리즘을 적용시켜 랜섬웨어인지를 판별하였다. 제안한 알고리즘의 성능 측정을 위해 더 적합한 추가 샘플 랜섬웨어 데이터를 수집하여 실험하였고 탐지성능이 향상되었음을 보여주었다.

[Abstract]

Ransomware detection has become a hot topic in computer security for protecting digital contents. Unfortunately, current signature-based and static detection models are often easily evadable by compress, and encryption. For overcoming the lack of these detection approach, we have proposed the dynamic ransomware detection system using data mining techniques such as RF, SVM, SL and NB algorithms. We monitor the actual behaviors of software to generate API calls flow graphs. Thereafter, data normalization and feature selection were applied to select informative features. We improved this analysis process. Finally, the data mining algorithms were used for building the detection model for judging whether the software is benign software or ransomware. We conduct our experiment using more suitable real ransomware samples. and it's results show that our proposed system can be more effective to improve the performance for ransomware detection.

색인어 : 데이터 분석, 랜섬웨어 탐지, API 호출 흐름 그래프, 데이터 마이닝, 컴퓨터 보안
Key word : Data Analysis, Ransomware Detection, API CFG (Calls Flow Graph), Data Mining, Computer Security

http://dx.doi.org/10.9728/dcs.2018.19.2.363

Received 14 February 2018; **Revised** 26 February 2018
Accepted 27 February 2018

***Corresponding Author; Sung-Ryul Kim**

Tel: +82-2-450-4134
E-mail: kimsr@konkuk.ac.kr

Ⅰ. 서 론

인터넷이 널리 퍼지면서 악성 소프트웨어(Malware)의 피해는 점점 늘어나고 있다. 특히 최근에는 랜섬웨어(Ransomware)라고 불리는 특별한 형태의 악성 소프트웨어가 사이버 보안 분야에서 등장하였다.

랜섬웨어는 악성 소프트웨어의 하나의 종류로 피해자 데이터의 접근을 막고 파일을 변형시켜 원상 복구의 대가로 돈을 요구한다. 랜섬웨어에는 두 가지 형태가 있다. 첫째로는 라커-랜섬웨어 (locker-ransomware) 이다. 이는 피해자의 컴퓨터를 사용할 수 없게 만드는 경우이다. 두 번째는 대부분의 현재 발생되는 랜섬웨어로 doc, jpg, pdf 등의 개인파일들을 암호화하여 피해자로부터 접근할 수 없게 만드는 방법이다. 특히 크기가 작은 디지털 콘텐츠 파일의 경우 백업을 통해 복구할 수 있는 가능성이 있지만 큰 디지털 콘텐츠의 경우 돈을 지급하는 방법 외에는 복구가 불가능한 경우가 많다.

랜섬웨어의 개념 자체는 1980년대 후반에 나왔지만, 현재 바이러스의 발전과 함께 다양한 램섬웨어가 등장하였다. 최근 SonicWall 과 같은 보안 팀의 보고에 따르면 2015년에 약 380만의 랜섬웨어 공격 시도가 있는 것으로 알려졌다. 전년도인 2014년 대비 19% 증가한 수치고 최근에는 더욱 많은 시도가 이루어지고 있다[1]. 이러한 공격들은 시그니쳐 기반의 메커니즘 으로는 쉽게 공격을 허용하고 있다. 또한 시그니쳐 기반의 탐지는 악성 소프트웨어의 수가 증가함에 따라 시그니쳐가 수가 기하급수적으로 늘어날 것이다. 더 나아가 현재 랜섬웨어의 기술은 탐지를 피하기 위한 여러 기술들이 접목되어 있다. 오늘날 행위 기반의 악성 코드 탐지를 위한 연구들은 다이나믹 분석을 사용한다. 특히 데이터 마이닝은 시그니쳐 기반의 방법을 대신하여 악성 소프트웨어를 탐지하는데 널리 적용되고 있다.

본 논문은 랜섬웨어를 탐지하기 위해 다이나믹 API 호출 흐름 그래프 (CFG; Call Flow Graph)를 기반으로 한 데이터 마이닝 방법을 제안하였다[2]. 본 연구의 목적은 높은 정확도를 가지고 랜섬웨어가 침투하기 전에 가능한 빨리 분류해 내는데 있다. 우리는 API 호출 들을 감시하여 특징들 (Features)을 추출하고 CFG를 만들었다. 그리고 랜섬웨어 식별의 정확성을 향상시키기 위하여 데이터 정규화 (Data Normalization)와 특징 선택 (Feature Selection)을 선행 작업으로 진행하였다. 그 후 RF (Random Forest), SVM (Support Vector Machine), SL (Simple Logistic), NB (Naive Bayes) 등의 네 가지 데이터 마이닝 알고리즘[3]을 이용하여 분류기 (Classifier)를 만들었다. 그리고 이렇게 만들어진 모델의 성능을 평가하기 위해서 탐지 정확도와 오탐지율 값을 실험을 통해 얻어냈다. 본 논문은 [2]의 기초 연구를 바탕으로 오류 수정, 알고리즘 및 실험 데이터 교체, 작업 프로세스 개선, 분석 설명을 보완하였다.

본 논문의 구성은 2장에서 관련연구, 3장에서 제안한 방법의 프레임워크, 그리고 4장에서 제안한 탐지 모델을 실험을 통해 성능을 측정하였고, 마지막으로 5장의 결론으로 구성된다.

Ⅱ. 관련 연구

악성 소프트웨어의 탐지와 분석을 위해 많은 종류의 방법이 개발되고 발전되어 왔다. 가장 널리 사용되는 방법은 시그니쳐 기반의 탐지 알고리즘이다[4]. 시그니쳐 탐지의 한계 때문에 이 방법은 새로운 악성 소프트웨어를 탐지하기 힘들다. 이러한 전통적인 시그니쳐 기반 모델의 단점을 보완하기 위해 많은 연구와 함께 의심스러운 악성 소프트웨어의 근본적인 행위에 대한 연구를 진행 하였다.

Jusuk[5] 등이 개발한 연구에서는 악성코드의 의미적인 특징을 얻기위해 API 호출 순서를 그래프로 변환하였다. 생성된 호출 그래프는 의미적인 특징을 찾아 코드 그래프로 변경하였다. 이 방법의 실제 악성 소프트웨어 탐지율은 91%의 정확도를 가지고 있다.

Fatemech[6] 등의 연구자는 현재 그래프 마이닝 기술을 기본으로 하여 악성 코드를 찾는 방법이다. 이 방법은 96.6%의 탐지 정확도와 3.4%의 오탐지율을 가지고 있다.

텍스트 범주화를 적용시킨 방법으로 Nir[7] 등의 연구자가 발표한 연구는 정적 분석 방법이다. 이 방법은 알려지지 않은 악성 코드를 탐지하는 방법을 정적 방법을 이용하여 개발하였다. 실험을 통해 97.83%의 탐지율을 기록했다.

Joshua[8] 등의 연구자는 딥 러닝 (Deep Learning) 기반의 악성 소프트웨어 탐지 방법을 소개하였다. 그리고 400,000이 넘는 소프트웨어 바이너리를 통한 실험을 통해 95%의 탐지율과 0.1%의 오탐지율을 기록하였다.

현재는 랜섬웨어 탐지를 목적으로 하는 방법에 대한 관련연구가 많지 않다. 2015년 Donghyun[9] 등의 연구자가 발표한 정량적 모델은 디스크 드라이브의 암호화 수행을 탐지 및 방어 하는 방법이다. 2016년 Daniele[10] 등이 개발한 EleRan은 다이나믹 분석과 랜섬웨어 탐지를 위한 머신 러닝 방법이다. EleRan은 랜섬웨어의 특징적 표시를 검색하고 첫 설치 과정에서 해당 소프트웨어가 수행하는 것을 감시한다. Sanggeun[11] 등은 프로세스와 특별한 파일 디렉토리등을 감시하는 기술을 개발하였다. 이 방법은 정적 방법을 기반으로 프로세스 시간, 메모리 사용, I/O 사용주기 등을 관찰하여 비 이상적인 동작을 할 경우를 찾아낸다. 2017년, Amin[12] 등이 개발한 UNVEIL은 랜섬웨어를 찾고 분석을 할 수 있는 방법이다. 이 시스템은 파일 암호화나 컴퓨터 잠금 같은 전형적인 랜섬웨어 행위를 탐지한다. 그리고 [2]는 본 논문에 대한 기초 연구로 국제 학술대회에 발표한 랜섬웨어 탐지 방법이다.

Ⅲ. 시스템 프레임워크

3-1 시스템 소개

이번 장 에서는 본 논문에서 소개하는 시스템의 구조를 설명

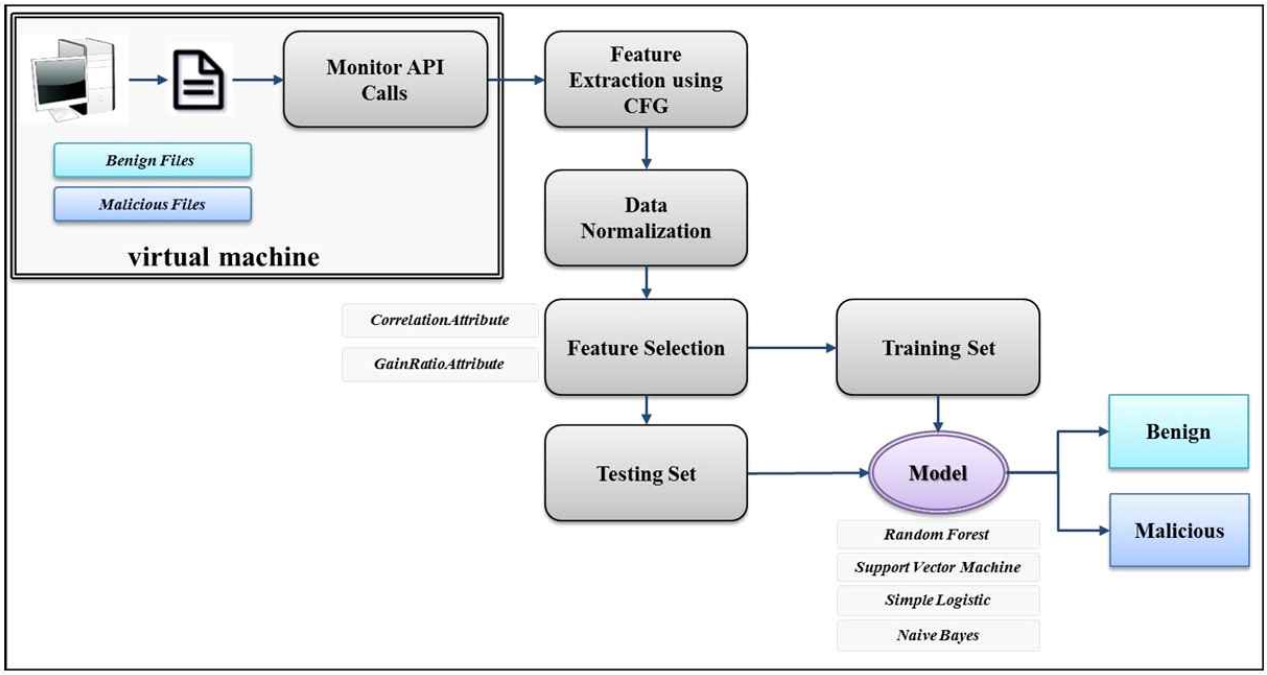

그림 1. 제안한 시스템 프레임워크
Fig. 1. Proposed System Framework

한다. 우리가 제안하는 탐지 시스템은 랜섬웨어 탐지 성능을 향상시키기 위해 데이터 마이닝 기법과 함께 API 호출 흐름 그래프를 사용한다. 제안한 시스템은 그림 1에와 같이 구성된다. 특이 이 시스템은 다음과 같은 네 가지 모듈로 구성 된다: (1) 특징 추출 (Feature Extraction) 모듈 , (2) 전 처리(Pre-Processing) 모듈, (3) 특징 선택 모듈, (4) 그림 1에 있는 기계학습 알고리즘과 훈련과 연습 세트를 포함한 모듈.

모듈 1: 시스템은 API 모니터 도구를 사용하여 API 호출 순서를 수집한다. 호출 순서는 소프트웨어가 실행될 때 표시되는 행위의 순서를 말한다.

모듈 2: 생성된 그래프를 특징 벡터로 변환하고 추 후 분석을 위해 레이블을 붙여 보관한다.

모듈 3: 특징 선택 기술을 이용하여 최소의 특징을 선택한다. 이 작업은 특징을 보관하는 공간을 줄여 성능을 향상시키고 다음 단계의 분석에서 계산 시간을 줄이기 위함이다.

모듈 4: 특징 선택 후 새로운 데이터를 이용하여 분류기를 훈련시킨다. 실제 예제 소프트웨어가 랜섬웨어인지 정상 소프트웨어인지를 판단하기 위해서는 훈련 과정을 거쳐야 한다.

3-2 API 호출 모니터링

API Monitor[13]는 API 모니터링 도구로서 소프트웨어와 서비스를 추적하고 API 호출을 탐지하고 저장할 수 있다.

우리의 실험을 위해 설치한 Microsoft 윈도우즈 7에 가상 환경을 구축하고 가상환경 안에서 API Monitor를 실행 하여 API 호출 순서를 저장하였다. 그리고 저장된 데이터는 가상환경 밖으로 가져와 분석을 위해 여러 단계에 거쳐 사용하였다. 그리고 다른 소프트웨어를 API Monitor 도구에서 실험할 경우 공정한 실험을 위해 가상환경을 복원하여 초기 상태로 만든 후 진행 하였다. 참고로 여기서 말하는 API는 실제 윈도우즈 API에 사용되는 함수들 뜻하지만 관례상 API로 칭한다.

3-3 CFG를 이용한 특징 추출

이번 단원부터는 각 모듈 1과 모듈 2에서 진행 되는 과정을 예제를 통해 설명한다.

그림 2.의 (a)는 API Monitor에서 추출한 API CFG 이다. API 호출 순서(Call Sequence) 보고를 *S*라고 가정하고 이 *S*는 그림

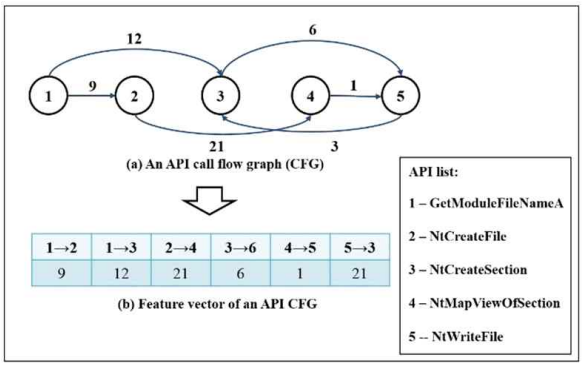

그림 2. 특징 선택과 데이터 표현
Fig. 2. Feature Extraction and Data Representation

2.의 (a)인 API 호출 순서들을 받는다. 5개의 API로 이루어진 예제의 CFG는 방향성을 가지고 서로 연결되어 있고 가중치를 가지고 있다. 이러한 CFG의 하나의 에지 (edge)는 API 들을 방향성에 맞게 이름을 부여하고 가중치로 표현할 수 있다. 이러한 가중치는 API_1 호출된 후 API_2가 호출된 빈도와 같다.

일반적으로 데이터 마이닝을 수행하기 전에 데이터를 처리하는 몇 가지 방법이 있는데 매우 중요한 단계이다. 그 중 가장 첫 단계가 정규화와 특징 선택이다[14]. 이 작업을 수행하기 위해서 우선 CFG 특징들을 특징 벡터 (Feature Vector)로 바꿔야 한다.

그림 2의 (a)와 같이 다섯 API (API_1, API_2, API_3, API_4, API_5)를 소프트웨어로부터 추출했다고 가정한다.

벡터로 변환된 그림 2의 (b)는 API 호출 순서 ($API_1 \rightarrow API_2$, $API_1 \rightarrow API_3$, $API_2 \rightarrow API_4$, $API_3 \rightarrow API_6$, $API_4 \rightarrow API_5$, $API_5 \rightarrow API_3$)와 빈도를 가지고 있다. 예를 들어 $API_1 \rightarrow API_2$ 는 API 호출 순서가 총 9번 나타났음을 알 수 있다. 그러므로 모든 API 흐름 그래프는 특징 벡터로 변환될 수 있다.

3-4 데이터 정규화

데이터 정규화는 데이터 마이닝에서 매우 중요한 단계이다. 이는 데이터의 다른 속성이 다른 단위와 스케일의 값을 가지기 때문이다. 몇몇 큰 속성 값은 다른 작은 속성 값보다 더욱 더 편향된 방향으로 결과 값을 나타낼 수 있기 때문이다. 그러므로 우리는 우리의 데이터를 정규화 하여 수행하였다.

데이터 정규화의 값은 다음과 같다[15]:

$$S'(x) = \frac{S(x)}{\|S(x)\|} \tag{1}$$

$\|S(x)\|$ 는 유클리드 노름 (Eucliean norm)을 나타낸다.

이 방법은 데이터의 크기를 조절하기 위한 방법이다. 각각의 속성의 값을 범위 [0,1] 사이의 값으로 바꾼다. 그리고 $d_{\min} = 0$ 부터 $d_{\min} = \sqrt{2}$ 사이의 범위 값 d 로 다시 설정 한다.

정규화 후에, 모든 훈련과 실험 데이터도 변환 시킨다. 그리고 이러한 정규화 과정은 우리가 선택한 분별기의 성능을 향상시킬 것이다.

3-5 특징 선택

탐지 모듈의 성능은 데이터 셋의 영향을 많이 받게 된다. 그러나 높은 차원의 데이터는 노이즈를 증가시키고 분별기가 복잡해지고 탐지 시간도 길어질 것이다. 그러므로 특징 선택은 데이터의 차원을 감소시키고 소프트웨어의 다양한 범주 사이에서 가장 좋은 특징을 선택할 수 있는 중요한 단계이다. 이 단계는 데이터 셋의 잡음과 불규칙한 특징을 제거해 시간 성능과 탐지 정확도를 높일 수 있다.

본 논문에서 우리는 특징으로 수행된 소프트웨어의 API들의 연속된 순서 (2-sequence)를 특징으로 추출하였다. 이러한 특징들의 개수는 매우 많아서 평균 73,202개의 특징을 가지고 있다. 그러므로 CORR (Correlation) 과 GR (Gain Ratio) 방법을 적용하여 특징선택을 수행 하여야 한다.

1) Correlation

CCOR[16]은 다른 속성 값들 사이에서 서로 관계있는 값들을 이용하여 속성 값의 가치를 평가하고 이를 이용하여 특징을 선택한다. CCOR은 Pearson Correlation Coefficient[17]에 의해 계산된다. 그러므로 데이터가 정규화 되어 있어야 한다.

$$r_{xy} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\ \sqrt{n\sum y_i^2 - (\sum y_i)^2}} \tag{2}$$

x와 y는 특징/속성을 나타낸다. 여기서 우리는 최댓값을 가지는 특징 m을 선택하기 위해 Pearson Correlation Coefficient (r_{xy}) 값들을 정렬 (sort) 시킨다. 이는 큰 r_{xy} 값을 가지는 특징들이 우리가 원하는 탐지 모델에서 더 중요한 정보와 연관이 높기 때문이다.

2) Gain Ratio

GR[18] 는 본질적인 정보를 위한 정보의 이득에 대한 비율이다. GR은 속성이 선택될 때 숫자 값과 연결된 크기 값이 다중 값을 가진 속성으로 치우치는 것을 감소시키는 역할을 한다.

$$GR(D,A) = \frac{IG}{IV} \tag{3}$$

D는 모든 트레이닝 데이터의 집합이고, A는 특징의 값, 그리고 IV는 고유 값을 나타낸다. 여기서 GR 값을 정렬시키고 최대가 되는 m 특징을 선택한다. 이는 탐지 모델에서 더 중요한 정보를 가지는 높은 이득 비율 값을 가진 특징을 선택하기 위해서이다.

3-6 탐지 모델

실험에서의 성능향상을 위하여, 우리는 Weka[19]에서 제공하는 서로 다른 데이터 마이닝 알고리즘을 이용하였다. Weka는 자바 (Java) 기반의 기계학습 프로그램 도구이다. 이 도구는 트레이닝과 테스트를 서로 다른 방법을 이용하여 만들고 실험할 수 있는 환경을 제공한다. 또한 잘 알려진 많은 기계학습 알고리즘을 미리 구현해놓고 이를 이용할 수 있다. 이론적인 접근 방법인 룰 기반 (Rule-based) 방법, 함수 기반 (Function-based) 방법, 확률 기반 (Probability-based) 방법 들을 선택할 수 있다. 그 중 우리는 RF, SVM, SL, NB 알고리즘을 선택하였고 이를 이용하여 랜섬웨어 분류기를 만들어 실험에 사용하였다.

Ⅳ. 실 험

본 논문에서 실험환경은 인텔의 Core i7-4790 CPU를 가진 머신 (3.60GHz 프로세서, 16 GB 메모리) 에서 수행하였다. 이 머신에는 Microsoft 윈도우즈 7을 설치하였다. 그리고 여기에 가상머신을 설치하여 실험을 수행하였다. 특히 소프트웨어를 30초 동안 실행 시켜 랜섬웨어 샘플을 분석하였다. 실행 시간을 둔 이유는 랜섬웨어에 가상머신이 감염되기 전의 특성을 찾기 위해서 대부분의 랜섬웨어가 감염되는 시간보다 적은 시간을 잡았다. 30초라는 시간은 머신의 성능, 운영체계, 가상환경, 모니터링 툴에 따라 다르므로 큰 의미를 두지 않아도 된다.

4-1 데이터 셋

비교를 위한 표준 데이터 셋이 없기 때문에 우리는 직접 83개의 다른 랜섬웨어 샘플을 수집했고, 이를 실험과 비교를 위해 정상적인 소프트웨어 샘플 85개와 섞어 사용하였다. 샘플들의 특징을 추출하고 제안한 시스템에 적용시키기 위해 트레이닝 테스트와 K 교차 검증 (Cross-Validation) 방법을 사용하였다. 정상적인 데이터 셋[20]은 윈도우즈 환경의 게임, 멀티미디어 도구, 뷰어, 브라우저 오피스, 스캐너, 개발 툴 등을 수집했다. 랜섬웨어샘플[21]은 CryptoWall, Kollah, Trojan-Ransom, TeslaCrypt등을 수집하였다. 표 1은 수집한 랜섬웨어 패밀리 종류와 샘플 수를 나타낸 표이다.

표 1. 랜섬웨어 데이터 셋
Table 1. Dataset of Ransomware samples

Ransomware Families	Number of Samples
CryptoWall	36
Kollah	17
Trojan-Ransom	24
TeslaCrypt	6

4-2 성능 측정

표 2. 성능측정을 위한 지표
Table 2. Parameters definition for performance estimation

Parameter	Definition
True Positive (TP)	Malicious code: O Judged as malicious code: O
False Positive (FP)	Malicious code: X Judged as malicious code: O
True Negative (TN)	Malicious code: X, Judged as malicious code: X
False Negative (FN)	Malicious code: O, Judged as malicious code: X

보통 침입탐지 시스템과 같은 보안도구의 성능을 평가하기 위해서는 정확도 (Accuracy), 정탐지율 (TPR; True Positive Ratio, TNR; True Negative Ratio), 오탐지율 (FPR; False Positive Ratio), 미탐지율 (FNR; False Negative Ratio)을 측정한다[21].

위 식을 구하기 위해서 표2와 같은 성능지표를 살펴볼 필요가 있다. 실제 악성코드와 그를 탐지하는 관계이지만 악성코드를 랜섬웨어로 보고 설명하면, TP는 랜섬웨어를 랜섬웨어로 판단하는 숫자를 나타내고, FP는 랜섬웨어가 아닌 경우를 랜섬웨어로 판단하는 숫자이다. TN는 랜섬웨어가 아닌 경우를 랜섬웨어가 아닌 것으로 판단하는 경우의 수이고, FN은 랜섬웨어인 경우인데 랜섬웨어가 아니라고 판단하는 경우이다.

이 중 우리가 실험에서 사용할 정확도, TPR, FPR에 대하여 살펴본다. TPR는 랜섬웨어로 판단한수 중에서 정확하게 실제 랜섬웨어를 선택한 비율이다.

$$TPR = Detection\ Rate = \frac{TP}{TP + FN} \tag{4}$$

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

FPR은 실제 랜섬웨어가 아니 전체 수 중에 잘못해서 랜섬웨어로 판단한 경우의 비율을 말한다.

정확도는 모든 경우의 수에서 정확한 판단을 한 비율을 말한다.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \tag{6}$$

4-3 교차 검증

k-fold 교차 검증은 기계학습 분류기에서 성능을 측정할 경우 자주 사용하는 방법이다[22],[23]. 이는 모델의 트레이닝을 위한 샘플과 원래 샘플을 나누어 실험하기 위한 분할을 위해 이용한다. k-fold 교차 검증은 원래 데이터를 불규칙적으로 k열의 서브셋으로 분할한다. 그리고 이 중 k-1 서브셋을 적용한 부분은 알고리즘의 테스트를 위해 훈련용으로 사용하고 나머지 k번째 서브셋은 실험 결과를 위해 사용한다. 그런 후에 교차 검증은 k 번 반복한다. 반복 한 후 얻어지는 평가 값의 평균값을 취하는 방법이다.

본 논문에서 우리는 k-fold 교차 검증을 위해 k를 10으로 설정하고 수행하였다. 데이터 셋은 10 서브셋으로 나뉠 것이고, 그 중 9개의 서브셋은 훈련을 위해 사용하였고 1개의 서브셋은 실험 데이터를 위해 사용하였다. 그리고 평균값을 얻기 위해 k번 반복하였다.

4-4 실험 결과 분석

본 논문의 랜섬웨어 탐지를 위해 제안된 알고리즘의 성능을 측정하기 위해 정상적인 소프트웨어와 랜섬웨어 소프트웨어의

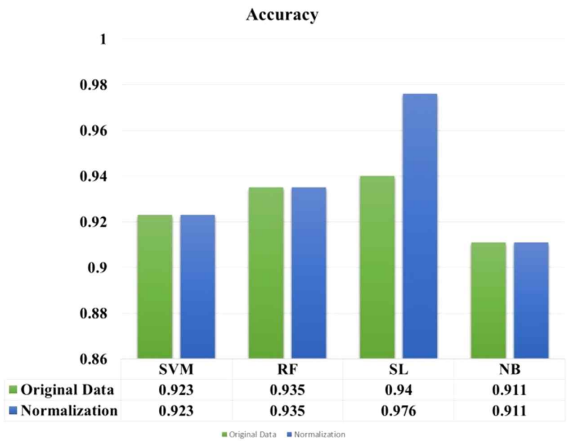

그림 3. 데이터 정규화 비교 실험
Fig. 3. Comparison of the result of Data Normalization

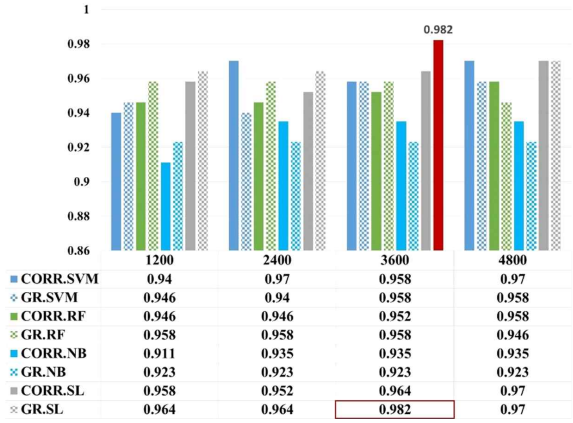

그림 4. 모든 알고리즘의 랜섬웨어 탐지 정확도 비교
Fig. 4. Comparison of the result in terms of Accuracy of the classifiers

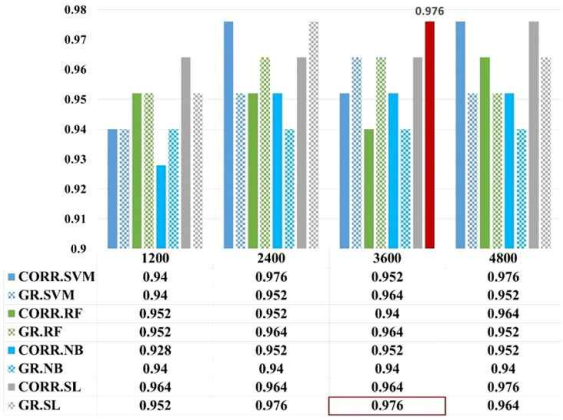

그림 5. 모든 알고리즘의 랜섬웨어 탐지율(TPR) 비교
Fig. 5. Comparison of the result in terms of Detection Rate of the classifiers

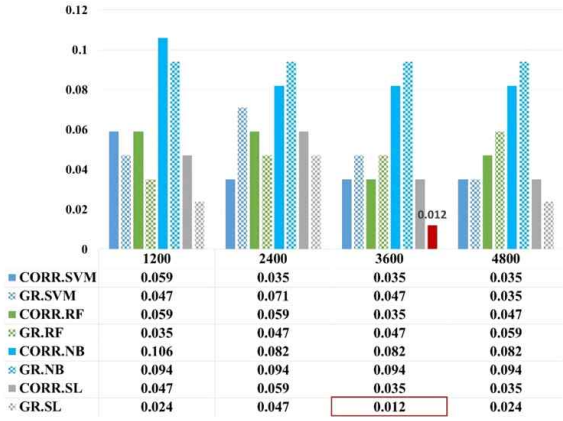

그림 6. 모든 알고리즘의 오탐지율 (FPR) 비교
Fig. 6. Comparison of the result in terms of FPR of the classifiers

두 가지 클래스의 소프트웨어를 수집하였다. 그리고 이렇게 수집된 소프트웨어를 RF, SVM, SL, NB 알고리즘들을 이용하여 비교 실험 하였다.

실험 1: 데이터 정규화가 주는 효과를 알기 위해 우리는 모든 데이터 셋을 3.4 장에 설명된 정규화 과정을 진행한 데이터 셋과 함께 준비했다. 이렇게 준비된 데이터 셋들에 대한 네 가지 다른 기계학습 알고리즘을 이용한 분류기에 적용하여 정확도를 측정하였다.

그림 3은 정규화를 적용한 데이터 셋과 적용하지 않은 데이터 셋을 네 가지의 분류기에 적용하여 정확도를 측정한 그림이다. 모든 분류기의 정확도의 측정값은 91%가 넘는다. 이것은 우리의 데이터 추출 방법이 데이터 마이닝 기술에 잘 맞는다는 것을 의미한다. 특히 정규화를 하지 않은 데이터를 사용한 SL 방법의 결과가 94%의 정확도를 가지고 있고, 정규화 데이터를 사용한 SL방법은 97.6%의 정확도를 가지고 있다. 이것은 데이터 정규화 방법이 SL 방법을 사용한 분류기에서 성능이 향상된 것을 증명하였다.

실험 2: 실험 1에서 데이터 정규화가 데이터 분류 성능 향상에 도움이 된다는 것을 보여주었다. 그러므로 정규화는 이 후 모든 실험 데이터에 적용되어 사용하였다.

본 논문에서, 특징 벡터로 바꾼 데이터의 차원개수는 평균 73,202개에 이를 만큼 매우 많다. 우리는 특징 선택이 분류기의 성능 향상에 도움이 되는 것을 알기 위해 두 가지 특징 선택 방법인 CORR과 GR 를 적용하였다. 우리는 특징 선택 방법에서 정렬된 상위 1,200개, 2,400개, 3,600개, 4,800개의 특징을 적용하여 분류기에 적용하였다. 그림 4는 다른 분류기에 따른 정확도를 비교하여 나타낸 그림이다. 그림 3과 4를 비교해보면 특징 선택 방법을 사용한 실험의 정확도가 높은 것을 발견할 수 있다. 즉 특징 선택 방법을 사용하지 않고 랜섬웨어를 탐지한 경우에는 특징 선택 방법을 사용한 경우보다 성능도 줄어들고 탐지 시간 부하도 커짐을 알 수 있다.

그림 4부터 그림 6까지의 실험은 정확도, 탐지율, 오탐지율

에 대한 실험을 나타낸 그림이다. 이 중 특히 SL 분류기를 사용하고 GR 특징 선택을 한 방법에서의 성능이 가장 우수함을 볼 수 있다. 특히 3,600개의 특징을 가진 SL 방법은 가장 높은 정확도와 탐지율인 98.2%와 97.6%를 기록했다. 그리고 그림 6에서 보듯이 GR과 SL 방법을 이용한 오탐지율은 1.2%을 기록하였다. 결과적으로 보면 우리가 제안한 랜섬웨어 탐지를 위해 사용된 방법들이 효율적이고 좋을 결과를 가져왔음을 알 수 있다.

Ⅴ. 결 론

본 논문에서 우리는 오프라인 상에서 자동으로 랜섬웨어를 탐지하고 분석할 수 있는 데이터 마이닝 기반의 시스템을 제안하였다.

이를 위해 본 논문에서는 실제 소프트웨어의 행위를 감시하고 특징을 추출하여 API CFG를 생성하였다. 그 후 데이터 정규화와 특징 선택을 통해 중요하고 가치 있는 특징들만을 선별하였다. 이러한 특징들은 네 종류의 기계학습 알고리즘을 이용한 분류기에 적용시켜 성능을 측정하였다.

실험 결과인 4.4에서는 정규화가 분류기의 성능향상에 효율적인 방법 중 하나임을 보여 주었다. 특징 선택방법인 CORR과 GR 방법은 랜섬웨어 탐지를 위한 최고의 알고리즘인 SL방법과 함께 가장 좋은 결과를 나타냈다. 그러므로 우리가 제안한 특징 추출, 정규화, 특징 선택 방법과 기계학습 알고리즘이 결합하여 높은 랜섬웨어 탐지율과 정확도, 그리고 낮은 오탐지율을 기록함을 증명하였다. 이를 통해 컴퓨터에 저장되어 있는 디지털 콘텐츠의 보호에 기여할 수 있었다.

앞으로 보완해야 할 점으로는 랜섬웨어 데이터 셋의 수와 탐지 시간과의 관계에 대한 실험을 수행하지 못하였다. 본 논문에서 우리는 단지 168개의 소프트웨어만 (랜섬웨어와 정상 소프트웨어 모두 포함)을 이용하여 특징을 추출하였고 전 과정에 대한 작업시간과 탐지시간에 대한 실험을 하지 못했다. 특히 수집된 랜섬웨어 패밀리의 수가 적어 유사한 API 함수 호출을 하는 같은 패밀리의 랜섬웨어의 경우에는 탐지가 쉽지만 새로운 구조의 랜섬웨어 대응에 대처하기가 힘들다. 그러므로 앞으로 더 많은 랜섬웨어 패밀리 종류와 샘플을 수집하고 알고리즘을 보완하는 연구를 통해 더 좋은 성능의 랜섬웨어 탐지 방법을 찾아내야 할 것이다.

감사의 글

이 논문은 2016학년도 건국대학교의 연구년교원 지원에 의하여 연구되었음

참고문헌

[1] Tech Times News, Cybersecurity: SonicWall Threat Report [Internet] Available: http://www.techtimes.com/articles/196580/20170208/cybersecurity-sonicwall-threat-report-shows-malware-slightly-dropped-but-ransomware-surged-in-2016.htm.

[2] Z.-G. Chen, H.-S. Kang, S.-N. Yin and S.-R. Kim, “Automatic Ransomware Detection and Analysis Based on Dynamic API Call Flow Graph,” in *Processing of 2017 Research in Adaptive and Convergent System*, Poland, 2017

[3] G. Nguyen, V. Nguyen, S. Nguyen, and K. Kim, “Efficient Association Rule Mining based SON Algorithm for a Bigdata Platform,” *Journal of Digital Contents Society*, Vol.18, No.8, pp.1593-1601, December 2017.

[4] M. A. Aydın, A. H. Zaim, and K. G. Ceylan, “A hybrid intrusion detection system design for computer network security,” *Computers & Electrical Engineering*, Vol.35, No.3, pp.517-526, 2009.

[5] J. Lee, K. Jeong, and H. Lee, “Detecting metamorphic malwares using code graphs,” in *Proceedings of the 2010 ACM symposium on applied computing*, pp. 1970-1977. ACM, 2010.

[6] F. Karbalaie, A. Sami, and M. Ahmadi, “Semantic malware detection by deploying graph mining,” *International Journal of Computer Science Issues*, Vol.9, No.1, pp.373-379, 2012.

[7] N. Nissim, R. Moskovitch, L. Rokach, and Y. Elovici, “Novel active learning methods for enhanced pc malware detection in windows os,” *Expert Systems with Applications*, Vol.41, No.13, pp.5843-5857, 2014.

[8] J. Saxe and K. Berlin, “Deep neural network based malware detection using two dimensional binary program features,” in *Proceedings of Malicious and Unwanted Software (MALWARE)*, 2015 10th International Conference on, pp.11-20, 2015.

[9] D. Kim and S. Kim, “Design of quantication model for ransom ware prevent,” *World Journal of Engineering and Technology*, Vol.3, No.03 pp.203, 2015.

[10] D. Sgandurra, L. Mu~noz-Gonzalez, R. Mohsen, and E. C. Lupu, “Automated dynamic analysis of ransomware: Benets, limitations and use for detection,” *arXiv preprint arXiv*:1609.03020, 2016.

[11] S. Song, B. Kim, and S. Lee, “The effective ransomware prevention technique using process monitoring on android platform,” *Mobile Information Systems*, 2016

[12] A. Kharraz, S. Arshad, C. Mulliner, W. K. Robertson, and E. Kirda, “Unveil: A large-scale, automated approach to detecting ransomware,” in *Proceedings of USENIX Security Symposium*, pp.757-772, 2016.

[13] APIMonitor.com, Win32 API Monitor tool [Internet], Available: http://www.apimonitor.com/.

[14] S. Kotsiantis, D. Kanellopoulos, and P. Pintelas, “Data preprocessing for supervised leaning,” *International Journal of Computer Science*, Vol.1, No.2, pp.111-117, 2006.

[15] K. Rieck, P. Trinius, C. Willems, and T. Holz, “Automatic analysis of malware behavior using machine learning,” *Journal of Computer Security*, Vol.19, No.4, pp.:639-668, 2011.

[16] M. A. Hall, “Correlation-based feature selection for machine learning,” 1999.

[17] J. Benesty, J. Chen, Y. Huang, and I. Cohen, “Pearson correlation coecient,” in *Noise reduction in speech processing*, pp.1-4, 2009.

[18] A. G. Karegowda, A. Manjunath, and M. Jayaram, “ Comparative study of attribute selection using gain ratio and correlation based feature selection,” *International Journal of Information Technology and Knowledge Management*, Vol.2, No.2, pp.271-277, 2010.

[19] Software.informer, Benign softwares [Internet], Available: http://software.informer.com/software/.

[20] VirusShare [Internet], Available: http://virusshare.com/.

[21] S. V. Stehman, “Selecting and interpreting measures of thematic classification accuracy,” *Remote sensing of Environment*, Vol.62, No.1, pp.77-89, 1997.

[22] R. R. Picard and R. D. Cook, “Cross-validation of regression models,” *Journal of the American Statistical Association*, Vol.79, No.387, pp.575-583, 1984.

[23] G. Seni and J. F. Elder, “Ensemble methods in data mining: improving accuracy through combining predictions,” *Synthesis Lectures on Data Mining and Knowledge Discovery*, Vol.2, No.1, pp.1-126, 2010.

강호석(Ho-Seok Kang)

2000년 : 홍익대학교 전자전기컴퓨터공학부 (공학사)

2002년 : 홍익대학교 전자계산학과 (이학석사)

2008년 : 홍익대학교 컴퓨터공학과 (공학석사)

2002년～2011년: 홍익대학교, 한국폴리텍I대학, 서강대학교 게임교육원 시간강사

2007년～2011년: 건다감플러스 주식회사 기술연구소 소장

2011년～현 재: 건국대학교 소프트웨어학과 AIS연구실 박사후연구원

2007년～현 재: 건국대학교 유비쿼터스정보기술연구원 박사후연수연구원

※관심분야 : 컴퓨터 보안, 네트워크 보안, 사물 인터넷, 분산 컴퓨팅 등

김성열(Sung-Ryul Kim)

1993년 : 서울대학교 컴퓨터공학과 (공학사)

1995년 : 서울대학교 컴퓨터공학과 대학원(공학석사)

2000년 : 서울대학교 컴퓨터공학과 대학원(공학박사)

2000년～2002년: 미국 WiseNut Inc. Engineering Manager

2008년～2009년: 국정원 보안표준 심사위원

2009년～2009년: 행정자치부 보안분야 평가위원

2013년～2017년: ACM RACS Program Chair

2002년～현 재: 건국대학교 소프트웨어학과 교수

※관심분야 : 암호 알고리즘, 분산 알고리즘, 컴퓨터 보안, 클라우드 컴퓨팅, 데이터 마이닝 등