그래프 기반 멀웨어 탐지

장민희*, 김상욱*, 하지운*, 조성제**
*한양대학교 전자컴퓨터통신공학과
**단국대학교 소프트웨어학과
e-mail:zzmini@agape.hanyang.ac.kr

Graph-Based Malware Detection

Min-Hee Jang*, Sang-Wook Kim*, Ji-Woon Ha*, Seong-Je Cho**
*Dept. of Electronics and Computer Engineering, Hanyang University
**Dept. of Computer Science and Engineering, Dankook University

요 약

최근 들어, 컴퓨터에 악영향을 미치고자 하는 목적으로 개발된 멀웨어들이 크게 증가하고 있다. 이러한 멀웨어들은 자신들의 변종을 생성함으로써 안티 멀웨어 프로그램들의 탐지에서 벗어나고자 한다. 본 논문에서는 멀웨어들의 변종을 자동으로 탐지하기 위한 기법들 중 그래프 기반 기법에 대해 논의한 후 그 기법의 대표적인 연구들을 소개한다. 그 후 그래프 기반 멀웨어 탐지 시 고려해야 될 사항들에 대해 설명한다. 이러한 논의를 통해 효율적으로 멀웨어를 탐지하기 위한 기술을 고안하는데 중요한 실마리를 제공할 수 있을 것이다.

1. 서론

멀웨어(malicious software, malware)란, 컴퓨터 시스템을 파괴하거나 컴퓨터 사용자의 정보를 유출하고자 하는 목적으로 제작된 악의적인 소프트웨어로서 최근 들어 이러한 멀웨어들이 크게 증가하고 있는 추세이다.[1]. 멀웨어들은 V3나 아바스트 같은 안티 멀웨어 프로그램들의 탐지에서 벗어나기 위해 자신의 변종을 생성한다[2]. 코드 수정이나 거짓 함수 호출과 같은 방법을 통해 멀웨어의 변종들이 다양하게 생성될 수 있기 때문에 현실적으로 전문가를 통해 일일이 프로그램들을 분석해가며 멀웨어를 탐지하기는 매우 어렵다[2].

이 문제를 해결하기 위해 멀웨어를 자동으로 탐지하기 위한 다양한 기법들이 연구되었다[1][2][3]. 이러한 기법들은 멀웨어를 분석하여 그 멀웨어를 대표하는 특징인 시그니처(signiture)를 자동으로 추출한다. 그 후, 추출된 시그니처를 기반으로 프로그램들을 분석하여 멀웨어와의 유사성을 측정함으로써 자동으로 멀웨어를 탐지한다.

최근 들어, 멀웨어의 변종들을 탐지하기 위한 기법으로 그래프 기반 기법들이 연구되고 있다[3][4]. 이러한 기법들은 프로그램을 노드와 간선들로 구성된 그래프로 표현한 후 그래프 간의 유사도를 비교함으로써 멀웨어의 변종을 쉽게 탐지할 수 있다[3].

본 논문에서는 멀웨어 탐지 기법들 중 그래프 기반 멀웨어 탐지 기법들에 대해 소개한다. 멀웨어 탐지 기법은 동적 접근법(dynamic approach)와 정적 접근법(static approch)으로 분류할 수 있다. 동적 접근법은 프로그램을 실제로 실행시켜 실행 패턴을 기반으로 멀웨어를 탐지하는 방법이고, 정적 접근법은 프로그램 코드 자체를 기반으로 멀웨어를 탐지하는 방법이다. 본 논문에서는 각 접근법들의 대표적인 기술에 대해 서술한 후 멀웨어 탐지에 대한 고려 사항에 대해 논의한다.

2. 동적 접근법

동적 접근법은 멀웨어를 실행시켜 해당 프로그램의 시스템 호출(system call)이나 함수 호출(function call)을 기반으로 멀웨어를 탐지한다[3]. 즉, API 호출이나 파일, 메모리 수정 등의 정보를 통하여 프로그램이 컴퓨터에 부정적인 영향을 미치는지 파악함으로써 멀웨어를 탐지하는 것이다. 실제로 프로그램을 실행시켜 분석해야 하기 때문에 멀웨어를 안전하게 실행시킬 수 있는 샌드박스(sandbox)와 같은 에뮬레이터가 요구된다[3].

동적 접근법에서 그래프를 기반으로 멀웨어를 탐지하는 대표적인 방법으로는 코드 그래프 시스템[3]이 있다. 이 방법은 프로그램을 그래프로 모델링하기 위하여 시스템 호출간의 관계를 기반으로 방향성 그래프인 콜 그래프를 생성한다. 즉, 하나의 시스템 콜을 노드로, 시스템 콜 간의 관계를 간선으로 표현함으로써 프로그램을 그래프로 모델링하는 것이다. 그 후 좀 더 정확한 분석을 위해 코드 그래프 분석기를 이용하여 콜 그래프에서 코드 그래프를 추출한다.

코드 그래프 시스템은 이러한 방법을 통해 멀웨어들을 대상으로 코드 그래프를 생성하여 멀웨어 코드 그래프 풀

(pool)을 구축한다. 그 후 컴퓨터로 들어오는 프로그램들을 대상으로 코드 그래프를 추출한 후 풀에 있는 코드 그래프들과 유사도를 비교하여 멀웨어와 그 변종들을 탐지한다. 그래프 간의 유사도는 비교하고자 하는 두 그래프의 합집합 노드 수 분의 교집합 노드 수로 계산한다. 그림 1은 전체적인 흐름을 나타낸다.실제 멀웨어들을 대상으로 코드 그래프 시스템을 실험한 결과, 약 91% 정도의 정확도로 멀웨어들을 탐지하는 것으로 나타났다[3].

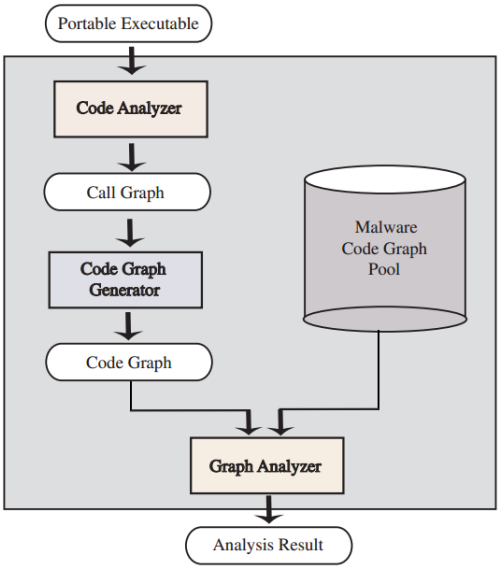

그림 1. 코드 그래프 시스템의 전체적인 흐름.

3. 정적 접근법

정적 접근법은 멀웨어의 프로그램 코드나 바이너리를 분석하여 멀웨어를 탐지한다. 즉, 역탐지 기법을 통해 프로그램을 역컴파일한 후 생성된 코드나 바이너리를 통해 컴퓨터에 악영향을 미치는 부분이 있는지 파악함으로써 멀웨어를 탐지하는 것이다. 패킹된 코드일 경우 패킹을 풀기 위한 역어셈블링과 같은 기술이 필요하다[4].

정적 접근법 중, 그래프를 기반으로 멀웨어를 탐지하는 대표적인 기법으로는 control graph flow (CFG) 기반 시스템[4]이 있다. 이 방법은 프로그램 코드를 분석하여 CFG로 모델링한다. CFG란, 프로그램 코드에서 점프(jump)가 있을 시 그 점프 전까지의 코드를 하나의 노드로, 그 점프를 간선으로 모델링한 그래프이다.

CFG 기반 시스템은 생성된 CFG의 노드들을 깊이 우선 순위로 정렬한 후 string형태의 시그니처를 추출한다. 추출된 멀웨어들의 시그니처를 데이터베이스에 저장한 후 프로그램이 컴퓨터로 들어오면 데이터베이스 안의 멀웨어 시그니처들과 프로그램의 시그니처 유사도를 계산한다. 유사도 계산 시 Dice coefficient를 이용한다[4]. Dice coefficent d는 특정 집합 X와 Y가 있을 때, 식 (1)과 같이 표현된다.

$$d = \frac{2|X \cap Y|}{|X| + |Y|} \quad \cdots\cdots\cdots\cdots\cdots\cdots (1)$$

그림 2는 CFG 기반 시스템의 전체적인 흐름을 나타낸다. 실제 멀웨어들을 대상으로 CFG 기반 시스템을 실험한 결과, 약 95%의 정확도로 멀웨어를 탐지하는 것으로 나타났다[4].

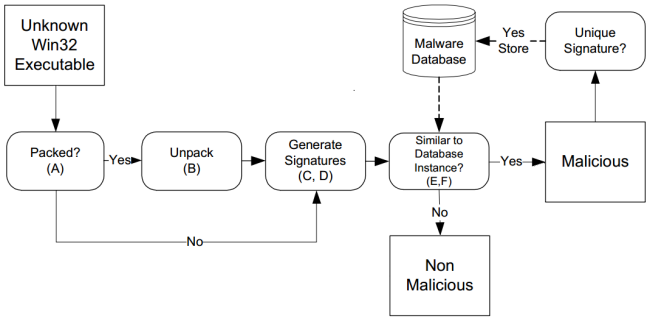

그림 2. CFG 기반 시스템의 전체적인 흐름.

4. 고려 사항

전술한 바와 같이 그래프 기반 멀웨어 탐지 기법들은 프로그램들을 그래프로 모델링한 후 그래프 간의 유사도를 비교하여 멀웨어들을 탐지한다. 따라서 좀 더 정확하고 빠른 멀웨어 탐지를 위해서는 다음과 같은 사항을 고려해야 한다. 첫 째, 멀웨어의 특성에 맞는 그래프 유사 척도를 사용해야 한다. 예를 들어, 단순히 두 그래프의 공통된 노드를 비교하는 것이 아니라 그래프의 위상학적 구조(topology)를 고려하여 멀웨어의 함수나 콜 관계를 비교하는 것이 멀웨어 변종을 좀 더 효과적으로 찾아낼 수 있다. 둘째, 빠른 멀웨어 탐지가 가능해야 한다. 일반적으로 그래프 간의 유사도를 비교하는 것이 높은 계산 복잡도를 요구하기 때문에 인덱싱과 같은 방법을 고려하여 빠른 시간 안에 멀웨어를 찾는 기법을 고려해야 한다.

5. 결론

본 논문에서는 그래프 기반 멀웨어 탐지 기법들과 정확하고 빠른 멀웨어 탐지를 위한 고려사항에 대해 논의하였다. 본 논문의 저자들은 이러한 고려사항을 기반으로 멀웨어를 효과적으로 탐지할 수 있는 다양한 기술에 대한 연구를 수행할 예정이다.

감사의 글

본 연구는 지식경제부 및 정보통신산업진흥원의 IT융합 고급 인력과정 지원사업(NIPA-2012-H0401-12-1001)과 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 지원(No. 20110029181)을 받아 수행된 연구임.

참고문헌

[1] J. Kolter and M. Maloof, "Learning to Detect Malicious Executables in the Wild," In *Proc. of the ACM SIGKDD*, pp. 470-478, 2004.

[2] Y. Ye et al., "IMDS: Intelligent Malware Detection System,," In *Proc. of the ACM SIGKDD*, pp. 1043-1047, 2007.

[3] J. Lee et al., "Detecting Metamorphic Malwares using Code Graphs," In *Proc. of the ACM SAC*, pp. 1970-1977, 2010.

[4] S. Cesare and Y. Xiang, "A Fast Flowgraph Based Classification System for Packed and Polymorphic Malware on the Endhost," In *Proc. of the 24th IEEE AINA*, pp. 721-728, 2010.