바이러스 탐지를 위한 휴리스틱 스캐닝 기법 및 행위 제한 기법 분석

°김은영, 오형근, 배병철
국가보안기술연구소
{eykim, hgoh, bcbae}@etri.re.kr

The Study of Heuristic Scanning Technique and Behavior Blocking Technique to Detect Virus

Eun-Young Kim, Hyung Geun Oh, Byungchul Bae
National Security Research Institute

요 약

급속도로 증가하고 있는 개인 단말 사용자와 네트워크의 광역화로 악성코드의 일종인 바이러스의 출현으로 그 해당 피해가 급증하고 있다. 이러한 현실에서 이전의 시그너쳐 기반 스캐닝 기법은 알려지지 않은 바이러스 및 신종 바이러스를 신속히 탐지할 수없으므로 탐지 성능이 급감하고 있다. 따라서 이전의 시그너쳐 기반의 스캐닝 기법의 단점을 보완하면서 새로운 기법의 바이러스를 탐지하기 위한 기법으로 제안된 휴리스틱 스캐닝 기법 및 행위 제한 기법에 대해 기술하겠다.

제 1 장 서론

컴퓨터가 보급될 초창기 시절에는 바이러스가 전파되어도 네트워크의 구성이 광역화되지 않아 그 파급속도는 상당히 느렸다. 따라서 바이러스는 초창기 시그너쳐 기반의 스캐닝 기법으로도 충분한 효과를 얻을 수 있었다. 즉, 바이러스의 전파속도가 느렸기 때문에 신종 바이러스 발견 후 신속한 시그너쳐 업데이트통해 전체 사용자에게 큰 피해를 입히지 않고 바이러스를 탐지할 수 있었다. 이로써 초창기 바이러스 탐지기법으로 시그너쳐 기반의 스캐닝 기법으로 상당한 효과를 거둘 수 있었다. 그러나 과거와는 달리 현실이 많이 달라졌다. 매일 새롭게 발견되는 바이러스는 웜이나 트로이 목마와 같은 특징을 갖춘 통합적인 악성 행위를 하는 바이러스로 진보하였고, 지능을 갖춘 바이러스의 등장으로 자기 스스로의 은닉 및 안티-바이러스 소프트웨어를 공격하는 활동도 할 수 있게 되었다. 또한 네트워크의 구성도 광역화되고 인터넷이 보편화되면서 신종 바이러스의 등장은 엄청난 파급 속도를 보이고 있다.

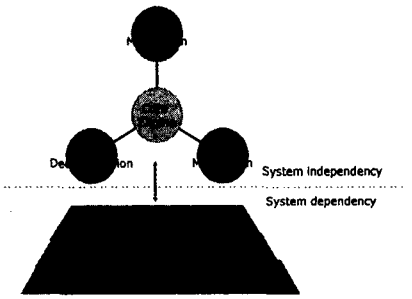
이와 같은 현실에서 시그너쳐 기반의 스캐닝 기법으로는 신종 바이러스를 탐지하는 것은 역부족이므로 현실에 맞는 안티-바이러스의 엔진을 설계해야 할 것이다. 따라서 본 논문에서는 급속이 변화하는 현실에 맞춰 산종 바이러스를 탐지하기 위해 제안되고 있는 새로운 바이러스 탐지 엔진의 구조 및 탐지 기법 등에 대해 분석 기술하겠다.

제 2 장 스캐닝 기법의 바이러스 탐지 엔진

본 장에서는 시그너쳐 기반의 스캐닝 기법을 이용한 안티-바이러스 탐지 엔진을 설계시 구성 요소 및 고려사항에 대해 기술하겠다[1].

1. 스캐닝 기법의 탐지 엔진 구성

안티-바이러스 탐지 엔진은 시스템과 의존적이거나 독립적인 요소로 구분할 수 있다. 안티-바이러스 탐지 엔진 구성 요소는 다음과 같다.



(그림 1) 안티-바이러스 엔진 구성 요소

시스템에 독립적인 요소는 핵심 엔진(Core Engine), 업데이트 매커니즘, 압축 파일 풀기, 바이러스 삭제 매커니즘이다. 안티-바이러스 엔진의 핵심 엔진 (Core Engine)은 파일 시스템과 분리된 순수 탐지 엔진이며, 새로운 모듈이 업데이트될 경우 핵심 엔진과 연계되어 운영하여야 한다. 업데이트 기능은 실시간으로 사용자가 안티-바이러스의 데이터 베이스를 업데이트를 할 수 있도록 고려되어야 한다. 그리고 다음은 삭제 기법이다. 바이러스는 각각의 특징이 다양하므로 바이러스에 따라 삭제하는 기법을 체계적으로 분류되어야 한다. 또한 운영체제에 따라서도 삭제 방법이 달라질 수 있도록 고려해야 한다. 압축 파일 풀기 매커니즘은 다음과 같다. 실제 안티-바이러스 엔진에서 압축 풀기의 기능이 차지하는 비중은 그다지 크지는 않지만, 구현시에는 상당히 복잡한 기능이다. 즉, 압축 파일 형식인 .zip, .tar 등의 압축 포맷은 외부에서 압축을 풀어주는 프로그램을 사용하지 않고 압축을 풀수 있는 기능을 갖추어야 한다. 또한 압축된 파일이 암호화된 파일이라면 이 암호를 해독할 수 있는 기능 또한 갖추어져야 한다. 따라서 압축된 파일 및 암호화된 파일을 해독한다는 것은 안티-바이러스 엔진에서 상당한 복잡도와 문제점을 가지고 있다.

운영 시스템과 밀접한 관련이 있는 파일 시스템, 파일 형식 스캐너, 메모리 스캐너의 구성 요소 특징은 다음과

같다. 파일 시스템 계층은 시스템 파일과 핵심 엔진을 분리하여 서로 다른 플랫폼에서 같은 API 를 호출할 수 있도록 정의하여 사용할 수 있도록 하는 것이다. 파일 형식 스캐너는 안티-바이러스에서 선행 되어야할 작업으로 해당 파일의 타입을 알아내는 것이다. 만약 해당 파일의 타입을 알게되면 탐지 엔진에서는 해당 파일이 압축된 파일인지 혹은 암호화된 파일인지 판단할 수 있으며, 압축된 파일인 경우 탐지 엔진에서 압축을 풀수 있는 루틴을 수행시켜 해당 파일을 검사를 할 수 있다. 메모리 스캐닝 기능은 해당 시스템에서 사용자가 스캔하기를 원하는 파일을 스캔할 수 있는 기능이다. 이러한 기능은 사용자에게 보다 많은 편의성을 제공해줄 수 있으므로 고려해야 할 것이다.

2. 스캐닝 기법의 탐지 엔진 설계시 고려사항

안티-바이러스 핵심 엔진은 다양한 운영체제하에서 운영될 수 있는 소스 코드로 제작되어야 하므로, 안티-바이러스 엔진을 설계할때에는 반드시 다음과 같은 사항을 고려해야한다.

첫째, 플랫폼 또는 프로그램 언어 선정이다. 서로 다른 주소 체계를 가진 실행 코드는 서로 다른 주소체계의 운영체제에서 실행될 수 없다. 따라서 안티-바이러스의 탐지 엔진을 설계할 때에는 어떤 플랫폼에서 개발 및 운영시킬 것인지에 따라 적절한 프로그램 언어를 선택하여 개발해야 할 것이다. 일반적으로 안티-바이러스 핵심 엔진은 모든 플랫폼에서 사용할 수 있는 프로그램 언어로 개발되어야고, 모든 플랫폼에서 구동가능한 컴파일러가 유용하며 탐지 엔진은 C 또는 C++의 프로그램 언어를 사용하여 개발한다. 그리고 각각의 프로그램 언어의 데이터 타입이 서로 다르므로 모든 플랫폼이 동일하게 적용되어 사용할 수 있는 표준 데이터 타입을 정의하여 사용하는 것이 효과적이다. 둘째, 파일 시스템과 안티-바이러스 엔진의 핵심 부분은 분리되어야한다. 이는 파일 시스템과 핵심이 되는 안티-바이러스 엔진과 분리하는 것이다. 이렇게 분리된 안티-바이러스 핵심 엔진은 다양한 플랫폼에 따라 컴파일이 가능하도록 구현한다. 안티-바이러스 엔진 부분에서 파일 시스템과 연결되는 부분을 엔진과 분리하여 개발하면 추후 프로그램의 확장성 및 유지관리가 쉬워진다. 그리고 실제 안티-바이러스 엔진에서 파일 시스템과 연결되어 분리된 계층은 메모리 실험기와 사용자 인터페이스 부분이다. 셋째, 모듈화는 소프트웨어 개발에 상당히 중요한 부분이다. 분명한 것은 사용자 인터페이스 부분과 핵심 모듈별 구분을 확실해둔다면 상당한 이점을 가지게 되며, 안티-바이러스 엔진을 탐지 대상별로 분류 및 개발한다면, 추후 알려지지 않은 새로운 바이러스가 발견되었을 경우 탐지 엔진의 확장이 가능하다.

제 3 장 휴리스틱 스캐닝 기법

이 장에서는 안티-바이러스 엔진의 새로운 탐지 기법인 휴리스틱 스캐닝 기법(Heuristic Scanning) 에 대해 기술하겠다[2].

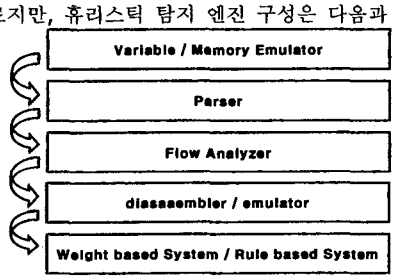
1. 휴리스틱 스캐닝 기법 개요

휴리스틱 스캐닝 기법은 시그너쳐 기반의 스캐닝 기법과 유사하다. 그러나 휴리스틱 스캐닝 기법은 시그너쳐 기반의 스캐닝 기법과 같이 어떤 특정 시그너쳐를 찾는 대신에 보통의 응용 프로그램에서 발견할 수 없는 어떤 특별한 명령이나 행위를 찾게 된다. 따라서 이러한 휴리스틱 기법을 이용하여 구현된 탐지 엔진은 바이러스, 웜 또는 트로이 목마 등 알려지지 않은 새로운 악성행위를 탐지할 수 있다.

휴리스틱 스캐닝 기법은 두가지로 구분할 수 있다. 가중치 기반 시스템(Weight-based systems)와 규칙 기반 시스템(Rule-based systems)로 구분할 수 있다. 가중치 기반 휴리스틱 엔진은 오래전부터 개발 및 발전된 기법으로 바이러스 행위에 따라 차등적으로 점수를 두어 위험 점수의 합이 특정 점수 이상이면 바이러스라 판단하는 기법이다. 따라서 이 기법은 아래에서 설명하는 규칙 기법 시스템보다는 구현이 쉽다. 휴리스틱 스캐닝 기법에서의 규칙 기반 시스템은 파일에서부터 어떠한 행위를 분석하고, 그 행위에 관한 규칙을 산출한다. 즉, 바이러스 행위에 해당하는 규칙이 생성되면 해당 규칙을 기반으로 바이러스를 탐지하게 된다. 이 기법은 위에서 설명한 가중치 기반 시스템보다 구현이 복잡하지만 바이러스 행위에 대한 규칙이 생성되어 규칙 기법으로 탐지가 되므로, 알려지지 않은 바이러스 행위 및 바이러스 등을 탐지하기에 용의하다.

2. 휴리스틱 탐지 엔진 구성

휴리스틱 탐지 엔진은 실제 운영 환경과 기술 수준에 따라 다르지만, 휴리스틱 탐지 엔진 구성은 다음과 같다.



(그림 2) 휴리스틱 탐지 엔진 구성 요소

위와 같이 구성된 휴리스틱 탐지 엔진은 다음과 같이 바이러스를 탐지한다. 만약 스크립트 기반의 바이러스 코드를 탐지할 경우, 주어진 입력 파일의 구문을 분석하여 잘못된 형식과 사용되지 않는 가변 변수등을 제거시킨다. 그리고 입력 파일의 구문 분석이 끝나면 해당 입력 파일의 파일 타입이 정해진다. 파일 타입이 정해지면 휴리스틱 엔진은 주어진 파일의 구조를 분석한다. 만약 해당 파일이 바이너리 파일인 경우, 바이너리 파일의 구조는 명확하여 스크립트 기반의 바이러스를 분석할 수 있을 것이다. 그 다음 단계로 휴리스틱 엔진의 메인 루프는 정보를 선택 및 추출을 해야하며, 핵심이 되는 명령문을 분석해야 한다. 이러한 구문 분석은 구문의 의미와 구문에 따른 플래그 등의 세팅 유무를 분석할 수 있다. 그리고 이러한 구문 분석은 사용 가능한 변수 실험기 또는 메모리 실험기를 통해 조정할 수 있을 것이다.

3. 함수의 위험도 선정 기준

휴리스틱 기법에서 프로그램의 분석을 마친 후에는, 발견된 함수를 위험도에 따라 등급을 구분할 수 있다. 이

작업은 가중치 기반 시스템 또는 규칙 기반 시스템에서 사용된다. 가중치 기반 시스템에서는 발견된 함수들의 일반적인 위험도와 특별 가중치 위험도로 구분할 수 있을 것이다. 또한 이러한 기법은 일반적인 형태에서는 발견되지 않는 것을 대상으로 선정해야한다. 하지만 이러한 함수 위험도 선정에는 많은 'False Positive' 가 있을 수 있다. 즉, 매크로 바이러스인 경우 가중치 기반 시스템에서는 일반적인 문서에서도 발견될 수 있는 "normal.dot" 이라는 문서에 높은 위험도를 적용하였을 경우 바이러스 행위가 어느 곳에서도 발견되지 않았더라도 위험도가 높은 결과로 나올 수 있다. 따라서 안티-바이러스 엔진은 특정 함수가 발견되었다고 위험도를 판단하기보다는, 위험도와 더불어 어떠한 특별 조건이 성립되었을 경우 높은 위험도를 적용할 수 있도록 설계되어야한다.

제 4 장 행위 제한 기법

이 장에서는 바이러스 탐지 엔진의 또 다른 기법으로 행위 제한 (Behavior Blocking) 기법에 대해 분석하겠다[3].

1. 행위 제한 기법 개요

일반적으로 시그너쳐 기반의 안티-바이러스 소프트웨어는 수천가지의 시그너쳐를 기반으로 파일, 디스크를 감시하여 바이러스를 탐지하게 된다. 이러한 시그너쳐 기반의 스캐닝 기법은 위에서 기술하였듯이 알려진 시그너쳐를 기반으로 탐지를 하기 때문에 새로운 바이러스를 탐지할 수 없다.

그러나 이와는 대조적으로 휴리스틱 스캐닝 기법은 프로그램의 종합적인 구조를 조사한다. 종합적 구조에는 컴퓨터 명령문과 파일안에 포함되어있는 데이터를 포함한다. 이러한 휴리스틱 스캐너는 논리적으로 명확하게 정의된 바이러스 행위를 탐지하게 된다. 따라서 이전의 시그너쳐 기반의 스캐닝 기법보다는 알려지지 않는 바이러스 행위를 탐지할 수 있다. 그러나 만약 복잡한 침입이 발생하였을 경우, 시그너쳐 기반의 스캐닝 기법과 휴리스틱 엔진은 "Sand-Boxing"을 통하여 비트-바이트 단위의 스캐닝해야 할 것이다. 그러나 이러한 가상 머신안에서 바이러스 코드를 실험한다는 것은 그 실험 자체에 상당한 제한을 가지고 있으며, 시스템의 위험도를 정확히 판단할 수 없을 것이다. 또한 시그너쳐 기반의 스캐닝 기법과 휴리스틱 기법은 사용자의 컴퓨터에 감염되기 전에 탐지될 가능성이 이전과는 현저히 떨어졌다. 따라서 이러한 단점을 보완하기 위해 행위 제한 기술이 나타나게 되었다.

2. 행위 제한 기법의 분류

행위 제한 시스템은 정책 기반 시스템과 전문가 기반 시스템으로 분류할 수 있다. 정책 기반 제한 시스템은 특정 행위를 모니터링하여 그 행위를 허용할 것인지 제한할 것인지를 결정하는 것이다. 만약 어떤 프로그램이 운영체제에 어떠한 행위를 요청하였을 경우, 정책 기반 시스템은 정책 데이터베이스에 비교하여 해당 행위를 허용할 것인지 제한할 것인지 결정하는 것이다. 예를 들어 자바 스크립트인 경우 정책 기반 시스템은 다음과 같은 정책을 정할 수 있다.

<표 1> 자바 스크립트 행위 정책 예제

Operation Description	Block Request?
Allows applet to open files	Yes
Allows applet to delete files	Yes
Allows applets initiate network connections	Yes
Allows applets to access files in the system	No

이와는 대조적으로 전문가 기반 시스템은 전문가가 전체 바이러스를 분석하여 그들이 시스템에 끼치는 바이러스 행위를 분석하고 의심이 가는 행위에 대해 직접 제한을 가할수 있게 구성한다. 또한 이러한 전문가 기반 시스템은 이미 알려진 바이러스 코드의 80%가 시작 프로그램에 해당하는 시스템 파일을 접근하기 전에 레지스트리를 먼저 접근하기에 이에 준하는 정책을 설정할 수 있다. 따라서 행동 제한 시스템에서는 시작 프로그램의 레지스트리 영역을 접근하여 수정을 하게되면 이 행위에 대해 제한을 하게 된다. 그러나 이러한 전문가 시스템 또한 여전히 'False Positive' 요소를 내제하고 있다.

제 5 장 결론

본 논문에서는 이전의 안티-바이러스 솔류션의 단점을 생각해보고 매일 새롭게 생겨나는 바이러스를 탐지 하기 위한 기법으로 휴리스틱 스캐닝 기법 및 행위 제한 기법에 대해 분석하였다. 휴리스틱 스캐닝 기법에서의 규칙 기반 시스템은 단순히 어떠한 규칙을 기반으로 비교하여 어떠한 기능을 발견하게 되는데, 만약 이미 규정하였던 규칙을 코드 안에서 발견하였다면 규칙 기반 시스템은 확실한 결과를 도출시켜준다. 그러나 휴리스틱 탐지 엔진은 'False Positive'를 야기할 수 있는 확률이 크므로 잘못된 결과를 도출할 수 있음을 간과해서는 안될 것이다. 또한 행위 제한 기술은 바이러스 행위에 해당하는 행위를 정책 및 전문가 기반 시스템으로 구성하여 바이러스 행위 자체에 대해 허용할 것인가 아니면 제한할 것인가를 정책적으로 규정하는 기술이다. 위에서 언급한 휴리스틱 스캐닝 기술과 마찬가지고 행위 제한 기술도 'False Positive'를 야기시킬 수 있는 문제점을 가지고 있지만, 휴리스틱 기술보다는 상대적으로 알려지지 않은 바이러스를 탐지하기에는 효율적인 기법으로 주목받고 있다.

앞으로는 다양하게 진보된 바이러스가 증가할 것이다. 따라서 이러한 새로운 바이러스를 탐지하기 위한 탐지 기법 및 엔진 개발이 시급하며 위에서 언급한 휴리스틱 기술 및 행위 제한 기술 등은 개발 및 발전시켜 바이러스코드 및 바이러스 등을 탐지해야 할 것이다.

참고 문헌

1. M. Schmall. "Building an Anti-Virus engine ", http://online.securityfocus.com/infocus/1552. March, 2002.
2. M. Schmall. "Heuristic Techniques in AV Solutions: An Overview ", http://online.securityfocus.com/infocus/1542, Feb, 2002.
3. C. Nachenberg. "Behavior Blocking : The Next Step in Anti-Virus Protection", http://online.securityfocus.com/infocus/1557, March, 2002.