Journal of The Korea Society of Computer and Information
Vol. 19, No. 7, July 2014
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2014.19.7.077

다중 필터를 이용한 실시간 악성코드 탐지 기법

박 재 경*

A Realtime Malware Detection Technique Using Multiple Filter

Jae-Kyung Park*

요 약

최근의 클라우드 환경, 빅데이터 환경 등 다양한 환경에서 악성코드나 의심 코드에 의한 피해가 늘어나고 있으며 이를 종합적으로 대응할 수 있는 시스템에 대한 연구가 활발히 이루어지고 있다. 이러한 악성행위가 내포된 의심코드는 사용자의 동의 없이도 PC에 설치되어 사용자가 인지하지 못하는 피해를 양산하고 있다. 또한 다양한 시스템으로부터 수집되는 방대한 양의 데이터를 실시간으로 처리하고 가공하는 기술뿐만 아니라 정교하게 발전하고 있는 악성코드를 탐지·분석하기 위한 대응기술 또한 고도화 되어야 한다. 최근의 악성코드를 원천적으로 탐지하기 위해서는 실행파일에 포함된 악성코드에 대한 정적, 동적 분석을 포함한 분석뿐만 아니라 평판에 의한 검증도 병행되어야 한다. 또한 대량의 데이터를 통해 유사성도 판단하여 실시간으로 대응하는 방안이 절실히 필요하다. 본 논문에서는 이러한 탐지 및 검증 기법을 다중으로 설계하고 이를 실시간으로 처리할 수 있는 방안을 제시하여 의심코드에 대한 대응을 근본적으로 할 수 있도록 연구하였다.

▸ Keywords : 악성코드, 의심코드, 탐지, 검증, 다중 필터

Abstract

Recently, several environment damage caused by malicious or suspicious code is increasing. We study comprehensive response system actively for malware detection. Suspicious code is installed on your PC without your consent, users are unaware of the damage. Also, there are need to technology for realtime processing of Big Data. We must develope advanced technology for malware detection. We must analyze the static, dynamic of executable file for fundamentally malware detection in recently and verified by a reputation for verification. It is need to judgment of similarity for realtime response with big data. In this paper, we proposed realtime detection and verification technology using multiple filter. Our malware study suggests a new direction of realtime malware detection.

▸ Keywords : Malware, Suspicious Code, Detection, Verify, Multiple Filter

•제1저자 : 박재경, •교신저자 : 박재경
•투고일 : 2014. 7. 1. 심사일 : 2014. 7. 13, 게재확정일 : 2014. 7. 26.
* 한국과학기술원 사이버보안연구센터(Cyber Security Research Center in KAIST)

I. 서 론

2013년 이후부터 보다 공격적이고 심각한 악성코드에 대한 피해는 날로 증가하는 추세이다. 국내에서는 개인정보 유출 등의 문제와 연계되어 인터넷에 대한 불신이 가중되고 있는 실정이다. 이는 국내뿐만 아니라 미국 등의 선진국에서도 일어나는 현상으로 2013년 12월 미국의 유명 대형 할인점인 타겟(Target)사의 고객 개인정보와 신용카드 정보가 해킹으로 유출되는 사건이 발생했다. 타겟사는 월마트에 이어 미국 내2위의 소매업체로 미국과 캐나다에서 1,900여개 매장을 운영하고 있다. 당초 피해자는 4천만 명 수준으로 알려졌으나 이 업체의 공식 성명서에 따르면 발생한 오프라인 방문 고객들의 계좌정보 유출사건으로 최대 7천만 명의 피해자가 발생했다[1]. 이러한 문제점을 해결하기 위해서는 보다 근본적인 대책마련이 시급하다. 본 논문에서는 단순한 형태의 탐지가 아닌 다중 검사 방식을 도입하여 실시간으로 의심코드를 탐지하는 기법을 주장하고자 한다. 또한 실제 악성행위를 하는 악성코드인지에 대한 검증을 통해 확진을 하여 해당 시그니처를 긴급히 배포하는 방식으로 설계하였다.

최근에 악성코드를 이용한 대표적인 공격 유형이 APT (Advanced Persistent Threat) 공격이며, 이 공격은 목표를 정하여 공격하는 방식으로 노골적인 공격을 감행하고 있다. 또한, 사이버전과 같은 국가 간의 전쟁의 양상으로도 확대되고 있는 추세이다. 이에 대한 기술적인 대응책으로 악성코드를 차단하기 위한 기술 개발이 이루어지고 있으나 단순히 악성코드에 대한 패턴을 이용한 시스템이거나 악성코드의 배포지에 대한 URL을 차단하는 수준에 머물고 있다. 2013년 7월에 발표된 악성코드 유형별 비율을 보면 악성코드 유형 중 원격제어가 59%의 비율로 가장 높았으며, 그 이외에도 다운로더 14%, 드롭퍼 8%, 감염PC 정보탈취 8%, 온라인 게임 계정 탈취 4% 등의 악성코드 유형이 다양하게 나타났다[2].

이처럼 다양하고 파악하기 어려운 의심코드를 악성코드로 검증하고 이를 차단하기 위한 방안을 위해 본 논문에서는 Drive-by-Download를 검증하고 또한 실제 코드를 실제 PC환경에 실행시켜 검증을 하고자 한다. 또한 이러한 방식은 실시간으로 이루어지며 자동화하여 결과를 생산하는데 큰 의미가 있다고 할 수 있다. 이러한 기법을 통하여 차단시스템과 연동하여 방어할 경우 보다 효과적인 방어수단이 될 수 있다고 할 수 있다.

본 논문에서 의미하는 의심코드는 악성코드 뿐만아니라 사용자가 판단하여 피해를 줄 수 있는 코드는 모두 포함하고자 한다. 즉, 내부 자료 유출의 경우 악성코드는 아니나 사용자가 유출되는 것을 꺼려하는 것으로 볼 수 있으며 또한 파밍(Phaming)과 같이 인터넷 주소를 변조함으로써 사용자들로 하여금 진짜 사이트로 오인하여 접속하도록 유도한 뒤에 개인정보를 훔치는 범죄도 예방하는 방안을 강구하고자 한다. 내부망의 정보유출을 막기 위해서는 아웃바운드로 나가는 통신에 대한 검사를 통해 의심코드가 포함되어 있는지를 판단할 수 있는 필터를 설계하도록 한다. 본 논문의 연구를 바탕으로 설계된 프로토타입 시스템을 통해 실제 환경에서의 활용 가치와 성능적인 문제를 해결할 수 있는 방안을 함께 제안한다. 기존에 구성된 네트워크 환경에 패킷을 복사하는 형태로 구성하여 실제 트래픽을 실시간으로 처리할 수 있는 실험을 진행하였다. 본 논문의 구성은 다음과 같다. 2장에서는 관련연구에 대해서 살펴본다. 3장에서는 다중 필터에 의한 의심코드를 실시간으로 탐지하는 기법에 대해서 기술하고, 4장에서는 본 논문이 제안하는 기법을 전체적으로 설계한다. 그리고 마지막으로 5장에서는 결론 및 향후 연구방향에 대해서 기술한다.

II. 관련연구

1. 악성코드 유포사례

1.1 국내 기관 대상 대규모 보안사고 유발

2013년 3.20 사고와 6.25 공격 사례를 살펴보면, 두 피해사례는 정부기관, 언론사, 금융기관들과 같이 사회 기반이 되는 산업들이 목표가 되었다는 점에서 공통점이 있다. 하지만 공격 기법 면에서는 서로 다른 형태를 보이고 있다. 3.20은 2011년과 그 이전에 발생했던 대규모 보안사고 사례와 유사하게 감염된 시스템의 정상적인 가동 방해 및 데이터 파괴를 위해 디스크의 MBR과 VBR을 특정 문자열로 강제 덮어쓰기를 수행하는 악성코드 유포가 목적이었다[2]. 6.25 공격 사례는 시스템의 정상적인 가동과 데이터 파괴 목적의 악성코드 유포, 국내 정치단체와 정부기관 웹사이트에 대한 대규모 DDoS 공격을 병행했으며, DNS 증폭 디도스 공격 및 스크립트 기반 공격 등 새로운 DDoS 공격 기법이 사용되었음을 알 수 있다.

1.2 메모리 패치 이용 인터넷 뱅킹 악성코드 출현

2013년 6월 발견된 온라인 게임 사용자 정보 유출 형태의 악성코드는 기존에 발견되지 않았던 국내 인터넷 뱅킹 사이트

에 대한 정보 유출 기능이 추가됐다. 이러한 온라인 게임의 악성코드가 게임 사이트 외에 다른 웹사이트를 목표로 삼은 시기는 2012년 8월부터이며, 당시에는 언론사 및 정부기관의 서버 관리자 페이지에 대한 정보 유출이 이루어졌다. 기존의 뱅키(Banki) 악성코드가 사용했던 호스트(hosts) 파일을 변조해 허위로 제작된 금융권 웹 사이트로 사용자 접속을 유도하는 방식이었다[3]. 2013년 처음 발견된 인터넷 뱅킹 사이트 공격 방식은 개별 금융권 웹사이트에서 설치되는 고유한 보안 모듈들에 대한 메모리 패치를 수행해 정보 유출이 발생하도록 설계되었다. 이러한 공격 기법은 정상적인 인터넷 뱅킹 이용 과정 중 정보 유출이 발생하므로 일반적인 사용자가 인지하는 것은 거의 불가능하다.

1.3 국내 소프트웨어 대상 제로데이 취약점 증가

최근 국내 소프트웨어를 대상으로 한 취약점과 이를 악용한 악성코드가 증가하기 시작했고, 이러한 추세는 가속화하고 있다. 예를 들어 인터넷 뱅킹에 많이 쓰이는 소프트웨어의 취약점이 발견됐고(특히 Active-X), 이 외에도 곰플레이어와 같은 동영상 플레이어 프로그램에서 취약점이 보고되어 업데이트가 권고되기도 했다. 국내 문서작성 소프트웨어의 취약점을 이용한 경우도 발생하였는데 전 세계적으로 많이 사용하고 있는 문서관련 프로그램(워드 및 PDF)의 취약점을 이용한 공격도 꾸준히 발생하고 있으며 전자우편을 이용하여 자극적인 문구로 공격을 감행한다.

1.4 파밍과 결합된 온라인 게임 계정정보 탈취 악성코드 등장

2013년 상반기에 발견된 보안 위협들의 특징 중 하나는 개인정보를 탈취하기 위한 공격이 활발했다는 점이다. 상반기 국내에서 가장 많이 발견된 개인정보 탈취 형태의 악성코드는 인터넷 뱅킹 정보를 탈취하는 형태와 온라인 게임 계정정보를 탈취하는 악성코드라고 할 수 있으며 공인인증서의 탈취도 많이 이루어진 것으로 보고되고 있다. 특히 온라인 게임 계정을 탈취하는 악성코드는 윈도우 시스템 파일을 변경하거나 패치하는 형태로 유포됐는데, 이는 사용자에게 발각되지 않도록 위장하기 위한 기법이다. 그리고 보안 소프트웨어의 진단을 피하기 위해 셀 수 없이 많은 변형들을 유포하거나, 목적 달성을 위해 다양한 보안 소프트웨어를 무력화시키는 기법들이 날로 발전하고 있는 실정이다. 또한 인터넷 뱅킹 정보 탈취 형태의 악성코드는 정상적인 은행 사이트와 구분이 어려울 정도로 유사한 피싱(Phishing) 웹사이트를 이용한 방법에서부터 호스트(hosts) 파일 변조 형태, IP차단을 피하기 위해 주기적으로 서버(C&C)와 통신하는 등 점점 고도화 되고 있다[3].

1.5 드라이브 바이 다운로드 공격

드라이브 바이 다운로드 공격은 궁극적으로 사용자컴퓨터의 취약점을 이용하여 악성코드를 유포하는 방법이다. 공격자는 사용자 PC를 악성코드 유포사이트로 유도하기 위해 취약점을 가진 웹 서버에 SQL인젝션(SQL injection)과 같은 방법으로 침투한 후 웹 페이지에 직접 악성코드유포지로 연결되는 코드를 삽입하거나, 광고 배너나 제3의 위젯과 연결되어 있는 링크를 변조하여 공격을 실행한다[4]. 드라이브 바이다운로드 공격은 사용자의 컴퓨터가 해킹된 웹 사이트인 최초 접속지에 접속하면 공격자가 삽입해 놓은 유도코드 또는 변조한 링크에 의해 수단계의 경유지를 거쳐 자동으로 악성코드 유포사이트로 연결된다. 사용자 컴퓨터가 유포사이트로 접속되면, 사용자 컴퓨터의 웹 브라우저나 어도비 플래시 플러그인과 같은 어플리케이션 취약점이 존재하는 사용자 컴퓨터로 실행 파일이 다운로드 되고 컴퓨터는 악성코드에 감염된다[5].

2. 웹 브라우저 기반 악성행위 탐지 시스템(WMDS)

기존 시스템의 한계를 몇 가지로 요약해 보면 첫 번째로 시스템에서 생성되는 모든 파일, 레지시트리, 프로세스, 네트워크 등의 이벤트 중 웹 어플리케이션 취약점 공격에 의해 파생되는 프로세스 및 스레드의 이벤트만을 구분해 내기에는 기준점이 모호하다는 것이고, 두 번대는 일번적인 텍스트 형식과 같은 스크립트 언어를 사용하는 악성코드는 탐지가 불가능하다는 것이다. 이러한 한계를 개선하기 위해 웹 브라우저부터 시작해 웹 어필리케이션 취약점 공격에 의해 파생되는 모든 프로세스 및 스레드 주체의 이벤트를 탐지 해주는 모듈인 Observer를 적용한 새로운 웹 브라우저 기반 악성행위 탐지 시스템이 WMDS이다. 이 WMDS는 프로세스에 Observer를 함께 실행하여 파일, 레지스트리, 서비스 등의 모든 리소스들을 모니터링하고 이에 대한 로그를 기록하는 시스템이다[9].

III. 본론

이번 장에서는 본 논문에서 제안하고자 하는 실시간 다중 필터를 이용한 의심코드 및 검증 기법을 소개한다. 본 기법은

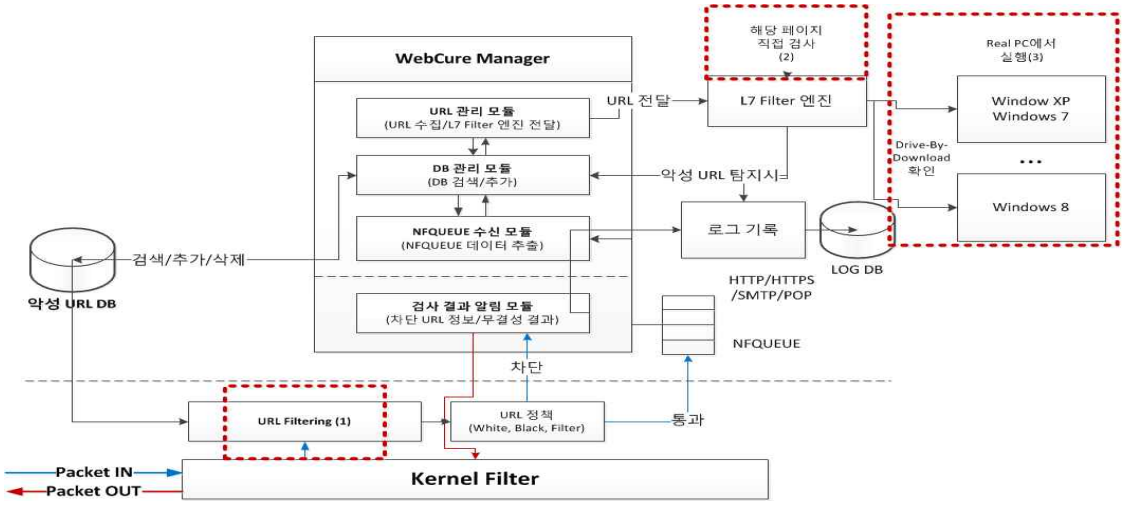

그림 1. 다중 필터 시스템 구성
Fig. 1. Multi-Filter System Architecture

어플라이언스 타입의 장비를 통해 실시간으로 패킷을 처리하면서 의심코드를 분류하고 해당 코드가 악성코드인지를 파악하는 다중 필터를 통해 탐지 및 검증하는 과정을 수행한다. 그림 1은 이러한 본 연구 시스템의 전체적인 구성을 나타내고 있다. 한 드라이브 바이 다운로드 기법을 원천적으로 방어하기 위해 실제 PC환경과 동일한 환경에서 실행을 함으로써 사용자의 개입 없이 실행 가능한 의심코드가 내려오는 지를 확인하는 일련의 과정을 진행한다.

3.1 다중 필터 시스템 구성

본 논문에서 제안한 필터는 총 3가지의 종류로 커널 레벨의 필터와 크롤링 기반의 필터 그리고, 동적 실행 필터로 구성되어 있다. 각각의 레벨에서의 필터링을 통해 성능 지연을 최소화하여 실시간으로 처리할 수 있도록 설계하였다. 다음절에서 각 레벨별 필터에 대한 구성을 설명하도록 한다.

3.2 커널 레벨 URL 필터링

기존의 커널에서의 URL 필터링은 단순히 패턴매칭에 의한 URL 필터링으로 해당 URL의 유무에 따라 추가 검사 여부를 결정하는 구조였으나 본 논문에서 제안하는 URL 필터링은 URL prefix 해시 트리를 이용한 방식으로 기존에 방식에 비해 검색 속도를 향상시켰다[6]. 또한 Web Patrol[7]은 웹 기반의 악성 코드에 감염되는 시나리오를 트리 형태의 그래프로 정의하고, 탐지 되는 시나리오를 수집, 캐싱 하여 추후 분석 과정에서 재현 할 수 있는 시스템을 제안 하였다[8]. 커널로부터 요청되어지는 HTML에서 URL을 추출하여 트리 형태의 테이블을 유지하여 해당 URL이 의심코드 추출을 위해 방문한 이력이 있는지 없는지를 판단하는 검사를 진행한다. 이 과정을 통해 향후 해당 URL을 크롤링하여 방문할 때 모든 페이지를 일일이 검사하는 불합리함을 개선하였다고 할 수 있다. 그림 2에서와 같이 특정한 URL에서 악성코드가 별견되었을 경우 하위 디렉터리 또한 감염되었다고 판단하는 것

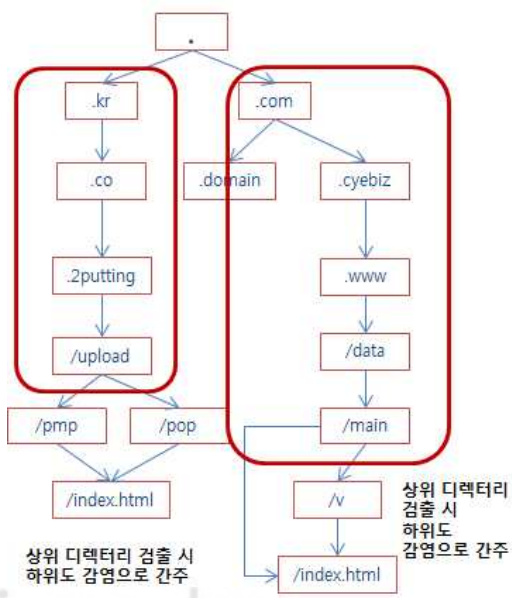

그림 2. 트리 구조에 의한 하위디렉터리 검출
Fig. 2. Below Directory Detection using Tree Structure

이 올바른 판단일 것이다.

즉, 그림 2에서처럼 /upload 디렉터리의 하위 페이지나 /main의 하위 페이지는 모두 동일한 형태로 감염이 되었으므로 동일한 검사를 반복할 필요는 없을 것이다. 따라서 도메인을 트리 형태로 유지하여 관리할 경우 불필요한 경로를 줄일 수 있는 방안이 된다. 이때 추가적으로 하위 디렉터리가 최종적으로 가리키는 곳이 기 검출된 악성코드 포함 페이지 인지를 확인하는 절차가 추가적으로 이루어져야 한다. 단순히 하위 디렉터리라는 것만으로는 모두 악성코드 연관 페이지라고 볼 수는 없기 때문이다. 또한 향후 해당 페이지에서 악성코드가 제거되었을 경우에는 해당 트리의 정보값을 초기화하여야 정상적으로 이용할 수 있는 과정도 필요하다.

3.3 L7 레벨 실시간 필터링

기존 악성코드 탐지 연구 방법에서는 해당 웹 페이지를 다운로드하고 그 페이지 내에 숨김 iframe과 같은 비정상 태그의 존재 여부를 확인하여 처리하였다[5]. 해당 방법은 SpiderMonkey[9]를 기반으로 스크립트 에뮬레이팅 모듈을 통해 페이지 내의 모든 URL을 추출하여 반복적으로 링크를 발견하지 못 할 때까지 반복하는 크롤링 기반의 탐지 방법이었다. 하지만, 이 방법의 경우는 실제 시스템을 적용하는데 있어서는 성능적인 한계가 있을 수 있다. 즉 하나의 페이지 요청에 따른 이후의 처리 시간이 무한적으로 길어질 수 있다는 단점이 있으므로 본 논문에서는 그림 3처럼 이러한 크롤링 방식을 배제하고 현재 요청된 페이지만 검사하는 방식을 채택하였다. 만약 사용자가 악성코드가 포함된 페이지를 요청하였을 경우 인바운드로 유입되는 패킷을 분석하면 즉시 해당 페이지에 악성코드가 포함되어 있다는 것을 알 수 있기 때문이다.

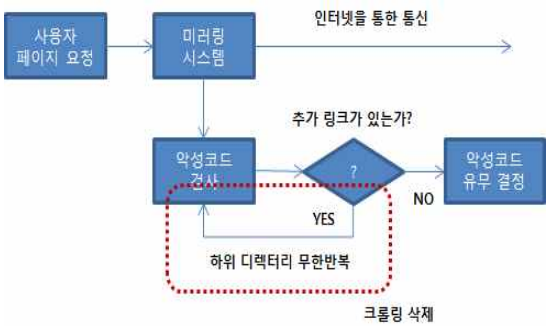

그림 3. 크롤링 배제 후 실시간 페이지 검사
Fig. 3. Realtime page validation after excluding crawling

본 논문에서 사용하는 필터로는 다음과 같은 조건에 의해 실시간으로 해당 페이지만을 필터링한다[5].

필터는 탐지 조건에 따라 확정 탐지 및 의심탐지로 분류된다. 확정 탐지는 시그니쳐 탐지를 기반으로 확인되며 의심탐지는 라인당 글자 수 초과 탐지, 비정상 문자열 포맷 스트링 탐지, 숨김 iframe 탐지, 숨김 applet 탐지, 숨김 object 탐지에 의해 확인 된다.

- 시그니쳐 탐지
  최초 패킷이 왔을 시 해당 URL의 웹 페이지 소스를 다운 받은 후 악성코드 판단 기준이 되는 시그니쳐와 매칭되는 부분이 있으면 악성 코드가 심어져 있는 것으로 판단 및 해당 정보를 저장한다.
- 라인당 글자수 초과 탐지
  한 라인에 특정 바이트의 초과 및 해당 라인에 25% 이상이 숫자로 이루어져 있으면 의심탐지로 분류한다.
- 비정상 문자열 포맷 스트링 탐지
  다운로드 받은 웹 페이지 소스코드 내에 평문 및 난독화 된 스크립트 안에서 문자열 포맷을 추출하여 메모리 주소 및 다른 정보를 보여줄 수 있는 포맷의 사용 및 해당 라인이 특정 바이트 수를 초과할 시 의심탐지로 분류한다.
- 숨김 iframe 탐지
  숨김 iframe 탐지는 다운로드 받은 웹 페이지 소스 코드내에 숨겨진 iframe이 있는지 확인한다. 소스코드 내에서 iframe tag를 추출한 후 width, heigth의 값이 0인 tag만 확인, 정상 iframe 이 아닌 숨겨진 iframe을 추출하여 의심탐지로 분류한다.
- 숨김 applet 탐지, 숨김 object 탐지
  소스 코드내에 applet 및 object tag를 추출하여 width, heigth의 값이 0인 tag만 확인, 숨겨진 applet tag 및 object tag를 추출하여 의심탐지로 분류한다.

확정 탐지인 시그니쳐 탐지로 분류된 URL은 악성코드로 분류되며 의심 탐지인 라인당 글자수 초과 탐지, 비정상 문자열 포맷 스트링 탐지, 숨김 iframe 탐지, 숨김 applet 탐지, 숨김 object 탐지로 분류된 URL은 악성인지를 정확히 판단하기 위하여 동적분석 모듈에 URL을 전달하여 악성 URL인지 확인한다.

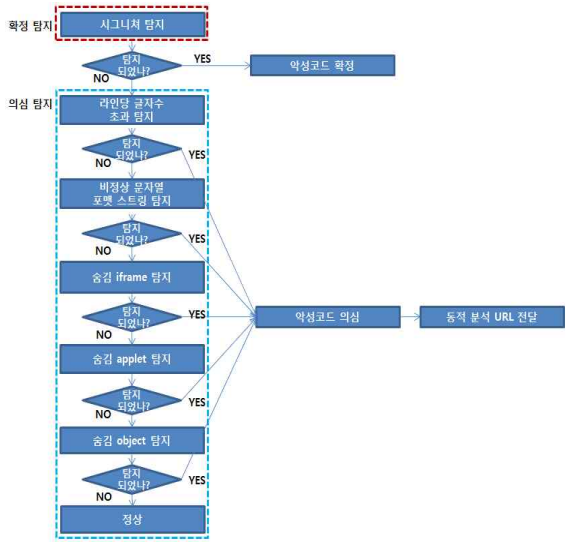

그림 4. 웹 페이지 검사 필터 조건
Fig. 4. Condition of web page validation

이와 같은 두 가지의 필터링을 통해 페이지를 검사할 경우 해당 페이지가 악성코드를 내포하고 있는지 여부를 파악할 수 있다. 하지만, 최근의 경우에는 현재 연구된 필터로도 탐지할 수 없는 유형들이 유포되고 있고 그 중 가장 보편화된 방법이 드라이브 바이 다운로드 기법이다. 이를 해결하기 위해 세 번째 필터를 제안하도록 한다.

3.4 실제 PC 레벨 파일 동적 검사

본 논문에서 제안하는 세 번째 필터로는 드라이브 바이 다운로드를 탐지하는 필터를 제안한다. 드라이브 바이 다운로드는 사용자의 개입 없이 웹 사이트를 방문만 해도 자동으로 파일이 로컬 PC에 설치되어 악성행위를 하는 방식으로 최근 들어 가장 많이 퍼져있는 공격 방법이라 할 수 있으나 마땅한 대응방안이 없는 공격이기도 하다. 따라서 본 논문에서는 실제 PC 환경을 도입하여 의심되는 사이트의 URL를 실제 브라우저로 접속하여 드라이브 바이 다운로드 공격이 이루어지는 지를 확인하고자 한다.

우선 실제 사용할 PC의 환경을 취약한 상태로 유지하는 것이 매우 중요한다. 취약점이 없는 PC의 경우에는 다운로드 되지 않는 형태도 있으므로 여러 환경을 결합하여 탐지한다. 그림 4에서와 같이 제안시스템을 통해 확정되지 않은 사이트가 별견되었을 경우 드라이브 바이 다운로드에 의한 공격인지를 확인하기 위해서 실제 수행을 진행한다. 이때 전체적인 프로세스 관리 및 파일이 다운로드 되는지를 모니터링 하기위해 에이전트를 개발하였다. 본 논문에서 개발한 에이전트는 TDI 계층에서 후킹을 통해 프로세스의 행위를 모니터링하게 된다.

기존의 드라이브 바이 다운로드를 자동적으로 탐지하기 위해 오브젝트나 스트링을 트랙킹하거나 쉘코드의 스트링을 검사하는 방식을 통해 검출해 왔다[12]. 그러나 본 논문에서는 스크립트를 통해 확인하는 것이 아니라 실제 사용자의 개입없이 파일이 다운로드 될 경우 이를 탐지하는 형태를 제안한다. 그림 5와 같이 다양한 환경에서 파일이 다운로드 된다는 것은 분명히 악성코드가 분명하다고 판단하기 때문이다.

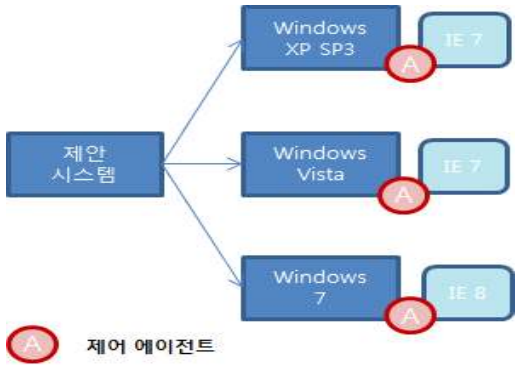

그림 6. 실제 PC 환경에서 에이전트를 이용한 Drive-by-Download 탐지
Fig. 5. Drive-by-download detection using agent on real pc environment

정상적인 흐름의 경우에는 클라이언트와 서버의 프로토콜이 대화방식으로 정해진 반면 비정상적일 경우에는 일방적으로 순서를 무시한 채 서버에 실행 가능한 파일을 요청하여 클

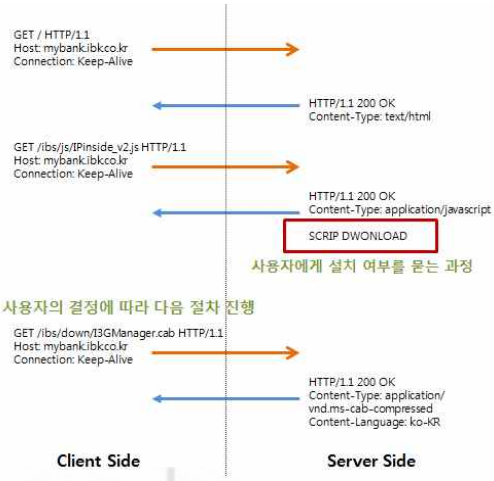

그림 5. 정상적인 웹 통신 흐름
Fig. 6. Normal web traffic flow

라이언트에 다운로드 되는 것을 알 수 있다. 그림 5에서는 일반적인 사이트에서 Active-X 모듈을 사용하기 위한 정상적인 흐름을 나타내고 있고 반드시 사용자의 개입이 있어야만 다음 절차로 넘어가는 것을 알 수 있다. 이러한 정상적인 절차에 대한 기준을 가지고 접근할 때 비정상적인 흐름을 발견하기가 훨씬 더 수월해진다.

그림 6의 정상적인 흐름과는 대조적으로 그림 7의 비정상적인 흐름은 어떠한 사용자의 개입이 없이도 프로그램이 다운로드 되는 것을 알 수 있으며 이를 정상적인 흐름과 비교하여 구별해 낼 수 있다. 정상적인 과정과 비교해보면 중간에 사용자에게 추가적인 파일을 설치하는 여부를 묻는 과정 자체가 생략되었고 초기에 응답을 통해 바로 실행가능 한 파일인 'hello.exe'를 요청하는 것을 알 수 있다.

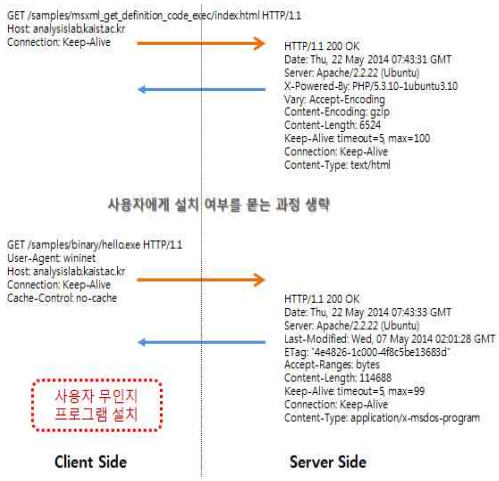

그림 7. 비정상적인 웹 통신 흐름
Fig. 7. Abnormal web traafic flow

위의 과정을 좀 더 세밀하게 분석을 하여 다음과 같은 흐름을 가질 때 비정상적인 흐름이며 악성코드 다운로드로 간주하여 해당 세션을 차단한다. 그림 7은 비정상적인 흐름에 대한 처리를 담당하는 엔진의 알고리즘을 기술하고 있다. 이 알고리즘을 통해 사용자 PC로 강제적으로 다운되는 파일을 막을 수 있다.

```
If (user request is first)
{
  CreateSession(user); // 초기 세션 생성

  SendUserReqest(); // 초기 서버 응답

  if (Server Request != NULL) // 두번째 응답
  {
    FindUserSession(); // 세션 검색
    reqData = ResponseData(); // 응답 데이터 분석
    if (reqData is Contain Executable File) // 악성코드 요청
    {
      DeleteSession();      // 세션 종료 및 악성 URL 등록
      EnrollMalware_URL();
      Alarm();
    }
  }
}
```

그림 8. 비정상 흐름에 대한 처리 알고리즘
Fig.8 propose algorithm of abnormal flow

추가적으로 고려해야할 사항은 그림8과 같이 비정상적으로 사용자가 인지하지 못하는 드라이브 바이 다운로드의 파일을 임의로 다운받아 이에 대한 패턴을 만들어 배포하여 이미 다운로드 되었거나 또는 다른 사이트에서 유입된 악성코드를 찾아내어 삭제하는 것도 중요한 요소라고 할 수 있다. 다만, 본 논문에서는 이러한 패턴 생성 및 치료에 대한 부분은 배제하기로 한다.

본 논문은 위에서 제시한 3가지의 다중 필터링 시스템을 통하여 악성코드가 사용자 PC에 다운로드 되기 전에 이를 탐지하고 새로운 기법에 의한 방식으로 드라이브 바이 다운로드를 처리할 수 있는 방안을 제시하였다. 이러한 필터가 복합적으로 이루어질 경우 기존의 시스템의 처리 방식에 비해 매우 효과적이고 오탐이 없는 방안이라 할 수 있겠다. 다음 장에서 이 시스템에 대한 실험결과를 제시하도록 하겠다.

Ⅳ. 실험 및 고찰

본 논문의 제안을 검증하기 위하여 본론에서 제시한 다중 필터를 프로토타입의 형태로 개발하여 실험하였다. 또한 드라이브 바이 다운로드의 지속적인 실험을 위한 실험 사이트로 두 개의 사이트를 직접 제작하여 운영하였고 외부의 사이트도 함께 실험을 진행하여 본 제안에 대한 검증을 수행하였다. 그림 9의 제안시스템의 내용은 그림 1에서 제시한 내용과 동일하며 외부의 드라이브 바이 다운로드를 위해 두 개의 실험 사이트를 제작하여 실험하였으며 실험 환경의 제약으로 인해 사용자 PC의 환경을 크게 두 종료로 나누어서 실험하였다. 이러한 실험을 진행한 두 사이트 및 실험 운영 환경은 다음과 그림 9와 같다.

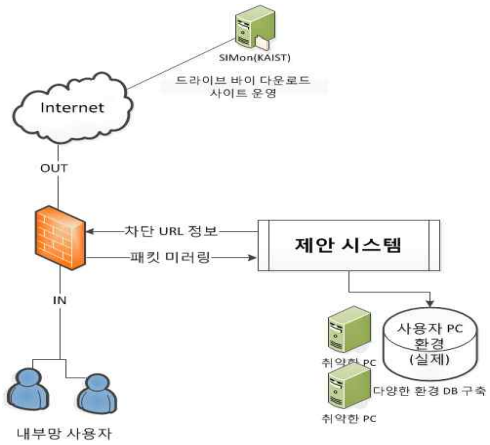

그림 9. 실험 환경
Fig. 9. Test environment

■ 첫 번째 사이트
- http://analysislab.kaist.ac.kr/samples/ms13_037_svg_dashstyle/index.html
- 취약점 : CVE-2013-2551
- 테스트 환경 : Windows 7 / IE 8

■ 두 번째 사이트
- http://analysislab.kaist.ac.kr/samples/msxml_get_definition_code_exec/index.html
- 취약점 : CVE-2012-1889
- 테스트 환경 : Windows XP SP3 / IE 7

■ 악성코드 파일 : calc.exe를 sample.exe로 변경 후 사용

위의 실험 환경을 통해 첫 번째 사이트를 방문하였을 경우 비정상적인 흐름을 감지하였고 이에 그림 10과 같은 결과를 출력 하였다.

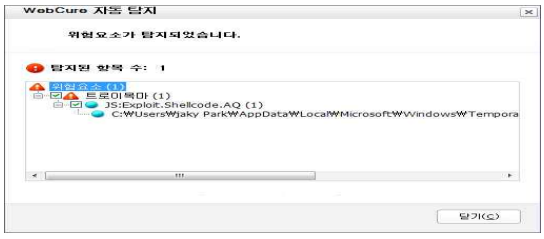

그림 10. 드라이브 바이 다운로드 자동 탐지 1
Fig. 10. No 1. - Auto detection of drive-by-download

두 번째 실험에서도 동일한 형태의 결과를 출력하였고 본 논문에서 제안한 다중 필터의 주장을 검증할 수 있었다.

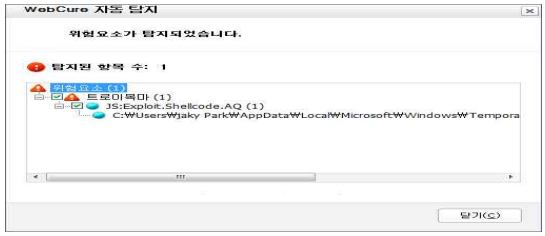

그림 11. 드라이브 바이 다운로드 자동 탐지 2
Fig. 10. No 2. - Auto detection of drive-by-download

추가적으로 해당 사이트를 외부의 공신력 있는 사이트를 통해 검증하기 위해 VirusTotal[11] 사이트에서 확인하였다.

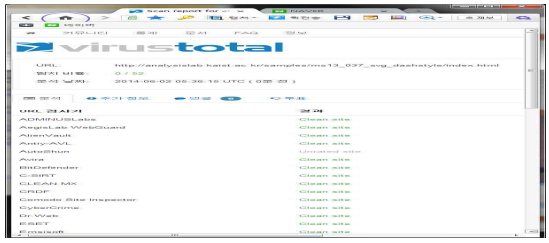

그림 12. Virustotal 검증 결과
Fig. 12. Validation result of Virustotal

그러나 예상한 것과 같이 VirusTotal에서는 해당 사이트에 대한 위협을 감지 못했다. 즉, 해당 사이트가 가지고 있는 바이너리 파일의 악성코드 여부만을 따지므로 본 실험에서 사용한 바이너리 파일은 calc.exe를 이름만 변경하여 사용한 것이므로 정상적인 것으로 판단한 것이다. 하지만 이러한 드라이브 바이 다운로드를 통해 얼마든지 악성코드를 사용자 PC에 다운로드 할 수 있음으로 인해 본 논문의 엔진이 매우 효과적임을 입증하였다. 추가적으로 국내에서 운영 중인 malwares.com에도 동일한 실험을 하였으나 그림 12와 같이 VirusTotal과 마찬가지로 정상적인 결과를 보였다.

본 논문에서 제시한 다중 필터는 우선적으로 기존에 발견된 악성코드 URL을 커널 레벨에서 고속으로 탐지하고 또한 응용 레벨에서 크롤링을 통하여 탐지함으로써 패턴이 없는 사이트도 실시간으로 탐지할 수 있는 방안을 제시하였다. 마지막으로 드라이브 바이 다운로드를 근본적으로 차단함으로 인해 악성코드가 사용자의 허가 없이 직접 다운되는 것을 막는 방안을 제시하였다. 이러한 과정을 통해 악성코드 자체가 어떠한 행위를 하는지를 알기 전에 악성코드를 숨겨놓은 사이트를 찾아서 막음으로 인해 보다 효과적인 방어 방안을 제시하였다고 볼 수 있다.

Ⅴ. 결론

본 논문에서는 기존의 악성코드를 직접 분석하거나 패턴에 의해 탐지하는 것이 아니라 악성코드를 웹 사이트에 숨겨 놓거나 해킹을 통해 사용자가 인지하지 못하는 방식으로 배포하는 것을 원천적으로 차단할 수 있는 방안을 제시하였다. 본 논문에서는 커널 레벨에서의 기존 URL을 차단하는 필터와 실시간으로 방문할 시점의 사이트를 크롤링 기법에 의해 추적하는 필터와 드라이브 바이 다운로드에 의한 악성코드 다운을 탐지할 수 있는 기법을 다중 틸터 형태로 제안하였다. 이에 대한 실험을 위해 실제 드라이브 바이 다운로드 사이트를 제작하였고 객관적인 검증을 위해 VirusTotal과 같은 글로벌 사이트에서 추가 검증을 진행하였다. 본 제안된 기법을 통해 실제 네트워크에서 활용할 수 있는 시스템을 제작할 경우 매우 효과적인 기법이라고 할 수 있다. 향후 악성코드 자체를 분석할 수 있는 기능 연구가 추가되어야 할 것이다. 본 연구를 통해 기업과 기관 및 개인에 적용할 수 있는 악성코드 탐지 기술을 보급함으로써 알려지지 않은 악성코드에 대한 예방 및 방어를 할 수 있을 것으로 기대한다.

참고문헌

〔1〕 http://www.ahnlab.com/kr/site/securityinfo /secunews/secuNewsView.do?menu_dist=2&seq=22325

〔2〕 Hyo-Nam Kim, Realtime hybrid analysis based on multiple profile for prevention of malware, Hongik Univ., Feb. 2014

〔3〕 Jin-Kyung Kim, A design of anomaly detection with automata dynamic profile, Hansei Univ., Feb. 2014

〔4〕 Provos, N., McNamee, D., Mavrommatis, P., Wang, K., and Modadugu, N. "The ghost in the browser analysis of web-based malware," Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, pp. 4-4, Apr. 2007.

〔5〕 Sang-Yong Choi, Multi-level emulation for malware distribution networks analysis, Journal of The Korea Institute of Information Security & Cryptology, VOL.23, NO.6, Dec. 2013.

〔6〕 Chang-Wook Park, First URL lookup using URL prefix hash tree, Journal of The Korea Institute of Information Science, Vol. 35, No.1, pp. 67-75, Oct. 2007.

〔7〕 *https://www.Malwares.com*

〔8〕 Chen, K.Z., Gu, G., Zhuge, J., Nazario, J., and Han, X.., "WebPatrol: Automated collection and replay of web-based malware scenarios," Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security, pp.186-195, Mar. 2011.

〔9〕 SpiderMonkey, "https://developer.mozilla.org/en-US/ docs/Mozilla/Projects/SpiderMonkey"

〔10〕 Yongwook Lee, Design and implementation of web-browser based malicious behavior detection system(WMDS), Journal of The Korea Institute of Information Security & Cryptology, Vol.22, No.3, pp. 667-677, Jun. 2012.

〔11〕 Manuel Egele, Peter Wurzinger, Christopher Kruegel, and Engin Kirda, Defending Browsers against Drive-by Downloads: Mitigating Heap-spraying Code Injection Attacks, ACM New York, pp. 281-290, 2010.

〔12〕 *https://www.VirusTotal.com*

저 자 소 개

박 재 경

1994: 동국대학교 컴퓨터공학과 공학사

1996: 홍익대학교 전자계산학과 이학석사.

2002: 홍익대학교 전자계산학과 이학박사

현 재: 학국과학기술원 사이버보안연구센타 책임연구원

관심분야: 네트워크 보안, 사이버 보안

Email : wildcur@kaist.ac.kr