진화알고리즘을 이용한 악성코드 자동생성 시스템 설계

권세훈*, 권재영*, 이승훈*, 이현우*, 이종락**, 원일용*
*서울 호서 전문학교 사이버 해킹 보안과
**영남이공대학교 사이버보안과
e-mail : weed700@gmail.com

Automatic malware generation system design using EA

Se-Hoon Kwon*, Jae-Yeong Kown*, Seung-Hun Lee*,Hyun-Woo Lee*,
Jong-Rak Lee**, Il-Yong Won*
*Seoul Hoseo College Cyber Hack Security
** Yeungnam University College Cyber Security

요 약

본 연구는 기존의 변종 악성코드와는 달리 진화알고리즘을 기반으로 한 악성코드 자동 생성 프레임워크에 대한 것이다. 우리가 제안하는 시스템은 소스가 알려지지 않는 바이너리 상태의 악성코드를 역공학적 기법을 이용하여 소스 상태로 복원하고 복원된 소스를 이용하여 다양한 변종 악성코드를 생성하는 것이다. 진화 연산을 적용하기 위해 평가함수의 설계가 중요한데, 우리는 행동 기반 분석 기반의 평가 함수를 포함하는 프레임워크를 제안하였다.

1. 서론

악성코드(Malware)란 컴퓨터 시스템에 악의적인 행위를 수행하는 소프트웨어를 의미하는데, 바이러스, 웜, 백도어, 트로이목마등을 포함하는 광의의 용어로 사용된다. 대표적인 악성코드 중 하나인 바이러스란 용어가 처음 등장한 1983 년 이래로 악성코드는 기능적인 면에서 많은 진화의 과정을 거치고 있다. 그 진화의 과정은 5 개의 세대로 정의될 수 있는데, 1 세대는 단순히 자신의 코드를 다른 소프트웨어에 복제함으로써 감염을 일으킨다. 2 세대는 이미 감염된 파일을 식별하는 기능(self-recognition)과 같은 추가기능이 있다. 3 세대 악성코드는 백신 소프트웨어에 의한 탐지를 피하고자 스텔스 기술을 사용한다. 4 세대 악성코드는 백신 소프트웨어에서 자신의 분석이 어렵게 하는 기술을 가지고 있다. 5 세대 악성코드는 프로그램 구조 자체를 변경하고 복제 알고리즘을 적용해 코드를 난독화한다 [1].

악성 코드를 자동으로 생성하는 문제에 대한 연구는 두 가지 관점에서 그 의의를 찾을 수 있는데, 첫째는 백신 개발자의 입장이다. 하나의 악성코드가 발견되면 이를 탐지하고 삭제하기 위한 백신을 개발한다. 또한 이 악성코드의 변종이 발견되면 자신이 만든 백신이 이에 대응할 수 있는지를 테스트한 후 완벽하게 대응하지 못한다면 백신의 기능을 보완하는 추가적인 개발이 이루어 진다. 하나의 악성코드에 대하여 매우 다양한 변종이 발생된다는 관점에서 볼 때 이는 매우 소모적인 일이며, 예방적 차원에서 볼 때 늦은 대처일 수 밖에 없다. 만약 한 개의 악성코드가 발견되었을 때, 다양한 변종에 대해 미리 백신을 테스트할 수 있다면 보다 완벽한 백신을 개발할 수 있을 것이다.

또 다른 관점은 사이버전쟁 등에서 사용하는 사이버무기라는 관점이다. 사이버 전쟁에서는 상대방이 사용하는 백신에 탐지되지 않는 사이버무기를 빠르게 만들어 내는 것이 관건인데 이를 위해서는 많은 시간과 높은 전문성이 요구된다. 따라서 백신에 탐지되지 않는 공격용 악성 코드 생성을 자동화하여 대량화 할 수 있는 가는 사이버 국방 무기의 관점에서 매우 중요하다고 할 수 있다.

과거의 진화 악성코드 예로는 다음과 같다. 1991 년 처음으로 다양한 감염을 일으키는 바이러스 Tequila 와 Maltese Amoeba 등이 있고, 2002 년 출시된 바이러스 Win32/Simile(Etap 및 MetaPHOR)는 windows 용 어셈블리언어로 작성된 변종 컴퓨터 바이러스이다. 제작자인 “Mental Driller”에 의해 작성됐으며 Tuareg 다형성 엔진을 사용하였다. 하지만 이는 간단한 코드 수정이나 위치의 변화, 혹은 코드의 압축기술 등을 이용한 방법으로써 코드의 구조적 변화를 사용한 것일 뿐 진정한 진화라고는 말 할 수 없다[2].

진화알고리즘은 다윈의 진화론에 기반을 두어 만들어졌다. 이 알고리즘을 이용한 악성코드 자동 생성에 대한 연구는 이미 해외에서 오래전부터 진행되고 있다. 이러한 접근 방법의 가장 큰 어려움은 변종을 생성하기 위한 최적의 알고리즘을 선정하는 것과 그 알고리즘에 의해 생성된 후보 개체들의 적합성을 평가하는 적절한 방법을 찾는 것이다. 기존 연구들은 전통적인 정적 평가 방법을 사용하였는데 이러한 접근은 새로운 악성코드 생성이라는 본질적 문제를 개선하기에는 제한이 많다[3].

본 연구는 악성코드의 변종을 자동으로 생성하기 위하여 진화알고리즘(Evolutionary Algorithm)을 이용하기 위한 프레임워크를 설계하는 것이다. 특히 평가함수를 정적인 방법이 아닌 행위기반의 동적인 방법을 사용함으로써 기존의 유사 연구와 차별화를 시도하였다. 본 논문의 구성은 다음과 같다. 2 장에서는 관련 연구를 언급하였고, 3 장에서는 악성코드 자동생성 프레임워크를 제안하였다. 4 장은 결론 및 향후 과제이다.

2. 관련 연구

2.1 악성코드 수집 툴

악성코드를 수집하는 방법으로는 대표적으로 3 가지가 있다. 첫 번째는 웹사이트로부터 수집하는 방법인데, 이를 위해서는 웹크롤러라는 도구를 사용한다. 웹 크롤러란 웹 사이트를 탐색하여 각종 정보 및 문서를 수집하고 데이터베이스에 저장하는 도구이다. 이를 통해 악성코드를 유포하는 웹 사이트를 판단하고 다운로드하여 수집할 수 있다. 두 번째는 이메일 수신 시스템으로부터 수집하는 것이다. 일반적으로 공격자는 악성코드를 전파하기 위하여 불특정 다수를 대상으로 악성코드를 첨부하여 스팸 메일을 발송한다. 이를 통해 스팸 메일을 차단하고 첨부된 악성코드를 수집한다. 세 번째로 허니팟은 실제 악성코드에 감염될 환경을 구축하고, 악성코드가 실행되는 동안 악성코드의 행위나 특징을 모니터링이 가능하다. 허니팟은 공격자에게 허용하는 활동의 정도에 따라 High/Low-interaction Honeypot 으로 분류할 수 있다 [4].

2.2 진화 알고리즘 (Evolutionary Algorithm)

EA 는 다윈의 진화 과정을 모방하여 복잡한 문제를 푸는 알고리즘이다.

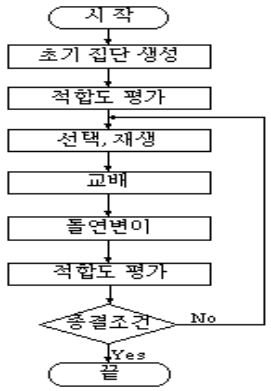
EA 에서는 많은 인공 객체의 문제의 해결방법을 생각하고 지속적으로 서로 경쟁하여 최적의 해결방법을 찾아낸다. 시간이 지남에 따라 이러한 인공 객체 중 가장 성공적으로 진화한 최적의 솔루션을 발견하게 된다.

EA 는 임의의 개체를 포함하는 크기 μ의 초기 모집단으로 시작한다. 모든 개인의 적합성 값이 지정된다. 개인의 적합성 값은 문제 해결 방법에 대한 값이다. 이후 해의 값은 적합도 함수를 사용하여 계산된다. 따라서 높은 적합성 값을 가진 집단은 낮은 적합성 값을 가진 개인보다 더 나은 해결책을 제시한다.

돌연변이(mutation), 재조합(recombination) 연산자를 사용하면 초기의 집단 μ 의 개인이 λ명의 자식을 생성한다. 생성된 자식 λ는 적합성 값이 지정되고 현재 집단의 적합성 값과 비교하여 새로운 집단을 생성하게 되고 위의 과정을 반복하여 높은 적합성 값을 가진 개인이 선택되고 어느 시점에 도달하면 진화를 멈추게 된다. 이러한 알고리즘을 EA 라 한다[5].

2.2.1 유전 알고리즘 (Genetic Algorithm)

자연 세계의 진화과정에 기초한 계산 모델로서 존 홀랜드(John Holland)에 의해서 1975 년에 개발된 전역 최적화 기법으로, 최적화 문제를 해결하는 기법의 하나이다. 생물의 진화를 모방한 진화 연산의 대표적인 기법으로, 실제 진화의 과정에서 많은 부분을 차용하였으며 선택, 교차, 변이 연산 등이 존재한다[3].



(그림 1) 유전 알고리즘 흐름도

2.2.2 유전 프로그래밍 (Genetic Programming)

사용자가 원하는 작업을 수행하는 컴퓨터 프로그램을 찾아내는 방법이다. 생물학적 진화를 통해 착안한 알고리즘으로, 유전 알고리즘의 확장된 형태이고 기본적인 특성은 유전 알고리즘 방식과 흡사하다. 하지만 간단한 명령어 집합을 이용하는 특성 때문에 개개의 결과물은 작은 하나의 컴퓨터 프로그램이 되어 기존 방식보다 복잡한 계산도 수행 가능한 것이 큰 장점이다[2].

2.3 다형성 바이러스

다형성 바이러스는 하나의 바이러스 본체가 암호화되어 수백만 가지 이상의 다양한 형태로 자신을 변형시킬 수 있는 바이러스이다. 또한, 다형성 바이러스는 암호화 기법 외에 쓰레기

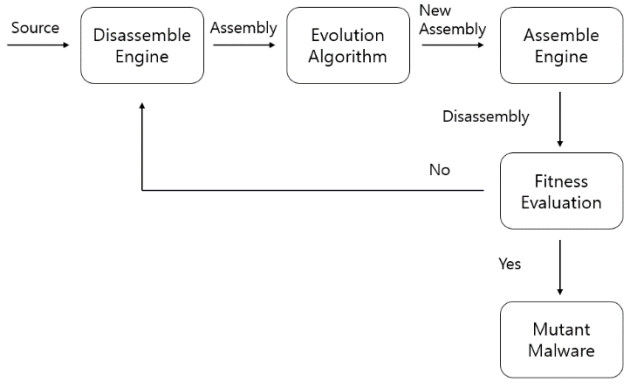
명령어의 삽입을 통해 실제로 자신이 사용하는 유효한 코드를 식별하기 어렵게 한다. 현재 악성코드 생성 툴은 대부분 다형성 바이러스를 생성한다. 하지만 이렇게 생성된 바이러스는 내용이 변했더라도 시그니처 기반 탐지와 메모리 블록 기법을 이용하면 쉽게 탐지가 된다[6].

3. 시스템 설계

3.1 프레임워크

본 논문에서 제안하는 악성코드 자동 생성 프레임워크는 (그림 2)와 같다. 바이너리 형식의 악성코드를 어셈블리언어 형식으로 변경한다. 이렇게 만들어진 어셈블리 소스를 진화알고리즘에 적합하도록 변형함으로써 다양한 악성코드 자손을 만들게 된다.

세대별로 만들어진 소스 레벨의 악성코드는 다시 어셈블 되어 바이너리로 변경된다. 다음 세대를 위해 적합도를 계산해서 그 값이 가장 큰 개체를 선택하여 다시 연산을 적용하는 과정을 반복하게 된다. 이 시스템은 제한된 시간이나 기준 적합도를 만족하게 하면 시스템이 종료되게 된다.



(그림 2) 시스템 전체 구성도

3.2 행동기반 동적 평가함수

만들어진 후보 악성코드의 적합도를 계산하는 것은 일반적인 진화 연산 적용의 단순한 정적 평가방법과는 다른 면이 있는데, 우리가 제안하는 동적 평가 방법은 (그림 3)과 같다.

해당 변종 악성코드를 3 가지의 기준으로 평가하는데 첫째는 진화알고리즘을 통해 만들어진 실행 파일이 정상적으로 실행되는지의 여부이다. 둘째는 변종 악성코들 샌드박스에서 실행하여 해당 변종 악성코드의 행위를 분석한 후 정의한 악성 행위의 정도에 따라 점수를 부여한다. 셋째는 만들어진 악성코드를 다양한 백신들로 테스트하여 탐지되는 비율에 따라 점수를 매기게 하는 방식이다. 이러한 평가 요소를 모두 종합적으로 고려하여 해당 악성코드의 적합도를 계산하게 된다.

본 논문에서 제한하는 적합도 평가 함수는 다음과 같다.

$$f(x) = \alpha A + \beta M + \gamma D$$

$$\alpha + \beta + \gamma = 1$$

$$0 < \alpha, \beta, \gamma \leq 1$$

(그림 3) 평가 함수

식에서 A 는 만들어진 후보 악성코드의 실행 여부를 의미하며 실행되지 않거나 무한 루프에 빠지게 되면 그 값이 0 이 되게 된다. M 은 후보 악성코드의 악성 행위를 판단한 값으로 악성코드의 성격에 따라 다양한데 예를 들어 다른 파일에 자신을 감염시키는 여부, 특정 레지스트리에 사인을 남기는 행위 등이 평가 대상이다. D 는 여러 백신이 해당 악성코드를 탐지할 수 있는 여부를 나타내는 것으로, n 개의 백신으로 탐지하여 탐지된 개수에 비례하여 그 값을 준다. α, β, γ 는 각 속성을 전체 평가에서 반영하는 정도를 나타내는 상수 값이다. (단, A=0 일때 $\beta = 0, \gamma = 0$)

4. 결론 및 향후 과제

본 논문에서 우리는 기존 변종 악성코드의 특징인 간단한 코드 수정과 위치 변화 또는, 코드 압축 등의 방법과는 다른 진화 알고리즘을 이용하는 자동 악성코드 생성 프레임워크 제안을 하였다.

이러한 진화 알고리즘을 효과적으로 적용하기 위해서는 생성된 후보 개체들의 평가함수를 정의하는 것이 핵심적 관건인데, 우리는 진화 알고리즘의 응용에서 흔히 사용하는 단순 정적 함수가 아닌 샌드박스를 이용한 동적 평가방법을 제안하였다.

향후 과제는 제안된 프레임워크를 실제 구축해서 다양한 진화 연산 알고리즘을 적용해 보는 것이 필요하며, 평가를 위한 샌드박스 구현 및 자동화 과정도 필요하다. 또 단일한 악성코드 샘플뿐만 아니라, 다양한 악성코드에서 유전자 풀을 만들어 의도하지 않은 새로운 변종 악성코드를 생성하는 연구도 필요하다.

참고문헌

[1] Sadia Noreen, Shafaq Murtaza, M.Zubair, Muddassar Farooq, “Evolvable Malware”, 2009

[2] A. Cani, M.Gaudesi, E.Sanchez, G.Squillero, A.Tonda, “Towards Automated Malware Creation: Code Generation and Code Integration”, 2014.1

[3] Dimitris Iliopulos1, Peter Szor2, and Christoph Adami11, “Darwin Inside the

Machines:Malware Evolution and the Consequences for Computer Security”, 2008
[4] 한경수, 김인경, 임을규, “악성코드 수집 및 분석 기술 평가지표 제안”, 보안공학연구논문지 2011.2
[5] Gareth Jones,“Genetic and Evolutionary Algorithms”,University of Sheffield, UK
[6] 최동균, 양하영, “다형성 바이러스의 소개”, 정보보호학회지 제 18 권 제 3 호, 2008.6