다중 서열 정렬 기법을 이용한 악성코드 패밀리 추천

Malware Family Recommendation using Multiple Sequence Alignment

저자
(Authors)

조인겸, 임을규
In Kyeom Cho, Eul Gyu Im

출처
(Source)

정보과학회논문지 43(3), 2016.3, 289-295 (7 pages)
Journal of KIISE 43(3), 2016.3, 289-295 (7 pages)

발행처
(Publisher)

한국정보과학회
KOREA INFORMATION SCIENCE SOCIETY

URL

http://www.dbpia.co.kr/Article/NODE06617071

APA Style

조인겸, 임을규 (2016). 다중 서열 정렬 기법을 이용한 악성코드 패밀리 추천. 정보과학회논문지, 43(3), 289-295.

ISSN 2383-630X(Print) / ISSN 2383-6296(Online)
Journal of KIISE, Vol. 43, No. 3, pp. 289-295, 2016. 3
http://dx.doi.org/10.5626/JOK.2016.43.3.289

다중 서열 정렬 기법을 이용한 악성코드 패밀리 추천

(Malware Family Recommendation using Multiple Sequence Alignment)

조 인 겸 †　　임 을 규 ††
(In Kyeom Cho)　　(Eul Gyu Im)

요 약 악성코드 개발자들은 악성코드 탐지를 회피하기 위하여 변종 악성코드를 유포한다. 정적 분석 기반의 안티 바이러스로는 변종 악성코드를 탐지하기 어려우며, 따라서 API 호출 정보 기반의 동적 분석이 필요하다. 본 논문에서는 악성코드 분석가의 변종 악성코드 패밀리 분류에 도움을 줄 수 있는 악성코드 패밀리 추천 기법을 제안하였다. 악성코드 패밀리의 API 호출 정보를 동적 분석을 통하여 추출하였다. 추출한 API 호출 정보에 다중 서열 정렬 기법을 적용하였다. 정렬 결과로부터 각 악성코드 패밀리의 시그니쳐를 추출하였다. 시그니쳐와의 유사도를 기준으로, 제안하는 기법이 새로운 악성코드의 패밀리 후보를 3개까지 추천하도록 하였다. 실험을 통하여 제안한 악성코드 패밀리 추천 기법의 정확도를 측정하였다.

키워드: 변종 악성코드, 행위 분석, 다중 서열 정렬, 악성코드 패밀리 분류

Abstract Malware authors spread malware variants in order to evade detection. It's hard to detect malware variants using static analysis. Therefore dynamic analysis based on API call information is necessary. In this paper, we proposed a malware family recommendation method to assist malware analysts in classifying malware variants. Our proposed method extract API call information of malware families by dynamic analysis. Then the multiple sequence alignment technique was applied to the extracted API call information. A signature of each family was extracted from the alignment results. By the similarity of the extracted signatures, our proposed method recommends three family candidates for unknown malware. We also measured the accuracy of our proposed method in an experiment using real malware samples.

Keywords: malware variants, behavior analysis, multiple sequence alignment, malware family classification

· 본 연구는 미래창조과학부 및 정보통신기술진흥센터의 대학ICT연구센터육성지원사업의 연구결과로 수행되었음(IITP-2015-H8501-15-1013).
· 이 논문은 2015 한국컴퓨터종합학술대회에서 '악성 코드 API에 대한 다중 서열 정렬 적용 연구'의 제목으로 발표된 논문을 확장한 것임

† 비 회 원 : 한양대학교 컴퓨터소프트웨어공학과
dlsrua1004@hanyang.ac.kr
†† 종신회원 : 한양대학교 컴퓨터공학부 교수(Hanyang Univ.)
imeg@hanyang.ac.kr
(Corresponding author임)

논문접수 : 2015년 7월 22일
(Received 22 July 2015)
논문수정 : 2015년 11월 5일
(Revised 5 November 2015)
심사완료 : 2015년 11월 13일
(Accepted 13 November 2015)

정보과학회논문지 제43권 제3호(2016. 3)

1. 서 론

악성코드 개발자들은 안티 바이러스 프로그램으로부터 악성코드를 보호하기 위하여 다형성 기법 등을 적용한 변종 악성코드를 제작하여 유포한다. 대부분의 안티 바이러스 프로그램은 정적 분석 기법[1-3]을 통한 악성코드 탐지를 수행한다. 그러나 정적 분석 기법은 변종 악성코드에 빠르게 대처하기가 어렵다는 한계가 있다. 이러한 한계점을 극복하기 위하여 동적 분석 기법[4-6]이 제시되었다.

동적 분석 기법은 악성코드가 실제로 동작하는 과정을 기록하고, 악성코드의 행위적 특성을 중점적으로 분석하는 기법이다. 일반적으로 동적 분석 기법의 결과로서 악성코드의 행위 정보, 즉 API 호출 정보 등을 얻을 수 있다. 동적 분석 기법을 이용하면 기존의 안티 바이러스 프로그램이 가지는 한계를 극복할 수 있다.

본 논문에서는 변종 악성코드를 탐지하고 분류하기 위하여 생정보학(Bioinformatics) 분야의 주요 기법 중 하나인 '다중 서열 정렬(Multiple Sequence Alignment)'[7,8] 기법을 적용하고자 한다. 다중 서열 정렬 기법은 다수의 서열들을 정렬하여 공통되는 부분들을 탐색하는데 쓰이는 기법이다. 동적 분석 기법을 통하여 같은 패밀리에 해당하는 변종 악성코드들의 API 호출 정보를 추출하고, 다중 서열 정렬 기법을 이용하여 각 악성코드에서 공통적으로 존재하는 API 호출 정보를 확인할 수 있다. 최종적으로 각 악성코드 패밀리 변종 악성코드의 공통 API 호출 정보 기반의 시그니쳐를 정의할 수 있으며, 새롭게 발견된 변종 악성코드를 탐지하고 분류하는데 이용할 수 있다. 본 논문에 기술한 연구의 목적은 변종 악성코드의 패밀리로 예상되는 3개의 패밀리를 추천하고, 이를 통하여 악성코드 분석가로 하여금 더욱 효율적인 악성코드 분석이 가능하도록 하는 것이다.

본 논문은 다음과 같이 구성하였다. 2장에서는 본 논문과 관련된 연구들에 대하여 소개하였다. 이어서 3장에서 본 논문에서 이용하고자 하는 다중 서열 정렬 기법에 대하여 설명하였다. 4장에서는 본 논문에서 제시하는 기법에 대하여 기술하였으며, 5장에서 제시한 기법을 이용한 악성코드 분류 실험을 실제 악성코드를 대상으로 수행한 결과를 제시하였다. 최종적으로 6장에서 본 논문의 결론을 기술하였다.

2. 관련 연구

악성코드 분석을 위하여 다중 서열 정렬 기법을 이용한 몇 가지 연구 결과가 존재하였다. Youngjoon Ki 외 2인은 다중 서열 정렬 알고리즘인 Clustal [9]을 구현한 도구인 Clustal X [10]를 사용하여 악성코드 API 호출 정보를 분석하고, 악성코드 분류를 수행하였다[11]. ClustalX를 이용하기 위하여 악성코드가 호출하는 모든 API를 26가지의 카테고리로 추상화하였으며, 다중 서열 정렬을 이용하여 생성한 API 시그니쳐를 새로운 프로그램을 분류하는데 사용하였다. 그러나 본 연구와 달리, 해당 연구에서의 API의 추상화 과정에서 손실되는 정보가 존재할 수 있으며 추가적인 검증이 필요하다. 또한 본 연구의 선행 연구로서, 서열 정렬 기법을 API 호출 정보 유사도 분석에 적용한 연구를 수행한 바 있다[12]. 해당 연구에서는 API 호출 정보에서의 반복문을 제거하여 서열 정렬 알고리즘의 성능 문제를 해결하였으며, 서열 정렬 기법을 통하여 악성코드 간 유사도 분석이 가능함을 보였다. 그러나 해당 연구 결과는 성능 문제가 여전히 존재하며, 실제 분류 및 탐지에 이용하기 어렵다는 한계를 가진다. Vinod P. 외 3인은 기존 정적 분석의 한계점으로 난독화 등의 기법에 대해 취약함을 지적하며, 이를 극복하기 위하여 다중 서열 정렬 기법의 적용을 제안하였다[13]. 악성코드로부터 명령어 정보를 정적 분석을 통해 추출하고, 이를 서열의 형태로 구성하여 다중 서열 정렬 기법을 적용하였다. 정렬 결과를 기반으로 하여 악성코드 탐지 척도를 정의하였고, 기존의 안티 바이러스 프로그램에 비해 성능이 뛰어남을 보였다. 본 연구가 행위적 특성에 주목한 것과 달리, 위의 두 연구는 악성코드의 정적 정보에 의존적이므로, 난독화 등의 기법에 매우 취약할 것으로 보인다.

악성코드 API 호출 정보를 이용한 악성코드 분석 기법에 관한 연구 역시 존재한다. Chun-I Fan 외 3인은 최근의 악성코드의 자신의 정적 및 동적 시그니쳐를 은닉하는 특징에 따른 새로운 악성코드 탐지 기법의 필요성을 주장하였다[14]. 이들은 동적 분석을 통하여 악성코드의 API 호출 정보를 확인하고, 또한 은닉 행위 정보를 확인하였다. 확인한 정보를 기반으로 악성코드와 정상 프로그램을 구분하기 위한 표현 모델을 생성하였으며, 표현 모델의 수를 최소화하였다. 최종적으로 80개의 표현 모델을 이용하여 95%의 탐지 정확도를 보였다. 연구 결과는 우수하나, 본 연구가 지향하는 악성코드 분류와 달리 탐지만을 위한 연구이므로, 패밀리 분류를 위해서 추가적인 연구가 필요하다. Kyung-Soo Han 외 2인은 API 호출 정보의 순차적 특성을 이용한 변종 악성코드 분류 기법을 제시하였다[15]. 악성코드의 IAT로부터 사용할 수 있는 API 정보를 분석한 뒤, 사전에 정의한 정상 프로그램에서 자주 호출하는 API를 제외한 나머지 API 호출 정보를 포함하는 서열을 얻을 수 있었다. 얻은 두 악성코드의 API 서열을 비교하여 변종 악성코드 탐지에 이용하였다. Ashkan Sami 외 5인은 변종 악성코드로 인한 피해에 대한 대응책으로서 API

호출 정보기반 악성코드 탐지 기법을 제안하였다[16]. API 호출 정보로부터 특성 벡터를 생성하고, 분류기 학습을 통해 악성코드 탐지를 수행하였다. 두 연구 모두 API 호출 정보를 이용하였으나, 이는 프로그램이 사용하는 API의 정보이며 실제 동작 중에 호출한 API 정보에 해당하지 않는다. 이러한 정적 정보 이용은 본 연구에서와 달리 동작 과정에서의 정보를 고려하지 않았다는 한계가 있다.

3. 다중 서열 정렬

서열 정렬 기법은 생정보학에서 널리 쓰이는 주요 기법 중 하나로, 단백질, DNA, RNA 등의 생물학적 정보를 표현하는 서열을 정렬하고, 이를 통한 각 서열 간 관계 파악을 목적으로 한다.

다중 서열 정렬 기법은 이러한 생물학적 정보를 담은 서열 3개 이상에 대하여 적용할 수 있는 서열 정렬 기법이다. 일반적으로 다중 서열 정렬 기법의 입력으로 진화적 관점에서의 유사성이 존재하는 서열들을 사용하며, 입력 서열들의 정렬 결과를 얻을 수 있다. 정렬 결과는 모든 서열들의 유사도가 최대가 되는 '최적의(Optimal)' 상태를 이루게 된다. 정렬 과정에서 필요에 따라 각 입력 서열에 공백(gap)을 추가한다. 이러한 공백의 추가는 기존의 LCS(Longest Common String) 탐색 방식과의 큰 차이점이다.

다중 서열 정렬 기법에 대한 가장 간단한 접근은 쌍 서열 정렬 기법을 복수의 서열에 대해 적용할 수 있도록 확장하는 것이다. 쌍 서열 정렬 기법은 오직 두 개의 서열에 대하여 서열 정렬을 수행하는 것이다. 쌍 서열 정렬 기법은 일반적으로 두 서열의 길이를 곱한 만큼의 크기를 가지는 2차원 행렬을 구성하게 된다. 이러한 행렬 구성을 다중 서열 정렬 기법을 위하여 확장할 수 있다. 서열의 개수가 N개인 경우, 행렬의 차원 역시 N차원이어야 한다. 따라서 다중 서열 정렬 기법의 시간 복잡도는 지수적으로 증가한다. 이미 기존 연구 결과에서 다중 서열 정렬 기법이 NP-complete 문제임을 증명하였다[17-23].

이러한 지수적 증가를 보이는 다중 서열 정렬의 시간 복잡도는 대량의 서열을 대상으로 하는 경우 그 성능이 매우 좋지 않음을 의미한다. 이를 해결하기 위한 여러 시도 중 가장 대표적인 것으로 점진적 기법이 있다[20]. 이는 휴리스틱 탐색을 이용한 다중 서열 정렬 결과를 얻는 접근법이다. 반복적인 쌍 서열 정렬 기법을 수행하고, 각 정렬 결과를 조합하여 최종적인 다중 서열 정렬 결과의 근사값을 얻는다. 쌍 서열 정렬 기법은 가장 유사한 두 서열부터 시작하여 가장 거리가 있는 서열까지 수행한다.

점진적 기법을 통한 다중 서열 정렬 기법은 크게 2단계로 나눌 수 있다. 먼저 그림 1의 왼쪽과 같이 각 서열의 관계를 나타내는 트리 구조를 형성한다. 이를 가리켜 계통수(phylogenetic tree)라고 하며, 쌍 서열 정렬을 어느 서열부터 수행할 것인지를 정하는데 사용한다. 따라서 이 트리 구조를 가리켜 '가이드 트리(guide tree)'라고도 한다. 계통수를 형성하기 위하여 UPGMA[21]과 같은 Neighbor-Joining 알고리즘을 사용할 수 있으며, 각 서열 간 유사도에 따른 거리를 기준으로 계통수 형성이 가능하다[22]. 이어서 그림 1의 오른쪽에서와 같이 형성한 계통수를 기준으로 쌍 서열 정렬을 수행하며 정렬 결과를 확장시켜 나간다.

점진적 기법을 통한 다중 서열 정렬 기법의 알고리즘으로 Clustal, T-Coffee[23]이 대표적이다. Clustal 알고리즘이 상대적으로 빠른 속도를 보이지만, 전체적으로 유사하지 않은 서열들에 대한 정렬 정확도는 T-Coffee

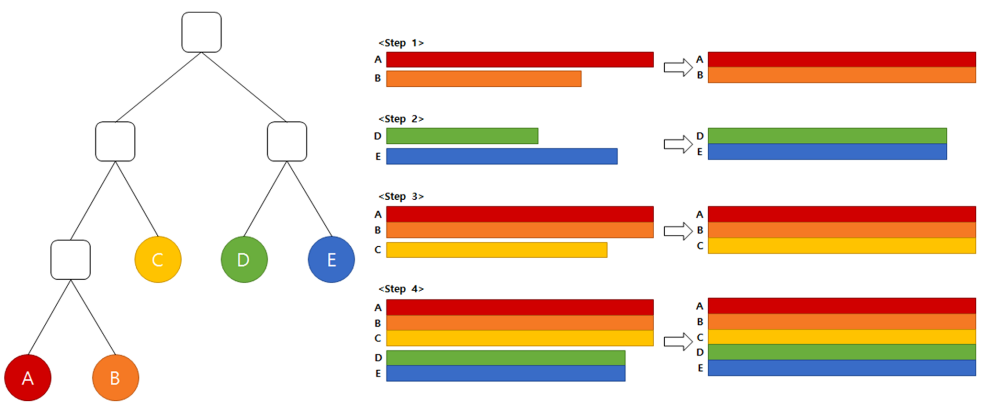

그림 1 점진적 기법을 통한 다중 서열 정렬
Fig. 1 Progressive multiple sequence alignment

알고리즘이 더욱 정확하다. Clustal 알고리즘이 가장 대중적이며, 특히 Clustal 알고리즘을 구현한 프로그램 중 최신 버전인 Clustal Omega [24]는 오픈 소스로 제공된다.

4. 제안하는 방법

본 논문에서는 변종 악성코드의 패밀리 정보 추천을 위한 기법을 제시하였다. 제안한 방법은 하나의 악성코드 패밀리의 각 샘플들에 대하여 동적 분석을 수행하고, API 호출 정보를 추출한다. 추출한 API 호출 정보를 다중 서열 정렬이 적용 가능한 형태로 변환한 뒤, 다중 서열 정렬 결과로부터 시그니쳐를 추출한다. 패밀리 정보를 알 수 없는 악성코드의 API 호출 정보를 각 악성코드 패밀리의 시그니쳐와 비교를 수행하여 유사도를 계산하였다. 계산한 유사도를 기준으로 가장 높은 유사도를 보인 3개 시그니쳐의 패밀리를 해당 악성코드의 패밀리 후보로서 추천하였다.

전체적인 과정은 그림 2와 같다. 이어지는 세부 절에서 시그니쳐를 추출하고, 악성코드의 패밀리 후보를 추천하는 과정을 설명하였다.

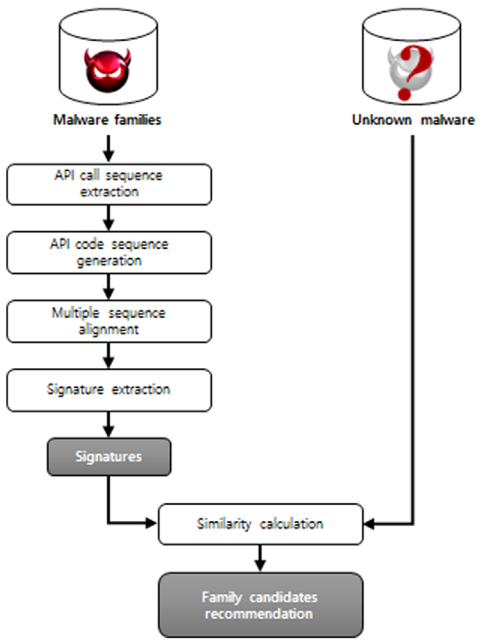

그림 2 제안하는 방법 개관
Fig. 2 Overview of the proposed method

4.1 API 호출 서열 추출

오픈소스 샌드박스 도구인 Cuckoo Sandbox [25]를 사용하여 동적 분석을 수행하고, 분석 결과 중 API 호출 정보만을 추출하였다. 대상 프로그램이 실행되는 도중에 호출되는 API들의 이름을 호출 순서대로 나열하여 API 호출 서열을 형성하였다.

4.2 API 코드 서열 생성

API 호출 서열은 각 API의 이름으로 이루어져 있으므로, 곧바로 다중 서열 정렬 기법을 적용하기에 부적절하다. 다중 서열 정렬 알고리즘은 입력으로서 문자열의 형태를 가진 서열을 요구한다. 즉 입력 서열은 고정 길이를 가지는 단위체로 이루어진 서열이어야 한다. API의 이름은 각 API마다 그 길이가 다양하므로, 고정 길이를 가지는 코드를 각 API마다 부여하여 새로운 서열을 생성해야 한다.

API 코드는 다음과 같이 정의하였다. 코드는 3자리의 알파벳으로 구성하였다. 첫 번째 알파벳은 각 API의 카테고리를 나타낸다. 카테고리는 총 13가지로, A부터 M까지 각각 코드를 부여하였다. 아래 표 1에 각 카테고리의 이름과 코드를 나열하였다. 나머지 알파벳들은 카테고리 내부의 순서를 나타낸다. 가장 처음의 API는 'AA'이며, 이후 'AB', 'AC', ... 순서이다. 예를 들어 2번째 카테고리의 5번째 API의 경우 대응하는 코드는 'BAE'이다. 13번째 카테고리의 27번째 API의 코드는 'MBA'이다.

표 1 API 카테고리 별 코드
Table 1 Codes for API categories

Category	Code	Category	Code
Registry	A	System	H
File System	B	Device	I
Process	C	Threading	J
Service	D	Hooking	K
Network	E	Windows	L
Socket	F	Misc.	M
Synchronization	G	-	-

API 코드를 이용하여 서열을 새로이 생성하는 과정에서, 추가적인 반복 부분 제거 처리를 적용하였다. 이러한 반복 부분은 프로그램이 가지는 루프 때문에 나타나는 짧은 API 호출 정보들을 가리킨다. 이러한 반복 부분은 제거하여도 분류 정확도에 큰 영향을 미치지 않는다. 그러나 경우에 따라 반복 부분이 지나치게 많은 경우에는 API 호출 정보의 양이 서열 정렬 기법을 적용하기에 너무 방대해지는 문제가 있다. 따라서 올바른 분석을 위하여 1개 혹은 2개의 반복되는 API 호출 정보를 제거하였다. 이러한 반복 부분 제거를 통하여 'AABABCC'와 같은 서열의 경우 'ABC'와 같이 그 길이를 단축할 수 있다.

4.3 다중 서열 정렬

다중 서열 정렬 기법은 본래 단백질이나 DNA 같은

유기체 정보를 대상으로 수행하도록 고안되었다. 단백질의 경우 20가지의 아미노산으로, DNA의 경우 4가지의 핵산으로 구성된다. 따라서 내부적으로 처리하는 경우 알파벳 하나로 표현이 가능하며, 입출력 또한 알파벳 하나로 이루어진 서열을 대상으로 동작하도록 구현하는 것이 일반적이다.

API 호출 정보는 동적 분석 결과에 따라 그 개수가 달라지나, 일반적으로 약 200 종류의 API로 이루어진다. 이는 단일 알파벳으로 모든 API 정보를 표현할 수 없음을 의미한다. 따라서 API 코드로 이루어진 서열의 형태로 입출력을 처리할 수 있도록 알고리즘의 수정이 필요하다.

앞서 언급한 바와 같이, Clustal Omega는 오픈소스의 형태로 이용할 수 있다. 내부 소스를 수정하여 API 코드에 대하여 동작할 수 있도록 하였다. 수정한 Clustal Omega를 이용하여 주어진 프로그램들에 대한 다중 서열 정렬 결과를 얻을 수 있다.

다중 서열 정렬 결과로서 두 가지의 결과물을 얻을 수 있다. 첫 번째는 주어진 악성코드 패밀리의 API 코드 서열 집합에 다중 서열 정렬을 수행한 정렬 결과물이다. 정렬 결과는 길이가 동일한 정렬된 서열들로 이루어지며, 행렬의 형태를 이룬다. 각 행은 정렬된 서열, 각 열은 API 코드 또는 공백이다. 두 번째로는 계통수가 있다. Clustal Omega는 필요에 따라 정렬 도중 생성하는 계통수를 파일로 출력한다.

4.4 시그니쳐 추출

시그니쳐 추출은 크게 두 단계로 나누어진다. 먼저 그림 3에 나타낸 것과 같이 계통수를 이용하여 각 API 코드 서열을 복수의 세부 그룹으로 분할한다. 이 때 세부 그룹으로 분할하는 기준은 세부 그룹의 개수가 가장 많은 계통수의 깊이로 정의하였다.

이어서 각 세부 그룹에 속하는 모든 API 코드 서열에 대하여 다중 서열 정렬을 각각 수행한다. 결과로서 얻어진 다중 서열 정렬 결과 역시 행렬의 형태를 이룬다. 정렬 결과의 각 열에 대하여, 공백을 제외한 API 코드 중 가장 많이 나타난 API 코드를 대표 API 코드로서 추출한다. 모든 열에서 추출한 대표 API 코드를 병합하여 새로운 API 코드 서열을 형성하고, 이를 해당 세부 그룹의 대표 API 코드 서열로서 정의하였다.

하나의 패밀리가 N개의 세부 그룹으로 나누어지는 경우, 패밀리가 가지는 대표 API 코드 서열 역시 N개이다. 이러한 모든 대표 API 코드 서열을 종합하여 해당 패밀리의 시그니쳐로서 정의하고 이용하였다.

4.5 악성코드 패밀리 추천

악성코드의 패밀리 후보를 추천하기 위하여 미리 생성된 각 패밀리 별 시그니쳐와의 비교를 수행한다. 변종

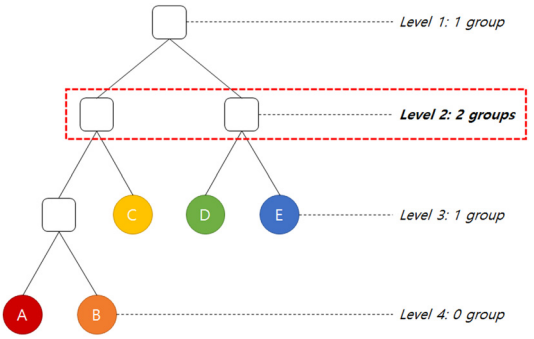

그림 3 계통수를 이용한 세부 그룹 분할
Fig. 3 Phylogenetic tree grouping

악성코드 역시 동일한 과정을 통해 API 코드 서열을 생성하였고, 쌍 서열 정렬을 통한 시그니쳐와의 비교를 통하여 유사도를 계산하였다. 유사도는 아래 식 (1)과 같이 정의하였다.

$$similarity = \frac{matched - unmatched}{length} \quad (1)$$

수식에서 *matched*는 두 API 코드 서열의 정렬 결과에서 같은 위치에 나타난 동일한 API 코드 쌍의 개수이며, *unmatched*는 같은 위치에 존재하지만 일치하지 않는 API 코드 쌍의 개수이다. 두 서열에서 공백과 API 코드가 같은 위치에 존재하는 경우는 무시하였다. *length*는 전체 비교 길이이다. 유사도는 -1부터 1까지의 값을 가지며, 1에 가까울수록 두 서열은 유사한 것이다.

각 패밀리는 복수의 대표 API 코드 서열을 시그니쳐로서 가질 수 있다. 시그니쳐에 대한 유사도를 정의하기 위하여, 변종 악성코드를 각 패밀리의 모든 대표 API 코드 서열과의 유사도를 계산한 후, 그 중 가장 높은 유사도 값을 해당 패밀리의 시그니쳐에 대한 유사도로서 정의하였다.

각 패밀리의 시그니쳐와의 유사도를 기준으로, 최상위 3개의 패밀리를 선정한다. 이렇게 선정한 3개 패밀리는 악성코드 분석가로 하여금 해당 변종 악성코드가 해당될 것으로 예상되는 패밀리의 범위를 좁히고, 더욱 효율적으로 악성코드를 분석하는데 도움을 줄 수 있다.

5. 실 험

본 논문에서 제안한 추천 기법을 통하여 실제 변종 악성코드와 패밀리를 대상으로 한 정확도 측정 실험을 수행하였다.

실험을 위하여 총 15개 패밀리, 전체 1,554개의 악성코드를 사용하였다. 악성코드는 VxHeaven [26] 등에서 수집하였다. 정확도 측정을 위하여 각 악성코드 패밀리를 각각 트레이닝 셋과 테스트 셋으로 나누었다. 각 집

합의 비율은 7:3이다. 각 패밀리의 트레이닝 셋은 시그니쳐를 추출하는데 사용하였으며, 테스트 셋의 악성코드 샘플들의 API 코드 서열과의 비교를 통하여 각 시그니쳐의 패밀리 추천 정확도를 측정하였다.

정확도는 다음과 같이 정의하였다. 분석 대상 악성코드의 본래 패밀리가 패밀리 예상 결과 중 상위 3위 내에 존재하는지를 확인한다. 존재하는 경우 해당 척도를 이용하여 악성코드의 패밀리 추천이 올바르게 이루어진 것으로, 반대로 존재하지 않는 경우 추천이 부정확한 것으로 정의하였다.

실험은 총 10번 반복 수행하였다. 각 실험마다 트레이닝 셋과 테스트 셋을 임의로 분할하여 수행하였다.

5.1 동적 분석

모든 악성코드 샘플에 대하여 Cuckoo Sandbox를 이용하여 동적 분석을 수행하였다. Cuckoo Sandbox의 분석 환경은 Windows XP SP3, 분석 시간은 120초로 제한하였다.

5.2 패밀리 추천 정확도

임의로 분할한 트레이닝 셋과 테스트 셋을 대상으로, 총 10번의 반복 실험을 수행하였다. 각 패밀리의 시그니쳐에 대하여 패밀리 추천 정확도를 측정하고, 각 실험에서의 추천 정확도의 평균을 그림 4와 같이 그래프로 나타내었다.

평균적으로 78%의 정확도를 얻을 수 있었다. 일부 패밀리에 대하여 상대적으로 낮은 정확도를 얻을 수 있었으며, 대체로 75% 이상의 정확도를 보였다.

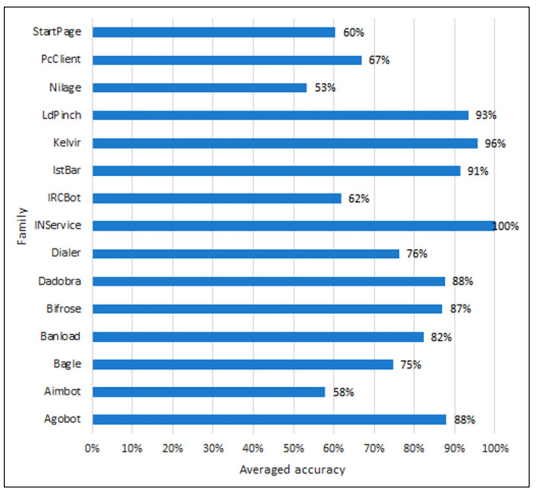

그림 4 정확도 결과
Fig. 4 Accuracy results

5.3 실험 결과 토의

일부 패밀리를 대상으로 한 실험에서 얻었던 낮은 정확도 결과에 대하여, 낮은 정확도의 원인에 대하여 분석을 수행하였다.

낮은 정확도의 원인은 여러 가지가 있을 수 있으나, 가장 큰 원인으로서 패밀리 분류 기준의 차이점이 있다. 실험에 사용한 악성코드 패밀리 정보는 Virus Total[27]에서의 Kaspersky 진단명을 이용하였다. Virus Total에서는 여러 개의 안티 바이러스의 분석 결과를 제공하며, 따라서 악성코드의 패밀리 정보는 정적 분석에 크게 의존적이다. 이와 달리 제안한 방법은 동적 분석에 의존적이므로, 패밀리 분류 결과 자체가 달라질 수 있다.

일부 악성코드 패밀리의 경우 정적 분석 결과에서 충분히 같은 패밀리로 분류될 수 있을 정도로 유사성이 존재하나, 행위적인 측면에서는 유사도가 다소 떨어질 수 있다. 이러한 경우 본 논문에서 제안한 기법을 통한 정확한 패밀리 분류 및 추천이 올바르게 수행되지 않을 수 있다.

따라서 자체적인 API 호출 정보 기반의 변종 악성코드 클러스터링 기법이 필요하다. 이는 기존의 클러스터링과 유사한 것으로, 서열 정렬 기법으로 얻을 수 있는 악성코드 간 거리 척도의 정의를 통하여 수행할 수 있다. 이렇게 얻어진 각 클러스터를 통하여 기존 악성코드의 패밀리 분류와의 차이를 확인할 수 있으며, 본 논문에서 제안하는 방법을 통한 패밀리 분류 및 측정 기법의 정확도를 재확인 할 수 있을 것이다.

6. 결 론

본 논문에서는 변종 악성코드에 대한 패밀리 추천 기법을 제안하였다. 패밀리 추천을 위하여 다중 서열 정렬 기법을 악성코드 API 호출 정보에 대하여 적용하였으며, 계통수와 다중 서열 정렬 결과를 이용한 시그니쳐 추출을 수행하였다. 시그니쳐와 변종 악성코드 API 호출 정보 간 유사도를 기반으로 상위 3개의 패밀리를 악성코드 분석가에게 추천할 수 있도록 하였으며, 실제 실험을 통하여 각 패밀리에 대한 추천 정확도를 측정하였다.

제안한 기법의 한계점으로 일부 패밀리에 대한 낮은 정확도 결과가 있었다. 이는 정적 분석 결과와 동적 분석 결과의 차이에서 비롯된 것으로, 후속 연구를 통하여 이러한 한계점을 극복할 수 있을 것으로 보인다. 패밀리의 분류 기준 자체를 다시 정의하고, 사전 클러스터링을 통한 자체적인 패밀리 분류 기법을 이용하여 API 호출 정보 기반의 악성코드 분류 시스템을 구현할 계획이다.

References

[1] D. Bilar, "Opcodes as predictor for malware," *International Journal of Electronic Security and Digital Forensics*, Vol. 1, No. 2, pp. 156-168, Jan.

2008.
[2] I. Santos, Y. Penya, J. Devesa, P. Bringas, "N-grams-based File Signatures for Malware Detection," *Proc. of ICEIS '09*, pp. 317-320, 2009.
[3] S. Tabish, M. Shafiq, M. Farooq, "Malware detection using statistical analysis of byte-level file content," *Proc. of the ACM SIGKDD Workshop on Cyber-Security and Intelligence Informatics*, pp. 23-31, 2009.
[4] C. Willems, T. Holz, F. Freiling, "Toward automated dynamic malware analysis using cwsandbox," *IEEE Security & Privacy*, Vol. 5, No. 2 pp. 32-39, Mar./Apr. 2007.
[5] M. Alazab, S. Venkataraman, P. Watters, "Towards understanding malware behaviour by the extraction of API calls," *Proc. of Cybercrime and Trustworthy Computing Workshop (CTC)*, pp. 52-59, 2010.
[6] M. Siddiqui, M. Wang, J. Lee, "A survey of data mining techniques for malware detection using file features," *Proc. of the 46th Annual Southeast Regional Conference on XX*, pp. 509-510, 2008.
[7] D. J. Bacon, W. F. Anderson, "Multiple sequence alignment," *Journal of molecular biology*, Vol. 191, No. 2, pp. 153-161, Sep. 1986.
[8] R. C. Edgar, S. Batzoglou, "Multiple sequence alignment," *Current opinion in structural biology*, Vol. 16, No. 3, pp. 368-373, Jun. 2006.
[9] D. Higgins, P. Sharp, "CLUSTAL: a package for performing multiple sequence alignment on a microcomputer," *Gene*, Vol. 73, No. 1, pp. 237-244, Dec. 1988.
[10] Clustal X, http://www.clustal.org/clustal2/
[11] Y. Ki, E. Kim, H. K. Kim, "A Novel Approach to Detect Malware Based on API Call Sequence Analysis," *International Journal of Distributed Sensor Networks*, Vol. 2015, 2015.
[12] I. K. Cho, T. G. Kim, Y. J. Shim, H. Park, B. Choi, E. G. Im, "Malware Similarity Analysis using API Sequence Alignments," *Journal of Internet Services and Information Security (JISIS)*, Vol. 4, No. 4, pp. 103-114, 2014.
[13] P. Vinod, V. Laxmi, M. Gaur, G. Chauhan, "MOMENTUM: metamorphic malware exploration techniques using MSA signatures," *Proc. of Innovations in Information Technology (IIT)*, pp. 232-237, 2012.
[14] C. I. Fan, H. W. Hsiao, C. H. Chou, Y. F. Tseng, "Malware Detection Systems Based on API Log Data Mining," *Proc. of Computer Software and Applications Conference (COMPSAC)*, pp. 225-260, 2015.
[15] K. S. Han, I. K. Kim, E. G. Im, "Malware family classification method using API sequential characteristic," *Journal of Security Engineering*, Vol. 8, No. 2, pp. 607-611, Dec. 2011.
[16] A. Sami, B. Yadegari, H. Rahimi, N. Peiravian, S. Hashemi, A. Hamze, "Malware detection based on mining API calls," *Proc. of the 2010 ACM Symposium on Applied Computing*, pp. 1020-1025, 2010.
[17] L. Wang, T. Jiang, "On the complexity of multiple sequence alignment," *Journal of computational biology*, Vol. 1, No. 4, pp. 337-348, WINTER 1994.
[18] W. Just, "Computational complexity of multiple sequence alignment with SP-score," *Journal of computational biology*, Vol. 8, No. 6, pp. 615-23, Nov. 2001.
[19] I. Elias, "Settling the intractability of multiple alignment," *Journal of Computational Biology*, Vol. 13, No. 7, pp. 1323-1339, Sep. 2006.
[20] P. Hogeweg, B. Hesper, "The alignment of sets of sequences and the construction of phyletictrees: an integrated method," *Journal of molecular evolution*, Vol. 20, No. 2, pp. 175-186, Jun. 1984.
[21] D. Mount, *Bioinformatics: Sequence and Genome Analysis*, 2nd Ed., Cold spring harbor laboratory press, New York, 2001.
[22] P. Legendre, L. F. Legendre, *Numerical Ecology*, 24th Ed., Elsevier, 2012.
[23] C. Notredame, G. Higgins, J. Heringa, "T-Coffee: A novel method for fast and accurate multiple sequence alignment," *Journal of molecular biology*, Vol. 302, No. 1, pp. 205-217, Sep. 2000.
[24] Clustal Omega, http://www.clustal.org/omega/
[25] Cuckoo Sandbox, http://cuckoosandbox.org/
[26] VxHeaven, http://vxer.org/
[27] Virus Total, http://www.virustotal.com/

조 인 겸

2014년 한양대학교 컴퓨터공학부 졸업(학사). 2014년~현재 한양대학교 컴퓨터·소프트웨어공학과 석사과정. 관심분야는 컴퓨터보안, 악성코드 분석

임 을 규

1992년 서울대학교 컴퓨터공학과 졸업(학사). 1994년 서울대학교 컴퓨터공학과 졸업(석사). 2002년 University of Southern California Computer Science Dept. 졸업(박사). 2005년~현재 한양대학교 컴퓨터공학부 부교수. 관심분야는 제어시스템 보안, 악성코드, 정보보호, 소프트웨어 취약 점검