Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.30, NO.6, Dec. 2020 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2020.30.6.1013

서바이벌 네트워크 개념을 이용한 저자 식별 프레임워크 : 의미론적 특징과 특징 허용 범위*

황 철 훈,[1†] 신 건 윤,[1] 김 동 욱,[1] 한 명 묵[2‡]
[1,2]가천대학교 (대학원생, 교수)

Authorship Attribution Framework Using Survival Network Concept : Semantic Features and Tolerances*

Cheol-Hun Hwang,[1†] Gun-Yoon Shin,[1] Dong-Wook Kim,[1] Myung-Mook Han[2‡]
[1,2]Gachon University (Graduate student, Professor)

요 약

악성코드 저자 식별은 알려진 악성코드 저자의 특징을 이용하여 알려지지 않은 악성코드의 저자 특징과 비교를 통해 악성코드를 식별하기 위한 연구 분야이다. 바이너리를 이용한 저자 식별 방법은 실질적으로 배포된 악성코드를 대상으로 수집 및 분석이 용이하다는 장점을 갖으나, 소스코드를 이용한 방법보다 특징 활용 범위가 제한된다. 이러한 한계점으로 인해 다수의 저자를 대상으로 정확도가 저하된다는 단점을 갖는다. 본 연구는 바이너리 저자 식별에 한계점을 보완하기 위하여 '바이너리로부터 의미론적 특징 정의'와 '서바이벌 네트워크 개념을 이용한 중복 특징에 대한 허용 범위 정의' 방법을 제안한다. 제안한 방법은 바이너리 정보로부터 Opcode 기반의 그래프 특징을 정의하며, 서바이벌 네트워크 개념을 이용하여 저자별 고유 특징을 선택할 수 있는 허용범위를 정의하는 것이다. 이를 통해 저자별 특징 정의 및 특징 선택 방법을 하나의 기술로 정의할 수 있으며, 실험을 통해 선행연구보다 5.0%의 정확도 향상과 함께 소스코드 기반 분석과 동일한 수준의 정확도 도출이 가능함을 확인할 수 있었다.

ABSTRACT

Malware Authorship Attribution is a research field for identifying malware by comparing the author characteristics of unknown malware with the characteristics of known malware authors. The authorship attribution method using binaries has the advantage that it is easy to collect and analyze targeted malicious codes, but the scope of using features is limited compared to the method using source code. This limitation has the disadvantage that accuracy decreases for a large number of authors. This study proposes a method of 'Defining semantic features from binaries' and 'Defining allowable ranges for redundant features using the concept of survival network' to complement the limitations in the identification of binary authors. The proposed method defines Opcode-based graph features from binary information, and defines the allowable range for selecting unique features for each author using the concept of a survival network. Through this, it was possible to define the feature definition and feature selection method for each author as a single technology, and through the experiment, it was confirmed that it was possible to derive the same level of accuracy as the source code-based analysis with an improvement of 5.0% accuracy compared to the previous study.

Keywords: Authorship Attribution, Survival Network, Call Graph, Cosine Similarity, Support Vector Machine

Received(06. 05. 2020), Modified(10. 07. 2020), Accepted(10. 27. 2020)

* 이 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(NRF-2018R1D1A1B07050864).

† 주저자, qewqsa@naver.com
‡ 교신저자, mmhan@gachon.ac.kr(Corresponding author)

I. 서 론

은닉화, 암호화, 변이 등 다양한 우회기술로 인하여 다양하고 복잡한 악성코드가 생성 및 배포되고 있다[1]. 악성코드 저자 식별은 이를 해결하기 위한 포렌식(Forensic) 기반의 탐지 기술로써 알려진 악성코드를 기반으로 저자의 특징을 추출하여 알려지지 않은 악성코드의 저자와 비교하는 것으로 악성코드를 탐지할 수 있다[2, 3]. 악성코드 저자 식별은 소스코드를 기반으로 한 저자 식별 방법과 바이너리를 기반으로 한 저자 식별 방법으로 구분되며, 바이너리를 이용한 식별 방법은 데이터 수집 및 분석에 용이하다는 점으로 활발히 연구되고 있는 분야이다[3].

바이너리 기반 방법은 문자열, 함수, 호출 흐름 등 다양한 특징을 이용하는 소스코드 기반 방법과는 다르게 제한적인 바이너리를 특징으로 이용하기 때문에 다수의 저자를 대상으로 한 분류 시 특징의 중복으로 인하여 정확도 저하 문제를 일으킨다[3]. 바이너리 특징 정의의 한계점을 해결하기 위하여 Call 명령 기반의 템플릿을 이용한 악성코드 탐지[4]와 연속된 Opcode 명령을 이용한 악성코드 카테고리 분류 방법[5] 등과 같이 특징 정의 방법을 고도화하는 연구가 이루어졌으며, 이외에도 바이너리를 디컴파일 하여 소스코드로 변환하여 분석하는 방법도 최근 연구되었다[6]. 위의 실험들은 저자에 대한 특징 중복 문제를 해결하기 위해서 특징 선택 단계에서 상호의존정보(mutual information), 정보 이득(information gain)기술을 이용하여 저자마다의 중요 특징을 선택하였으며, 선택 기준은 실험마다 별도로 정해진다.

본 연구는 바이너리를 기반으로 한 저자 식별 연구에서 발생하는 저자의 중요 특징 선택 방법에 대하여 새로운 방법을 제안한다. 이는 바이너리로부터 저자별 특징을 정의하며, 정의된 특징을 기반으로 저자 고유의 특징 허용 범위를 지정하는 것이다. 별도의 특징 선택 기술 없이 특징 정의 단계에서 저자 고유의 특징 관계를 나타낼 수 있다는 점에서 의미있는 결과라고 판단하였다. 이를 위해 바이너리를 이용한 특징 정의 단계에서 의미론적 접근을 위해 시스템 흐름을 기반으로 한 그래프 특징 정의 방법을 제안하였으며, 또한 서바이벌 네트워크 개념을 접목하여 중복 허용 범위를 지정함으로써 저자의 특징 중복 문제를 완화하고 저자 고유 특징을 정의하고자 하였다.

본 글은 2장에서 관련된 연구를 소개하며, 3장에서 제안한 프레임워크를 설명한다. 4장을 통해 실험 및 결과를 보이며, 5장 결론을 통해 마무리한다.

II. 관련 연구

2.1 바이너리 기반의 저자 식별

저자 표시는 많은 포렌식 기반의 분석에 있어서 중요한 특징이 될 수 있다. 악성코드를 작성한 저자의 특징을 기반으로 제작 범위를 좁혀 악성코드 출처로 표현될 수 있으며, 알려진 악성코드와 연관시켜 알려지지 않은 악성코드에 대한 중요한 특징을 얻을 수 있다. 이는 인간의 습관 또는 약속 등과 같이 습관적인 저자의 스타일 특징이 악성코드 탐지에 활용될 수 있다는 점을 이용한 연구 방법이다[1].

저자 식별은 크게 소스코드를 이용하는 방법과 바이너리를 이용하는 방법이 있다[3]. 소스코드를 이용하는 방법은 변수명, 함수명, 함수 호출 순서 등 저자의 제작 습관을 반영한 다양한 특징을 정의하여 분석한다. 선행연구에서는 Linguistic Feature, Formatting, Bugs, Execution Path, Abstra-ct Syntax Tree, Control Flow Graph 등이 활용되어 연구되었다. 소스코드 분석은 다양한 특징을 활용하여 저자를 식별함으로써 정확도가 높지만, 현실적으로 소스코드를 얻기 힘들다는 점에서 연구의 한계를 갖는다. 바이너리를 이용하는 방법은 분석을 위한 데이터 수집이 용이하다는 장점을 갖지만, 바이너리로부터 특징을 정의한다는 점에서 특징 정의의 한계를 갖는다. 선행연구는 System Call, Contrl Flow Graph 등을 이용한 의미론적 분석 방법과 n-gram, Idiom, Graphlet과 같은 템플릿을 이용한 방법으로 저자 특징을 정의는 연구가 진행되었다. 최근에는 디컴파일을 통해 바이너리에서 소스코드로 변환하여 저자를 식별하는 연구도 진행되었으나 디컴파일 과정에서 많은 시간이 소모된다는 점에서 아쉬운 성과를 도출하였다[6]. 바이너리 특징을 활용한 저자 식별 방법의 공통적인 문제는 특징 정의의 한계점으로 인해 식별할 저자가 증가할수록 특징이 중복되어 탐지 정확도가 낮아진다는 점이다[7]. 이에 중복된 바이너리 특징을 제거하고 저자 고유의 특징을 판별하는 방법에 대한 연구가 필요하지만 관련 연구 결과가 활성화되지 않았다.

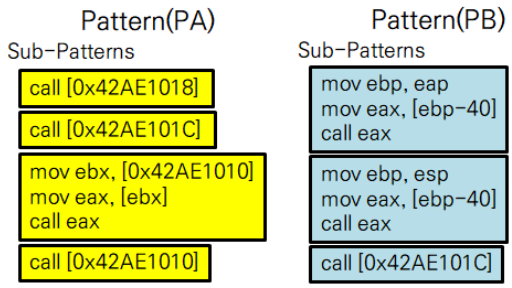

Fig. 1. Some of the Sapphire Worm's snippet patterns (PA) and variant versions (PB).

2.2 Call Template을 이용한 악성코드 탐지

정적 기반 악성코드 탐지는 우회기술 등으로 고도화된 악성코드를 탐지하는 한계성을 갖는다. 이를 해결하기 위해 행동 기반의 의미론적 특징 패턴 정의 방법을 제안하였다. Call 기반의 템플릿 특징 정의를 통해 시스템 흐름 패턴을 이용하는 악성코드 탐지할 수 있도록 하였다[4]. 이를 실현시키기 위해 두 가지 방법을 소개하였으며, Call 명령이 포함된 연속된 Opcode 명령을 하나의 템플릿으로 정의하는 방법과 템플릿 기반 패턴 매칭 방법이다.

시스템 호출 및 라이브러리 함수 호출은 악성코드의 기능을 나타낼 수 있으며, 이를 특징으로 활용하여 악성코드 제어 흐름 및 데이터 흐름 분석에 사용할 수 있다. 디셈블 과정을 통해 추출된 Call 명령은 이러한 시스템 및 라이브러리 호출을 나타내줄 수 있으며, 이러한 연속된 명령은 Fig.1.과 같이 템플릿 특징으로 활용할 수 있다. 정의된 패턴에 대한 매칭 방법은 비교적 간단하며 클래스 라벨별 정의된 특징 전체에 대하여 입력된 데이터의 특징 포함 수를 평균으로 구하면 된다. 산출된 평균값을 통해 입력된 데이터의 클래스 분류가 진행되며 분류 기준은 실험을 통해 유동적으로 변경된다.

제안된 방법은 실험을 통해 변이에 대한 어느 정도 의미 있는 결과를 얻을 수 있었으나 모든 우회 기술에 대응할 수 없으며, "junk" 기능에 대한 별도의 대처를 하지 않기 때문에 차후 문제가 발생할 수 있다는 점을 한계로 지적하였다.

2.3 System Call 특징을 이용한 악성코드 탐지

악성코드의 감염의 수와 정교함이 크게 증가하였으며, 이러한 위협을 완화시키기 위한 탐지 기술이

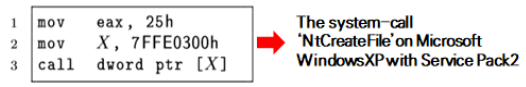

Fig. 2. Consecutive commands including Opcode's Call command can be converted into a function of System-call.

제안되어왔다[7, 8]. D. Canali 외 5명의 연구자들은 System Call 기반의 악성코드 탐지 선행 연구들을 기반으로 정확성에 대한 정량적 연구에 대해 조사하였다[9]. 제안된 방법들은 기존의 정적 기반의 탐지 방법을 보완하기 위하여 행동을 기반으로 한 의미론적 특징 정의 방법을 이용하였으며, 이는 탐지 정확도가 높아지는 결과로 발전하였다. 연구에서는 System Call, API Call 등을 이용한 악성코드의 기능과 시스템 흐름을 중심으로 한 의미론적 특징을 활용하였다. 논문을 통해 n-gram을 통한 정확도 변화 실험뿐만 아니라 System Call과 Argument, Action에 대한 특징 활용 시 고려할 점을 제시하였다.

선행 연구[5]에서는 악성코드의 의미론적 특징을 활용하는 방법으로 System Call을 이용하였으며, System Call은 악성행동 연구를 통해 Fig.2.와 같이 Opcode의 Call 명령이 포함된 연속된 명령 특징으로 변환 가능함을 보였다[10]. Opcode의 Call 명령을 기준으로 한 특징 정의 방법이 의미론적 특징으로 이용될 수 있음을 알 수 있다.

2.4 서바이벌 네트워크를 이용한 네트워크 관리

서바이벌 네트워크(Survival Network)는 여러 대상의 정보를 네트워크로 통합하는 과정에서 중복되거나 반복되는 정보를 관리할 수 있도록 하는 개념이다. 중복되거나 반복되는 정보들은 서바이벌 네트워크의 규칙을 통해 제거하거나 재정의 하는 방법 등을 이용하여 해결한다. 이는 학습을 통해 비대해지는 네트워크 크기를 관리하거나 특징 중복에 대한 해결 방법으로 주로 이용한다. Boolean Network 모델을 통한 림프구 백혈병 식별 연구에서 학습을 통해 비대해지는 네트워크 모델의 문제를 해결하기 위한 방법으로 이용되었다[11]. 각 노드를 림프구 백혈병을 발생시키는 특징으로 하며, 각 간선을 AND와 OR의 관계로 나타낸다. 학습을 통해 점차 Boolean Network 모델의 노드 및 간선이 복잡하게 정의되며 이는 Running Time 증가 및 자원 증가의 문제

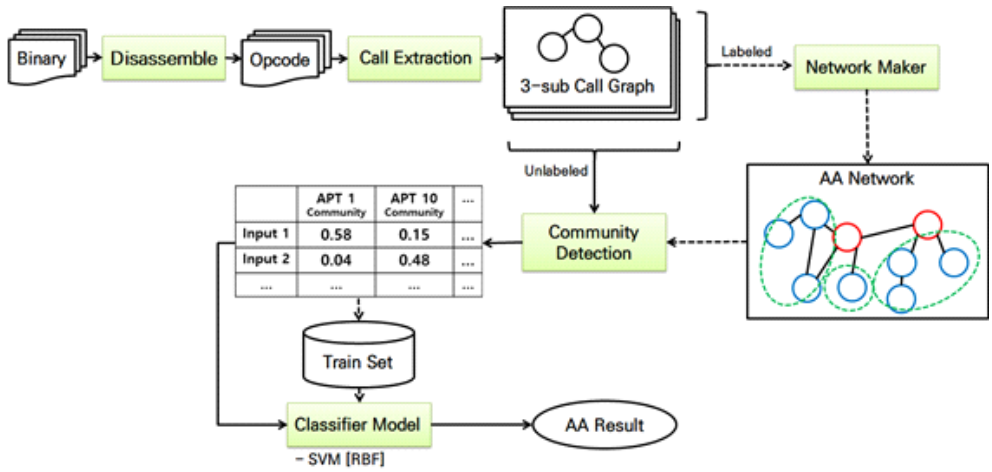

Fig. 3. System flow of the proposed framework.

점을 발생시킨다. 이를 해결하기 위한 방법으로 서바이벌 네트워크를 이용하여 노드와 특징 관계를 완화시키도록 하였다. 'TCR OR SIP AND CTL4' 라는 관계에 있어서 SIP 특징은 크게 중요한 영향을 주지 않으므로 SIP 특징을 제거하고 'TCR AND CTL4' 특징으로 정의한다. 영향력 또는 산술적인 방법 등을 통해 규칙을 만들어 중복 또는 제거하여 관리하는 것을 서바이벌 네트워크라 한다. 해당 기술은 보안 분야에서 네트워크 침입 탐지 분야에 활용되었다[12]. 네트워크 침입 시 발생하는 여러 방법들에 대하여 중복 및 제거를 통해 네트워크 침입의 중요 특징을 정의하였다.

III. 제안 프레임워크

본 연구는 선행 연구[4, 5, 6]에서부터 발생되고 있는 바이너리로부터 특징 정의 한계점과 특징 중복으로 발생하는 정확도 저하를 완화시키고자 연구를 진행하였다. 바이너리로부터 특징 정의 한계를 보완하기 위해 디셈블 과정을 통해 얻는 Opcode를 활용하여 의미론적인 특징을 정의며, 행동 패턴을 기반으로 분류하고자 하였다. 또한 다수의 저자를 대상으로 나타나는 특징 중복을 해결하고자 서바이벌 네트워크 개념을 이용하여 중복된 특징에 대한 허용 범위를 정의하여 관리하고자 하였다.

제안하는 프레임워크는 바이너리로부터 저자를 식별하는 것을 목표로 한다. 전체 프레임워크는 Fig. 3.과 같으며, 크게 네 가지 단계를 갖는다.

1. 바이너리로부터 3-sub Call Graph 정의.
2. 서바이벌 네트워크 개념을 통한 저자 식별 네트워크 구축.
3. 정의된 네트워크로부터 저자별 커뮤니티를 활용하여 가중치 테이블을 정의.
4. 가중치 테이블을 이용하여 분류 모델 학습 및 저자 분류.

첫 과정은 Disassemble를 통해 바이너리를 Opcode로 변환한다. 변환된 Opcode로부터 연속된 Call 명령을 통해 3-sub Call Graph를 정의한다. 3-sub Call Graph는 Fig.2.와 같이 연속된 Opcode 명령이 하나의 System Call 명령과 일치될 수 있다는 선행연구를 기반으로 한다[9, 10]. 정의 시 Opcode의 연속된 순서에 따른 방향성을 지니며, Call 명령을 마지막에 위치한다는 기준을 가진다. 또한 3-sub Graph 특징으로 정의한 이유는 그래프의 크기 N을 4 이상으로 정의하였을 때 정확도의 큰 차이가 없으며, 자원 소모가 크다는 선행 연구 결과를 반영하였다[9]. 만약 클래스 라벨이 정의되었을 때, Network Maker를 통해 저자 식별 네트워크(Authorship Attribution Network)를 정의한다. 만약 클래스 라벨이 정의되지 않았을 때, 저자 식별 네트워크에 정의된 각 클래스 별 커뮤니티와 입력 데이터에서 추출한 그래프 특징을 비교하여 가중치 테이블을 산출한다. Network Maker는 서바이벌 네트워크 개념을 기반으로 저자 식별 네트워크를 정의하며, 해당 개념을 통해 중복 허용 범위를 정의한다. 특징 중복 정도를 산출하기 위해 Cosine

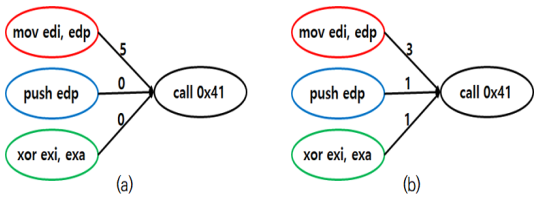

Fig. 4. There are examples of overlapping features a and b. Suppose red is Class A, blue is Class B, and green is Class C. The weight of each edge is the number of times each call is made.

Similarity를 이용하였다. 예를 들어 Fig.4.와 같은 상황이 있다고 가정하자. 노드는 Opcode 명령이며, 간선은 이전 명령이 이후 명령에 도달하는 횟수이다. 횟수가 높을수록 학습을 통해 이전 명령이 이후 명령으로 자주 도달함을 알 수 있다. (a) 상황에 대하여 먼저 클래스별 유사도를 산출한다면, 클래스 순서대로 (5, 0, 0)이라는 횟수 값을 추출할 수 있다. 이를 정규화(1, 0, 0) 하여 Class A(1, 0, 0), Class B(0, 1, 0), Class C(0, 0, 1)에 대한 Cosine Similarity를 산출한다. 결과적으로 1.0, 0.0, 0.0의 값을 산출되며, (a)의 상황은 Class A에 1.0 정도의 유사도를 지니고 있음을 확인할 수 있다. (b)의 상황에서도 동일하게 진행할 경우, 0.9, 0.3, 0.3의 값이 산출되며 Class A에 0.9의 유사도를 보이고 나머지에 0.3의 유사도를 보이고 있음을 알 수 있다. 만약 0.5이상의 유사도를 기준으로 중복을 허용하게 된다면 (a), (b)는 각각 1.0, 0.9의 Class A의 유사도를 보이고 있기 때문에 모두 Class A에 대한 특징이 된다. 0.95 이상의 유사도를 기준으로 중복을 허용하게 된다면 (a)의 상황만 Class A에 대한 특징으로 이용된다. 이와 같은 방법을 통해 Cosine Similarity를 이용하여 중복 허용 범위를 정의하였다.

중복 허용 범위를 통해 중복된 특징을 제거 또는 허용하여 각 저자별 특징이 모인 커뮤니티를 생성한다. 커뮤니티는 Community Detection을 통해 입력 데이터가 각 저자별 커뮤니티에 대하여 가중치를 가지고 있는지 산출한다. 입력 데이터를 G라고 하고 입력 데이터 G의 크기를 n이라고 한다. 각 클래스별 커뮤니티를 C라고 하며, 커뮤니티의 크기를 m이라고 정의한다. x는 3-sub Call Graph의 값 $[G=x_1, x_2, ..., x_n]$을 나타내며, y는 클래스별 커뮤니티 $[C=y_1, y_2, ..., y_m]$를 나타낸다. 가중치를 구하는 공식은 아래와 같다.

$$f(C_m) = \frac{\sum_{i=1}^{n} cmp(x_i, C_m)}{n} [m = 1, 2, ..., N] \quad (1)$$

$$cmp(x, y) \begin{cases} 1 (x \in y) \\ 0 (x \notin y) \end{cases} \quad (2)$$

공식 (2)에서 정의한 cmp함수를 통해 입력 데이터에서 생성된 3-sub Call Graph 특징 x가 커뮤니티 y에 포함되는지 확인한다. 만약 포함된다면 1, 아니면 0으로 결과를 반환한다. 공식 (1)에서 이를 통해 전체 입력데이터의 3-sub Call Graph에 대하여 커뮤니티에 포함된 수를 평균화하여 가중치를 산출한다. 가중치 테이블은 입력된 데이터에 대하여 각 클래스별 가중치로 변환하여 데이터를 가공한다. 가공된 데이터가 만약 클래스 라벨이 존재하는 경우 학습데이터로 이용되며, 라벨이 없는 경우 분류 모델을 통해 저자 분류를 진행한다. 최종 결과는 저자 분류 결과가 도출된다.

IV. 실험 및 결과

본 실험에서는 '제안한 프레임워크가 선행 연구보다 좋은 결과를 도출하는 가'이다. 비교를 위한 실험으로 Call 기반의 템플릿을 이용하여 악성코드를 탐지한 선행 연구를 선정하였다. 이유는 오래된 연구 방법이지만 바이너리 기반 연구에 특징 정의 시 핵심이 되는 연구이며, 선행연구와 제안한 연구 모두 Call 기반의 Opcode를 활용하여 행동 분석을 위한 특징 정의 방법을 이용한다는 점이다. 선행연구는 템플릿 기반 특징 정의 방법을 활용하였다는 점에서 그래프를 이용한 방법과의 차이점을 추가로 확인할 수 있다[4]. 제안된 실험에서 분류 모델은 RBF (Radial Basis Function) 커널을 사용한 서포트 벡터 머신(Support Vector machine, SVM)을 이용하였다. 이는 다중 클래스 구분에 있어서 매개변수 조정을 통해 높은 안전성을 얻을 수 있기 때문이다[14]. 검증은 10-fold cross validation을 이용하였다.

4.1 데이터 세트

본 실험은 악성코드 저자 식별 연구에 이용된 데이터를 이용하였다[13]. 데이터는 Table 1.과 같다. 클래스 라벨은 APT 그룹을 대상으로 하며, 선행 연구와 동일하게 APT 그룹별 작성한 저자들의 특징을 추출 및 식별하는 것을 전제로 한다. 총 10개 클래스를 대상으로 하며 총 데이터는 992개의 바이너리 악성코드를 분석한다.

Table 1. Experiment Data

Country	APT Group	Number
China	APT 1	133
China	APT 10	100
China	APT 21	51
Russia	APT 29	100
China	APT 30	100
China	Winnti	108
North-korea	Dark Hotel	100
Russia	Energetic Bear	100
Usa	Equation Group	100
Pakistan	Gorgon Group	100

4.2 정확도 비교 실험

실험 과정은 3장에 서술된 내용과 같이 진행하였다. 성능에 대한 정확도 비교 실험을 위해 최적의 중복 허용 범위에 대한 정의가 필요하다. 이를 위해 APT1, APT10, Winnti 그룹을 대상으로 먼저 실험을 진행하였다. 비교를 위해 중복 시 제거(remove), Cosine Similarity 0.80 이상(cos 〉= 0.80), 0.85 이상(cos 〉= 0.85), 0.90 이상(cos 〉= 0.90), 0.95 이상(cos 〉= 0.95)으로 구분하여 진행하였다. 실험 결과는 Table 2.와 같다. 가장 좋은 결과를 보여준 것은 Cosine Similarity가 0.90 이상일 때이며, 0.95 이상일 경우 중복 시 완전히 제거한 경우와 같은 결과를 보였다. 실험 결과를 보아 특징 중복이 정확도에 영향력을 미치는 것을 확인할 수 있었다. 또한 유사도를 활용한 중복 허용 범위를 정의하는 것으로 특징 중복에 유동적으로 대처할 수 있음을 확인할 수 있었다. 특히, 높은 유사도는 오히려 특징 중복 시 제거하는 경우와 같을 수 있기 때문에 적정 기준에 대한 관찰 및 실험이 필요함을 확인할 수 있었다.

Table 2. Classification accuracy results according to overlapping tolerance.

	APT 1	APT 10	Winnti
remove	90.50	91.26	92.56
Cos 〉= 0.8	91.30	92.10	92.90
Cos 〉= 0.85	92.10	93.26	93.70
Cos 〉= 0.9	92.76	93.95	94.10
Cos 〉= 0.95	90.50	91.26	92.56

가장 좋은 결과를 보인 'Cosine 0.90 이상'으로 중복 허용 기준을 정의하고 선행 연구와 비교를 위한 10개의 저자에 대한 분류 실험을 진행하였다. 두 실험은 Opcode의 연속된 Call 명령을 통해 특징을 정의하는 점에서 유사한 점을 갖으나 선행 연구는 템플릿을 통한 특징을 정의한다는 점과 중복 특징에 대한 처리를 별도로 진행하지 않는다는 점에서 제안한 방법과 차이를 갖는다. 이는 Table 3.을 통해 차이를 확인할 수 있다. 10개의 APT 그룹에 대하여 제안한 방법이 모두 높은 정확도를 보였다. 최대 12.26%의 차이를 보였으며, Fig.1.과 같이 연속된 Call 명령 또는 단일의 Call 명령을 템플릿으로 지정하여 특징으로 이용한다는 점에서 클래스별 중복되는 특징이 많을 것으로 판단하였다. 확인 결과 10개의 APT 그룹 간에 약 33.5%의 특징 중복이 나타

Table 3. Comparison of suggested methods for author identification and prior studies (DH : Dark Hotel, EB : Energetic Bear, EG : Equation Group, GG : Gorgon Group, - : Original, RFR : Repeated Feature Removal).

APT Group	Call Template		Propose - Cos 〉= 0.90
	-	RFR	
APT1	85.70	88.52	92.75
APT10	86.90	89.48	93.95
APT21	87.73	90.02	96.55
APT29	86.52	88.94	95.75
APT30	89.23	93.27	97.33
Winnti	89.90	92.16	94.10
DH	83.55	86.34	95.50
EB	82.47	85.36	94.73
EG	80.38	83.14	91.95
GG	81.45	85.67	92.45

남을 확인하였다. 중복 특징을 단순 제거하여 다시 실험을 진행하였을 때, 최대 4.04%의 정확도 향상을 확인할 수 있었다. 그러나 제안한 방법보다 낮은 정확도를 보였으며, 이는 그래프 특징을 이용하였다는 점에서 템플릿을 이용한 특징 정의 방법과 그래프를 이용한 의미론적 특징 정의 방법의 차이로 볼 수 있다. 또 다른 관점에서는 중복 시 단순 제거하는 경우, 한번 중복되는 경우와 여러 번 중복되는 경우의 가중치를 동일하게 본다는 점에서 나타나는 결과로 볼 수 있다. 이는 중복 허용 범위를 정의하는 제안한 방법과의 차이를 보이며, 이에 따른 결과 차이로 해석할 수 있다.

위의 실험을 통해 제안한 방법이 비슷한 방법으로 진행된 선행연구보다 좋은 결과를 보임을 확인할 수 있었다. 이에 바이너리 기반 저자 식별에 선행연구인 연속된 Opcode를 기반으로 식별하는 방법(Sequence Opcode)[5]과 바이너리 정보를 디컴파일 하여 소스코드 정보로 변환하여 분석하는 방법(Decompiled Binary)[6]에 대하여 추가적인 비교 실험을 진행하였다. 이는 Table 4.를 통해 확인할 수 있다. 결과적으로 제안한 방법이 기존 선행연구보다 높은 정확도를 보임을 알 수 있었다. 연속된 Opcode 선행연구와 비교하였을 때, 연속된 Opcode 명령 특징을 그래프로 이용한다는 점에서는 동일하지만, 저자의 고유 특징 정의 방법 없이 그래프 특징을 이용한다는 점에서 발생하는 차이로 해석할 수 있다. 바이너리를 디컴파일 하여 소스코드 정보를 얻어 소스코드 기반 분석을 활용하는 경우는 디컴파일 소요 시간에 많이 소모하여 Running Time이 평균적으로 10배 이상 차이가 났다. 저자 특징에 대하여 상호의존정보를 활용하였으며, 결과적으로 제안한 방법과 같거나 오히려 낮은 정확도를 보이기도 하였다. 이는 예상외의 결과이며, 오히려 소스코드 정보를 활용하기에 더 높은 정확도가 도출될 것으로 판단하였다. 아쉽게도 소스코드 특징과 비교하여 제안하는 방법으로 도출된 특징과 같은 특징임을 확인할 수 없었으나, 디컴파일로 정보 변환 시 저자 중요 특징이 제거되거나 수정됨을 예상할 수 있었다. 또한 제안하는 방법이 소스코드와 비슷한 결과로 도출될 가능성이 있음을 확인할 수 있었다.

Table 4. Comparison of suggested methods and prior studies (DH : Dark Hotel, EB : Energetic Bear, EG : Equation Group, GG : Gorgon Group).

APT Group	Sequence Opcode	Decompiled Binary	Propose Method
APT1	88.14	92.75	92.75
APT10	88.63	92.90	93.95
APT21	90.47	95.50	96.55
APT29	91.14	95.75	95.75
APT30	92.74	97.33	97.33
Winnti	89.64	94.10	94.10
DH	90.50	95.15	95.50
EB	89.38	94.33	94.73
EG	87.84	91.67	91.95
GG	88.12	91.94	92.45

V. 결 론

본 논문에서는 바이너리 기반 저자 식별 연구에서 나타나는 특징 중복 문제를 해결하기 위한 방법을 제안하였다. 제안한 방법은 기존 특징 정의 및 특징 선택을 별도의 기술로 이용하는 방법과는 다르게 Opcode를 기반으로 그래프 특징을 정의하는 의미론적 특징 정의 방법과 서바이벌 네트워크 개념을 통해 저자 고유의 특징을 선택할 수 있도록 중복 허용 범위를 정의하는 것이다. 제안한 방법은 선행 연구와의 실험 비교를 통해 특징 정의 방법에 대한 차이뿐만 아니라 특징의 중복 허용 범위를 정의함으로써 최대 5.0%라는 정확도 향상뿐만 아니라 소스코드 기반의 분석 방법과 큰 차이가 나지 않음을 확인할 수 있었다.

차후 연구에서는 정의한 저자 식별 네트워크에 대한 해석, 컴파일러 또는 API 관련 바이너리 특징 제거 및 분류 모델에 따른 정확도 분석 등의 연구를 진행하려고 한다.

References

[1] D. I. Holmes, "Authorship Attribution," Computers and the Humanities, vol. 28, no. 2, pp. 87-106, Apr. 1994.

[2] S. Alrabaee, P. Shirani, M. Debbabi, and L. Wang, "On the Feasibility of Malware Authorship Attribution," *International Symposium on*

Foundations and Practice of Security. Springer, pp. 256-272, Jan. 2017.

[3] E. Stamatatos, "A survey of modern authorship attribution methods," Journal of the American Society for information Science and Technology, vol. 60, no. 3, pp. 538-556, Mar. 2009.

[4] Q. Zhang, DS. Reeves, "MetaAware: Identifying Metamorphic Malware," Twenty-Third Annual Computer Security Applications Conference IEEE, pp. 411-420, Dec. 2007.

[5] B. Kang, S. Yerima, K. McLaughlin, S. Sezer, "PageRank in Malware Categorization," Proceedings of the 2015 Conference on research in adaptive and convergent systems, pp.291-295, Oct. 2015.

[6] A. Caliskan-Islam, R. Harang, A. Liu, A. Narayanan, C. Voss, F. Yamaguchi, "De-anonymizing Program-mers via Code Stylometry," 24th USENIX Security Symposium Security 15, pp. 255-270, Aug. 2015.

[7] Y. Ye, T. Li, D. Adjeroh, and S.S. Iyengar, "A survey on malware detection using data mining techniqu-es," ACM Computing Surveys CSUR, vol. 50, no. 2, pp. 1 - 41, Jun. 2017.

[8] Su-jeong Kim, Ji-hee Ha, Soo-hyun Oh, and Tae-jin Lee, "A Study on Malware Identification System Using Static Analysis Based Machine Learning Technique," Journal of the Korea Institute of Information Security & Cryptology, vol. 29, no. 4, pp. 775 - 784, Aug. 2019.

[9] D. Canali, A. Lanzi, D. Balzarotti, C. Kruegel, M. Christodorescu, E. Kirda, "A Quantitative Study of Accuracy in System Call-Based Malware Detec-tion," Proceedings of the 2012 International Symposium on Software Testing and Analysis, pp. 122-132, Jul. 2012.

[10] M. Christodorscu, S. Jha, C. Kruegel, "Mining specifications of malicious behavior," Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering, pp. 5-14, Sep. 2007.

[11] A. Saadatpour, RS. Wang, A. Liao, X. Liu, T. P. Loughran, I. Albert, R. Albert, "Dynamical and Structural Analysis of a T Cell Survival Network Identifies Novel Candidate Thera-peutic Targets for Large Granular Lymphocyte Leukemia," PLoS computa-tional biology, vol. 7, no. 11, pp. 1-15, Nov. 2011.

[12] V. Q. Marinho, G. Hirst and D. R. Amancio, "Authorship Attribution via network motifs identification," In Proceedings of 5th Brazilian Conference on Intelligent Systems, pp. 355-360, Oct. 2016.

[13] B. Coen, E. Poll, A. C. Searban "Applying Supervised Learning on Malware Authorship Attribution," Digital Security Group Institute for Computing and Information Sciences, Radboud University Nijmegen, pp. 1-85, May 2019.

[14] S. Han, C. Qubo, and H. Meng, "Parameter selection in SVM with RBF kernel function," In Proceedings of World Automation Congress, pp. 1-4, Jun. 2012.

〈저 자 소 개〉

황 철 훈 (Cheol-Hun Hwang) 학생회원
2019년: 가천대학교 컴퓨터공학과(공학사)
2019~현재: 가천대학교 일반대학원 소프트웨어학과 석사과정
〈관심분야〉 정보보호, 머신러닝, 인공지능, 모바일

신 건 윤 (Gun-Yoon Shin) 학생회원
2017년: 가천대학교 인터랙티브 미디어 융합학과(공학사)
2018년: 가천대학교 일반대학원 컴퓨터공학과(공학학사)
2018~현재: 가천대학교 컴퓨터공학과 박사과정
〈관심분야〉 기계 학습, 악성코드 분석, 공격자 식별, 저자 분석, 인공지능

김 동 욱 (Dong-Wook Kim) 학생회원
2015년: 가천대학교 컴퓨터공학과(공학사)
2017년: 가천대학교 일반대학원 컴퓨터공학과(공학석사)
2017~현재: 가천대학교 컴퓨터공학과 박사과정
〈관심분야〉 Data Mining, 인공지능, Data fusion, Anomaly Detection

한 명 묵 (Myung-Mook Han) 종신회원
1980년: 연세대학교 공과대학(공학사)
1987년: 뉴욕공과대학교 대학원 컴퓨터공학과(공학석사)
1997년: 오사카시립대학교 대학원 정보공학부(이학박사)
1998~현재: 가천대학교 소프트웨어학과 교수
〈관심분야〉 정보보호, 인공지능