Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.24, NO.1, Feb. 2014 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2014.24.1.107

디지털 포렌식 기법을 활용한 알려지지 않은 악성코드 탐지에 관한 연구*

이 재 호,[1†] 이 상 진[2‡]
고려대학교 정보보호대학원

A Study on Unknown Malware Detection using Digital Forensic Techniques*

Jaeho Lee,[1†] Sangjin Lee[2‡]
Center for Information Security Technologies(CIST), Korea University

요 약

DDoS 공격과 APT 공격은 좀비 컴퓨터들로 정해진 시간에 동시에 공격을 가하여 사회적 혼란을 유발하였다. 이러한 공격에는 공격자의 명령을 수행하는 많은 좀비 컴퓨터들이 필요하며 좀비 컴퓨터에는 안티바이러스 제품의 탐지를 우회하는 알려지지 않은 악성코드가 실행되어야한다. 그동안 시그니처로 탐지하던 안티바이러스 제품을 벗어나 알려지지 않은 악성코드 탐지에 많은 방법들이 제안되어 왔다. 본 논문은 디지털 포렌식 기법을 활용하여 알려지지 않은 악성코드 탐지 방법을 제시하고 정상 파일과 악성코드의 다양한 샘플들을 대상으로 수행한 실험 결과에 대하여 기술한다.

ABSTRACT

The DDoS attacks and the APT attacks occurred by the zombie computers simultaneously attack target systems at a fixed time, caused social confusion. These attacks require many zombie computers running attacker's commands, and unknown malware that can bypass detecion of the anti-virus products is being executed in those computers. A that time, many methods have been proposed for the detection of unknown malware against the anti-virus products that are detected using the signature. This paper proposes a method of unknown malware detection using digital forensic techniques and describes the results of experiments carried out on various samples of malware and normal files.

Keywords: Zombie Computer, Unknown Malware Detection, Digital Forensics

I. 서 론

'7.7, 3.4 DDoS(Distributed Denial Of Service, 분산서비스거부) 공격'은 많은 양의 패킷을 동시에 공격 목표로 보내 정상적인 서비스를 제공할 수 없도록 하였다. 이 후 금융, 언론사, 기관, 기업 등을 타깃으로 하는 APT(Advanced Persistent Threat, 지능형지속보안위협) 공격은 '3.20, 6.25 사이버 공격' 형태로 사회적 이슈가 되고 있다. 이러한 모든 공격에는 알려지지 않은 악성코드에 감염된 좀비 컴퓨터(zombie computer)가 이용된다. 공격자는 좀비 컴퓨터의 확산을 위하여 개인 사용자에게 보편화된 윈도우 운영체제에서 실행 가능한 악성코드를 제작한다. 이 악성코드는 시그니처(signature) 방식으로 탐지하던 안티바이러스(anti-virus) 제품들을 우회하고 제로데이(zero-day) 취약점을 포함한 다양한

접수일(2013년 10월 8일), 수정일(2013년 12월 2일), 게재확정일(2013년 12월 3일)
* 이 논문은 2013년도 정부(미래창조과학부)의 재원으로 한국연구재단-공공복지안전사업의 지원을 받아 수행된 연구임(2012M3A2A1051106)
† 주저자, explod@korea.ac.kr
‡ 교신저자, sangjin@korea.ac.kr (Corresponding author)

방식으로 유포되어진다. 따라서 윈도우 운영체제에서 알려지지 않은 악성코드를 신속히 확보할 수 있는 방법이 필요하게 되었다.

그동안 악성코드의 흔적을 찾는 디지털 포렌식 기법과 정적 분석, 동적 분석 등을 이용하여 알려지지 않은 악성코드를 탐지하는 방법이 각각의 분야에서 왕성하게 연구되어 왔다. 본 논문은 각 분야의 연구를 보완하여 디지털 포렌식 기법을 활용한 알려지지 않은 악성코드 탐지 방법을 제시한다.

II. 관련 연구

2.1 침해사고 대응 절차

정보통신망 이용촉진 및 정보보호 등에 관한 법률에 따르면 침해사고란 해킹, 컴퓨터바이러스, 논리폭탄, 메일폭탄, 서비스 거부 또는 고출력 전자기파 등의 방법으로 정보통신망 또는 이와 관련된 정보시스템을 공격하는 행위를 하여 발생한 사태를 말한다[1]. 이러한 침해사고에 신속한 대응으로 피해 확산을 막고 재발 방지를 수행하기 위해 침해사고 대응 절차를 수립해야 한다[2][3][4]. 침해사고 대응 절차는 연구기관 마다 차이가 있으나 일반적으로 Fig.1.의 단계들로 구성되어 진다.

Fig.1. An incident handling process of SANS[5]

Fig.1.의 탐지와 분석 단계에서는 이벤트들을 수집하고 분석한다. 그리고 어떠한 사고인지 판단하게 된다. 해당 자료[4][6]와 포렌식 서비스로 침해사고를 식별하는 자료들[7][8]은 침해사고 조사에 필요한 항목들을 알려준다. 그러나 많은 정보로 인해 분석가들에게 혼란을 불러오고 분석가들의 경험에 따라 다른 판단 결과를 가져온다.

침해사고 이후 분석가들의 능력에 상관없이 자동으로 악성코드를 탐지할 수 있는 방법이 필요하다.

2.2 제안된 악성코드 분석 기반의 탐지 방법

Fig.2.를 보면 새로이 만들어진 악성코드가 기하급수적으로 증가하는 것을 알 수 있다. 시그니처 방식의 안티바이러스 제품은 새로운 악성코드가 만들어지는 속도에 맞추어 대응하는데 어려움을 겪게 되었다. 시그니처 방식을 벗어나 새로운 악성코드 또는 변종을 탐지하기 위해 많은 방법들이 제안되어왔다.

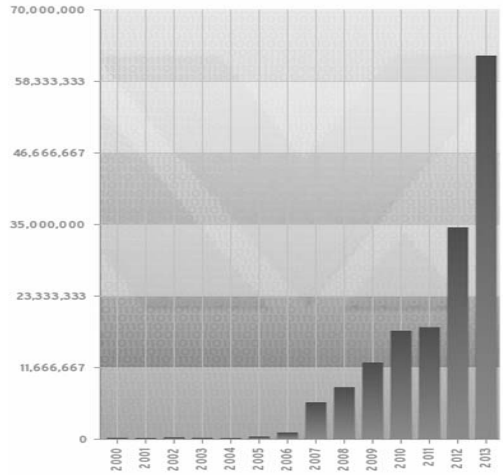

Fig.2. The statistics of the number of malware by year from AV-TEST

악성코드는 크게 세 가지 방법으로 분석할 수 있는데 정적(static) 분석, 동적(dynamic) 분석 그리고 하이브리드(hybrid) 분석으로 나뉜다.

정적 분석은 악성코드를 실행하지 않고 구성 요소들의 연관성 및 실행 코드 등을 분석하는 방법이다. 이러한 정적 분석을 이용한 악성코드 탐지 방법으로는 실행파일의 주요 블록을 비교하거나 API의 순차적 특징을 이용하거나 파일의 시그니처로 n-grams를 이용하는 방법 등[9][10][11]이 제안되었다.

동적 분석은 분석환경 내에서 악성코드를 실행하고 모니터링하여 악성코드의 행위와 시스템내의 행위를 분석하는 방법이다. 이러한 동적 분석을 이용한 악성코드 탐지 방법으로는 정보 흐름의 원인이 되는 프로그램 조각을 추출하고 실행하여 행위 특징을 비교하거나 레지스트리 모니터링에 따른 통계 정보를 이용하는 방법 등[12][13]이 제안되었다.

하이브리드 분석은 정적 분석과 동적 분석을 결합하여 악성코드를 분석하는 방법이다. 하이브리드 분석

을 이용한 악성코드 탐지 방법으로는 악성코드의 공통 속성을 이용하거나 공격용 툴킷의 특징을 이용하는 방법 등[14][15]이 제안되었다.

동적 분석과 하이브리드 분석은 제한된 환경 구축과 실행이 수반되는 특징 때문에 악성코드 분석 속도가 느리다. 또한 특정 악성코드로 분류되고 이에 해당하는 악성코드를 기준으로 탐지하는 방법들은 다른 종류의 악성코드 탐지에 어려움이 있다.

악성코드를 효과적으로 탐지하기 위해서는 정적 분석의 속도와 하이브리드 분석에서 얻을 수 있는 정보들이 필요하다.

III. 탐지 방법 연구

악성코드는 이메일, 메신저, 웹, USB, 응용 프로그램의 업데이트 등 여러 경로에서 유입될 수 있다. 예를 들어 웹 사이트가 해커에게 이미 장악되어 악성코드를 배포하고 있다면 사용자는 해당 웹 사이트에 접근하는 것만으로도 악성코드에 감염될 수 있다. 그리고 악성코드는 감염 이후에 지속성을 유지하기 위해 자동으로 실행될 수 있는 다양한 방법들을 이용한다. 이렇게 실행된 악성코드에 대해 운영체제는 관련 정보를 흔적으로 남기게 된다. 물론 정상 파일과 관련한 많은 흔적들도 포함된다. 운영체제에서 남긴 흔적 정보만을 대상으로 한다면 조사에 필요한 파일의 개수를 줄일 수 있다. 디스크에 존재하는 모든 파일을 대상으로 하는 안티바이러스 제품에 비해 속도 향상을 기대할 수 있다. 이러한 흔적 정보를 대상으로 정상 파일보다 악성코드에 더 높은 확률로 존재하는 항목들을 찾는다면 악성코드의 가능성을 판단할 수 있을 것이다.

3.1 분석 대상의 정보

악성코드나 정상 파일에 상관없이 실행파일(PE, Portable Executable)이 메모리에 로드되어 실행되면 운영체제는 여러 정보를 변경하고 흔적을 남기게 된다.

Table 1.은 디지털 포렌식 기법으로 얻을 수 있는 많은 정보 중에서 실행파일의 흔적을 찾을 수 있는 항목들이다. 알려지지 않은 악성코드를 탐지하기 위해 실행 흔적 정보, 실행 모듈 정보, 자동시작 위치 정보, 인터넷 정보를 분석 대상으로 수집한다.

Table 1. Information about subject of analysis

Information	Category	Collection Method
Execution Trace	Prefetch	File Format
	UserAssist	Registry
	MUICache	Registry
Execution Module	Loaded PE (exe,dll,sys)	Windows API
	Service	Registry
	Network	Windows API
Automatic Execution	Run	Registry
	Shell	Registry
Internet	Index.dat	File Format

3.1.1 실행 흔적 정보

실행 흔적 정보는 프로그램 실행 이후 남는 로그와 같은 정보로 프로그램이 삭제되어도 실행과 관련된 부가 정보를 얻을 수 있다.

3.1.1.1 프리패치 파일

프리패치(prefetch) 파일은 시스템의 부팅 시간, 응용 프로그램 실행의 시작 시간을 단축하기 위한 방법으로 코드 및 데이터를 효과적으로 로드하기 위해 윈도우 운영체제에서 생성하는 파일이다.

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x00	Prefetch version				Signature (SCCA)				Service version				Prefetch file size			
0x10	Name of executable as unicode string															
0x20																
0x30																
0x40													Prefetch hash			
0x50	0 or 1				Section Info Offset				Number of Sections				Page Info Offset			
0x60	Number of Pages				Filepaths Info Offset				Filepaths Info Size				Metadata Info Offset			
0x70	Number of Metadata Records				Metadata Info Size				Last execution time of executable (FILETIME)							
0x80	MinRePrefetch Time								MinReTrace Time							
0x90	Execution counter				Sensitivity											

Fig.3. Prefetch file format (Windows XP)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x00					Signature (SCCA)											
0x10	Name of executable as unicode string															
0x20																
0x30																
0x40													Prefetch hash			
0x50																
0x60					Filepaths Info Offset				Filepaths Info Size							
0x70																
0x80	Last execution time of executable (FILETIME)															
0x90									Execution counter							

Fig.4. Prefetch file format (Windows Vista, 7)

Windows XP 이후로 프리패칭 기능을 보유하게 되었으며 Windows XP, Vista, 7이 기본적으로 응용 프로그램 프리패칭을 수행한다. 프리패치 파일은 %systemroot%\Prefetch 폴더 안에 pf 확장자로 존재하며 Fig.3.과 Fig.4.와 같이 윈도우 운영체제 버전에 따라 2가지 파일 포맷을 지원한다.

프리패치 파일의 존재는 해당 실행파일이 메모리에 로드되어 실행되었던 흔적 정보이다. 또한 Fig.3.과 Fig.4.의 프리패치 파일 포맷에서 Filepaths Info Offset과 Filepaths Info Size를 통해 실행파일과 함께 로드되는 DLL 파일들의 경로와 이름을 얻을 수 있다.

3.1.1.2 레지스트리의 UserAssist

운영체제는 사용자에게 편의성을 제공하기 위하여 사용자의 행위 내역을 레지스트리의 UserAssist에 ROT-13으로 암호화하여 저장한다[16].

Fig.5. UserAssist in the registry

레지스트리의 UserAssist에서 ROT-13으로 암호화된 알파벳 문자는 그 뒤 13번째의 문자로 치환하여 Fig.6.과 같이 복호화 할 수 있다.

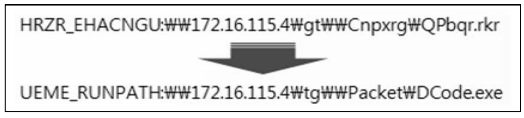

Fig.6. ROT-13 decryption

복호화된 문자에서 과거 실행되었던 프로그램의 경로와 이름을 얻을 수 있다.

3.1.1.3 레지스트리의 MUICache

레지스트리의 MUICache(Multilingual User Interface Cache, 다중언어지원 캐시) 키(key)는 다중언어를 지원하기 위해 시스템 내에서 실행되는 프로그램의 이름을 캐싱한다[17].

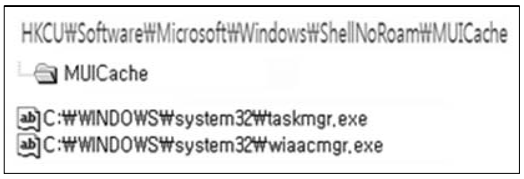

Fig.7. MUICache in the registry

Fig.7.과 같이 MUICache 키의 각각의 값(value)에서 실행된 프로그램의 경로와 이름을 얻을 수 있다.

3.1.2 실행 모듈 정보

윈도우 운영체제에서 실행된 프로그램은 프로그램 종료 이전까지 실행 모듈 정보에 해당 프로그램과 관련된 정보를 유지한다.

3.1.2.1 로드된 실행파일

모든 프로그램들은 실행되어지기 위해서 프로세스, 동적 로드 라이브러리, 드라이버와 같은 형태로 메모리에 로드되어야 한다.

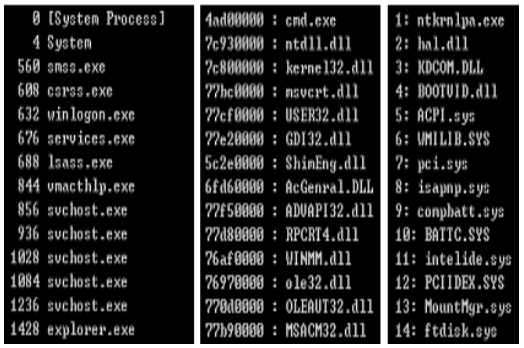

Fig.8. The list of processes, dynamic load libraries, and drivers

실행 중인 프로세스(exe), 동적 로드 라이브러리(dll), 드라이버(sys)의 목록 정보를 Fig.8.과 같이 Windows API를 이용하여 수집한다.

3.1.2.2 서비스

윈도우 서비스는 시스템이 부팅될 때 백그라운드로 실행되는 프로그램이다. 윈도우 서비스로 등록되면 레지스트리에서 파일의 정보를 확인할 수 있다.

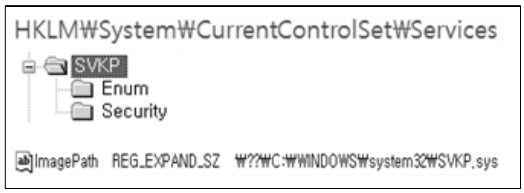

Fig.9. The information of programs that are run as Windows Service

HKLM\System\CurrentControlSet\Services\〈서비스 이름〉 키의 ImagePath 값에서 윈도우 서비스로 실행되는 프로그램의 경로와 이름을 얻을 수 있다.

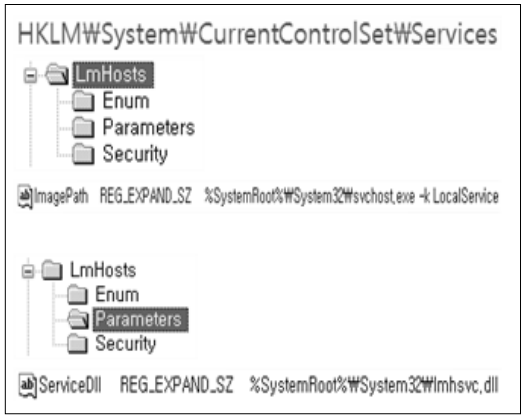

Fig.10. The information of DLL loaded together with the Svchost process

또한 ImagePath 값에 'svchost.exe'가 있을 경우 HKLM\System\CurrentControlSet\Services\〈서비스 이름〉\Parameters 키의 ServiceDll 값에서 Svchost 프로세스와 함께 로드되는 DLL 파일의 경로와 이름을 얻을 수 있다.

3.1.2.3 네트워크 연결 정보

네트워크 연결 정보는 현재 시스템과 네트워크 연결이 이루어지는 상태 정보를 보여준다. 로컬 IP, 로컬 Port, 원격 IP, 원격 Port, 프로세스 등의 정보를 제공한다.

Fig.11.과 같이 Windows API를 이용하여 네트워크 연결에 이용되거나 포트를 열고 있는 프로세스 정보를 수집한다.

Proto	Local Address	Foreign Address	State	PID
TCP	0.0.0.0:135	0.0.0.0:0	LISTENING	900
TCP	0.0.0.0:445	0.0.0.0:0	LISTENING	4
TCP	127.0.0.1:1029	0.0.0.0:0	LISTENING	1024
TCP	192.168.1.128:139	0.0.0.0:0	LISTENING	4
TCP	192.168.1.128:3494	192.168.1.93:139	SYN_SENT	3700
TCP	192.168.1.128:3495	192.168.1.94:139	SYN_SENT	3700
TCP	192.168.1.128:3496	192.168.1.95:139	SYN_SENT	3700
TCP	192.168.1.128:3497	192.168.1.96:139	SYN_SENT	3700
TCP	192.168.1.128:3502	192.168.1.97:139	SYN_SENT	3700
TCP	192.168.1.128:3505	192.168.1.98:139	SYN_SENT	3700
TCP	192.168.1.128:3506	192.168.1.99:139	SYN_SENT	3700
TCP	192.168.1.128:3507	192.168.1.100:139	SYN_SENT	3700
TCP	192.168.1.128:3508	192.168.1.101:139	SYN_SENT	3700
TCP	192.168.1.128:3509	192.168.1.102:139	SYN_SENT	3700

Fig.11. Network connection information

3.1.3 자동시작 위치 정보

3.1.3.1 레지스트리의 Run, Shell

시스템이 부팅되거나 사용자가 로그인할 때 레지스트리의 Run, Shell에 등록된 프로그램은 자동으로 실행된다.

Table 2. Run in the registry

HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServicesOnce HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunServices 〈Logon Prompt〉 HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\RunOnce

Table 2.는 부팅 순서에 따른 Run의 실행 순서이다. 각각의 레지스트리 키에서 프로그램의 경로와 이름을 얻을 수 있다.

Table 3. Shell의 각각의 레지스트리 키에서 실행 파일과 DLL 파일의 경로와 이름을 얻을 수 있다.

Table 3. Shell in the registry

HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\Notify

Winlogon 프로세스는 사용자가 로그인한 이후, 초기 프로세스를 생성하기 위하여 Table 3.의 Winlogon 키의 Userinit 값에 해당하는 프로그램을 실행한다. 또한 Winlogon 프로세스는 사용자의 로그인, 쉘 로드, 스크린세이버 시작 등을 통지받기 위해 Table 3.에서 Notify 키의 서브키에 DLLName 값으로 등록된 DLL 파일들을 로드한다.

3.1.4 인터넷 정보

3.1.4.1 Index.dat 파일

index.dat 파일은 Internet Explorer 웹 브라우저에 의해 이용되는 데이터베이스 파일로 웹 URL, 검색어, 최근 열어본 파일과 같은 정보들을 포함하고 있다.

Table 4. The path of the index.dat file

Windows Version	The path of the index.dat file
XP	Documents and Settings\<username>\Local Settings\Temporary Internet Files \Content.IE5
Vista, 7	Users\<username>\AppData\Local\Microsoft\Windows\Temporary Internet Files\Content.IE5

index.dat 파일은 MSIECF(MS IE Cache File) 포맷으로 되어 있다[18].

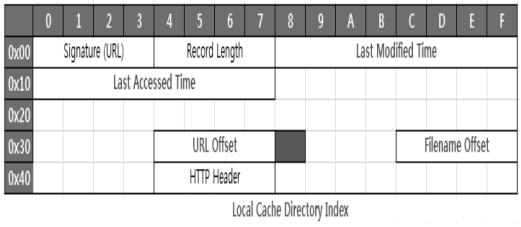

Fig.12. URL Activity Record in MSIECF format

Fig.12.는 MSIECF 포맷의 URL Active Record 부분이다. 해당 부분에는 인터넷으로부터 다운 받은 파일의 이름과 인터넷 경로(URL, Uniform Resource Locator) 등의 정보가 포함되어 있다.

3.2 악성코드 판단 항목

Table 5.는 악성코드 판단을 위한 항목들로 3.1절에서 수집한 정보를 분석 대상으로 한다.

Table 5.의 항목들은 악성코드에 존재하는 주요 특징들이다. 그러나 정상 파일에서도 해당 특징들을 찾아 볼 수 있다. 본 논문에서는 해당 특징들이 악성코드에 더 높은 확률로 존재할 것이라 가정하고 실험을 통해 확인한다.

Table 5. Malware testing standards

Information	Standards
Sign	Catalog
	Digital Signature
Module	Loaded PE
	Network Connection
	DLL Injection
File	Malware Path
	Packing
	File Properties
	File Attribute
	Suspicious File Name
	Abnormal File Path
	File Signature
	File Size
	Malicious API
Registry	Run
	Shell
Time	Service

악성코드 판단 항목으로 서명 정보, 모듈 정보, 파일 정보, 레지스트리 정보, 시간 정보를 이용한다. 그리고 해당 항목들이 악성코드와 어떤 관련이 있는지 다음에서 설명한다.

3.2.1 서명 정보

3.2.1.1 카탈로그

WFP(Windows File Protection) 기능은 프로그램에 의해 중요한 윈도우 시스템 파일이 변경되지 않도록 한다. 시스템 파일이 변경 되거나 설치 될 경우 WFP는 카탈로그(catalog) 파일에서 해시 값을 찾아 새로운 파일의 해시 값과 비교한다. 이 파일의 해시 값이 다르다면 WFP는 해당 파일을 캐시 폴더(파일이 캐시 폴더에 있는 경우) 또는 설치 원본에 있는 파일로 교체한다.

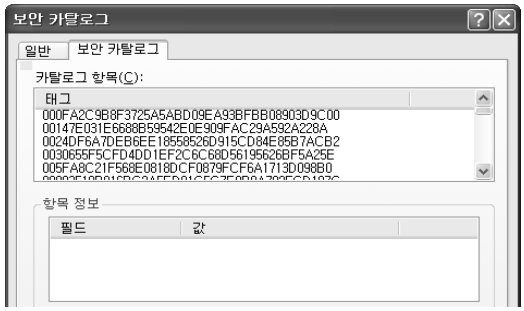

Fig.13. A list of hash values in a catalog file

악성코드가 WFP를 무력화하고 윈도우 시스템 파일을 변경하였다면 카탈로그 파일에 들어 있는 해시 값을 비교하여 악성코드 중 바이러스(virus), 패치드(patched) 형태의 트로이목마를 탐지할 수 있다. 반면 카탈로그 파일에 들어있는 해시 값과 동일한 윈도우 시스템 파일은 정상 파일로 악성코드 판단을 진행하지 않는다.

3.2.1.2 디지털 서명

디지털 서명(digital signature) 정보는 특정 기업이나 단체에서 해당 파일은 자신들에 의해 개발 및 제작된 것이며 위변조가 되지 않았다는 것을 증명하는 용도로 사용하고 있다.

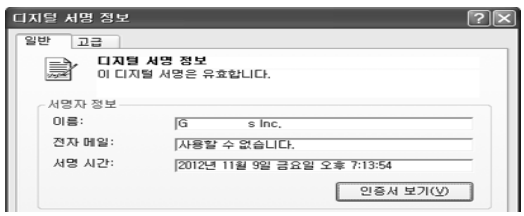

Fig.14. The digital signature of a file

악성코드 생성기로 자동으로 만들어진 악성코드들은 디지털 서명을 사용하지 않았으나 최근에 몇몇 악성코드들은 APT 공격으로 보안 제품들을 우회하기 위해 디지털 서명을 이용한다.

3.2.2 모듈 정보

3.2.2.1 로드된 실행파일

악성코드도 프로그램이기 때문에 실행되기 위해서는 메모리에 로드되어 동작해야만 한다. 만약 시스템이 좀비 컴퓨터가 되어 악성코드가 실행 중이라면 로드된 실행파일 정보에서 동작중인 악성코드를 찾을 수 있다.

3.2.2.2 네트워크 연결 정보

악성코드가 네트워크로 제어되거나(IRCBot, 악성코드 진단명) 악의적인 행위를 하거나(DDoS, 악성코드 진단명) 감염시킬 다른 시스템을 검색 및 배포(Worm, 악성코드 진단명)할 경우 네트워크 연결 정보를 통해 확인할 수 있다.

3.2.2.3 DLL 인젝션

DLL 인젝션(injection)은 악성코드로 만든 DLL을 다른 프로세스의 메모리 공간에 로드하여 원하는 코드를 실행하는 것을 의미한다.

Winlogon, Explorer 프로세스는 사용자가 로그인 한 이후 항상 메모리에 로드되어 있는 프로세스로 악성코드의 표적이 된다. Winlogon, Explorer 프로세스에 로드되는 DLL 파일들은 일반적으로 카탈로그 파일 내에 해시 값을 가진 정상 파일들이다. 이러한 정상 파일들을 제외하고 로드되어진 DLL 파일이 다른 프로세스들에도 함께 로드되어 존재한다면 DLL 인젝션으로 탐지한다.

3.2.3 파일 정보

3.2.3.1 악성코드 경로

악성코드는 사용자에게 정상 파일인 것처럼 속이기 위해 Table 6.과 같이 정상 파일들이 많이 존재하는 시스템 폴더, 드라이버 폴더 등에 주로 존재한다.

Table 6. The main paths of malware

- %TEMP%
- %WINDIR%
- %WINDIR%\Temp
- Root Directory
- %WINDIR%\system
- %WINDIR%\system32
- %WINDIR%\system32\dllcache
- %WINDIR%\system32\drivers
- %WINDIR%\Downloaded Program Files
- CSIDL_COMMON_APPDATA*
- CSIDL_APPDATA*
- CSIDL_LOCAL_APPDATA*
- CSIDL_COMMON_STARTUP
- CSIDL_STARTUP
- CSIDL_INTERNET_CACHE

Table 7.에 해당하는 경로의 경우 윈도우 탐색기에서 실행파일이 보이지 않거나 실행파일 자체가 존재하지 않는 경로들이다. 만약 해당 경로에 실행파일이 존재 한다면 악성코드일 확률이 높다.

Table 7. The paths without any executable files

- %WINDIR%\Task - CSIDL_FONTS - CSIDL_COMMON_FAVORITES - CSIDL_FAVORITES - CSIDL_PROFILE - CSIDL_APPDATA

CSIDL(constant special item ID list)는 응용 프로그램에 의해 주로 이용되는 특별한 폴더로서 식별을 위해 시스템에서 독립된 유일한 값을 제공한다.

3.2.3.2 패킹

패킹(packing)은 실행파일의 크기를 줄일 수 있고 악성코드 분석을 어렵게 하여 판단을 흐리게도 한다. 이러한 이유로 악성코드 제작자들은 패킹을 자주 이용한다.

한승원은 '악성코드 포렌식을 위한 패킹파일 탐지에 관한 연구'에서 패킹파일을 탐지하는 방법으로 두 가지를 소개 하였다[19]. 그 중 한 가지 방법은 진입점 섹션의 WRITE 속성의 포함 여부로 패킹파일을 판단하였는데 총 100개의 샘플로 실험한 결과 98% 확률로 패킹 여부를 확인할 수 있었다. 이러한 패킹파일 탐지 방법은 많은 파일들에 대해 신속한 판별이 가능하여 본 논문에서 패킹의 판단 여부를 확인하는데 이용한다.

Fig.15. File properties

3.2.3.3 등록 정보

Fig.15.에서 볼 수 있듯이 파일의 원본을 확인하기 위해 등록 정보의 제품 이름, 파일 버전, 회사 정보를 이용한다.

과거에는 대부분의 악성코드에서 등록 정보의 제품 이름, 파일 버전, 회사 정보를 볼 수 없었다. 그러나 최근에는 정상 파일처럼 속이기 위하여 등록 정보를 포함하는 악성코드들이 빈번히 발견된다.

3.2.3.4 파일 속성

파일 속성(attribute)은 파일에 관한 메타 정보로 Table 8.과 같다.

Table 8. File attribute

Flag	Contents
Hidden (H)	This file is not shown in the dir list, and is not affected by any file commands.
System (S)	This file is used by an operating system.
Read Only (R)	This file cannot be deleted nor modified.
Archive (A)	This file changed after the recent backup.

파일 속성을 변경하면 악성코드의 존재를 간단히 숨길 수 있어 자주 이용된다. 특히 USB로 전파하는 악성코드(Autorun, 악성코드 진단명)에서 많이 찾을 수 있다.

3.2.3.5 의심스러운 파일 이름

악성코드는 악성코드의 이름을 기본 프로세스 이름과 비슷하게 만든다. 그리고 사용자에게 정상 파일이 동작하는 것처럼 위장을 한다.

Table 9. The name of default processes

winlogon.exe, services.exe, csrss.exe, smss.exe, explorer.exe, iexplorer.exe, spoolsv.exe, lsass.exe, conime.exe, ctfmon.exe, userinit.exe

Table 9.의 기본 프로세스 이름들은 작업 관리자에서 자주 보이는 정상 프로세스들이다.

기본 프로세스 이름과 얼마만큼 동일한지 확인하기 위한 방법으로 문자열 매칭 알고리즘을 이용한다.

Table 10. String matching algorithm

(matched letters count / average length of two letters) * 100

Table 10.의 문자열 매칭 알고리즘에서 matched letters count는 리벤슈타인 거리(levenshtein distance)를 이용하여 구한다[20].

```
E:₩compare>compare.exe
source filename = svchost.exe
compare filename = svch0st.exe
Levenshtein distance algorithm = 1
E:₩compare>compare.exe
source filename = csrss.exe
compare filename = csrcs.exe
Levenshtein distance algorithm = 1
E:₩compare>compare.exe
source filename = explorer.exe
compare filename = exploit.exe
Levenshtein distance algorithm = 3
```

Fig.16. The result of computing levenshtein distance

리벤슈타인 거리를 이용하면 한 문자열에서 몇 번의 연산(삭제/추가/수정)으로 다른 문자열과 동일해지는지 측정할 수 있다.

Table 11. The experiment result of the correspondence rate

String 1	String 2	Correspondence Rate
svchost.exe	svch0st.exe	91%
csrss.exe	csrcs.exe	89%
explorer.exe	exploit.exe	75%

문자열 매칭 알고리즘을 이용하여 기본 프로세스 이름과 80% 이상으로 동일하다면 의심스러운 파일 이름으로 판단한다.

3.2.3.6 비정상적인 파일 경로

사용자는 시스템에서 어떤 프로세스가 실행 중인지 확인하기 위해 윈도우 디폴트 프로그램인 작업 관리자를 실행한다. 작업 관리자는 전체 경로가 아닌 프로세스 이름만을 보여주기 때문에 악성코드는 윈도우 시스템 파일들과 동일한 이름으로 경로를 바꾸어 사용자를 속일 수 있다.

로드되는 DLL의 검색 순서는 로드되는 프로그램의 폴더부터 시작하여 현재 폴더, 시스템 폴더, 윈도우 폴더, PATH 환경에 나온 폴더 순이다. 따라서 정상 파일(DLL)이 존재하는 위치보다 먼저 검색이 되는 위치에 동일 이름의 악성코드(DLL)가 존재한다면 정상 파일이 아닌 악성코드가 로드된다.

윈도우 시스템 파일과 동일 이름으로 다른 경로에 존재하면 비정상적인 파일 경로에 있다고 판단한다.

3.2.3.7 특이 API

악성코드 동작에 사용되는 주된 API로는 프로세스, 서비스, 네트워크, 소켓, 인터넷, 다운로드, 인젝션, 후킹, 파일 검색 및 쓰기 등이 있다. 실행파일의 IAT(Import Address Table), EAT(Export Address Table)에서 해당 API들을 검색할 수 있으나 패킹되어 있다면 검색에 어려움을 겪게 된다.

3.2.3.8 파일 시그니처

윈도우 운영체제는 파일의 종류를 구별하기 위해 파일 확장자를 사용한다. 사용자가 파일을 열면 파일 형식에서 확장자에 맞는 프로그램으로 연결한다.

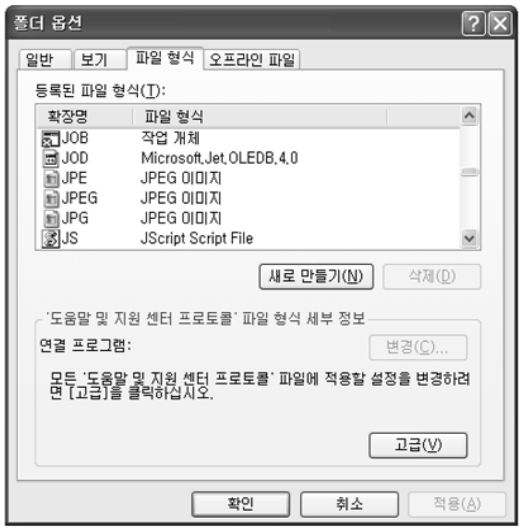

Fig.17. File types

그러나 Fig.18.과 같이 명령 프롬프트에서 실행파일은 확장자와 상관없이 실행할 수 있다.

Fig.18. The execution result of calculator with tmp extension

악성코드는 사용자를 속이기 위하여 의도적으로 실행파일의 확장자를 TMP, DAT, SCR 등으로 변경한다.

Table 12. File signature and file extension

File Signature (header)	File Extension
4D 5A	COM, DLL, EXE, PIF, SYS, ACM, OCX, CPL …
FF D8 FF E0 xx xx 4A 46 49 46 00	JFIF, JPE, JPEG, JPG
25 50 44 46	PDF, FDF
D0 CF 11 E0 A1 B1 1A E1	DOC, DOT, PPS, PPT, XLA, XLS, WIZ

파일 시그니처(file signature)는 파일의 앞부분(header) 또는 마지막 부분(footer)에 있는 특정한 형식의 고유 정보이다. Table 12.와 같이 파일 시그니처를 이용하면 파일 확장자와 상관없이 파일의 본래 형식을 확인할 수 있다.

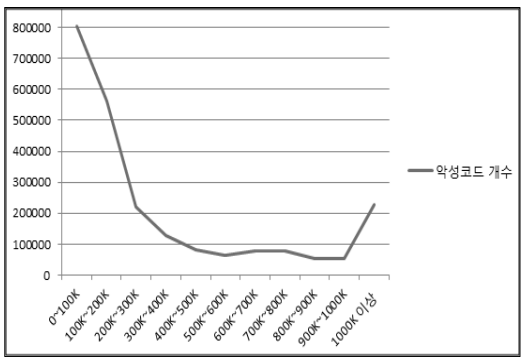

Fig.19. The size of malware

3.2.3.9 파일 크기

대부분의 응용 프로그램은 GUI(Graphical User Interface)를 포함하고 있어 파일 크기가 크다. 그러나 악성코드는 일반적으로 GUI를 사용하지 않기 때문에 파일 크기가 작다. Fig.19.에서 볼 수 있듯이 전체 악성코드 중에서 500KB 이하의 파일 크기를 갖는 악성코드는 76.3%에 해당한다.

3.2.4 레지스트리 정보

3.2.4.1 레지스트리의 Run, Shell

앞에서 설명한 Table 2., Table 3.의 레지스트리는 대부분 정당한 목적으로 응용 프로그램을 만드는 개발자에 의해 이용된다. 그러나 사용자와 상호작용 없이 자동으로 프로그램을 시작하며 지속성을 유지하기 위한 방법으로 악성코드에서도 이용하고 있다.

3.2.5 시간 정보

3.2.5.1 서비스 등록 시간

악성코드는 로그인한 사용자와 상관없이 백그라운드로 동작하고 부팅 시 자동실행으로 지속성을 유지하기 위해 윈도우 서비스를 악용한다. 윈도우 서비스는 윈도우 운영체제 설치 시점에 대부분 등록되기 때문에 최근에 등록된 윈도우 서비스는 추가 분석이 필요하다.

윈도우 서비스가 등록되면 레지스트리의 HKLM\SYSTEM\CurrentControlSet\Enum\Root\LEGACY_〈서비스 이름〉 키가 추가적으로 생성된다. HKLM\System\CurrentControlSet\Services\〈서비스 이름〉 키와는 달리 LEGACY_〈서비스 이름〉 키는 부팅 과정에서 LastWriteTime 정보가 변경되지 않는다. 따라서 윈도우 서비스에 등록된 최초 시간을 확인할 수 있다.

3.3 판단 항목의 점수

앞에서 설명한 Table 5.의 판단 항목들이 악성코드 탐지에 얼마만큼의 영향을 미치는지 실험을 통해 확인한다. 그리고 이를 반영하기 위해 각각의 판단 항목에 점수를 부여한다.

이번 실험은 Windows XP SP2 운영체제 환경에서 악성코드 151개와 정상 파일 158개로 다음과 같이 진행하였다.

Table 13. The standards of malware detection

Standards	Contents
Catalog	The hash value of Windows system files is different with the one of files in the catalog.
Digital Signature	No digital signature of the target file is included.
Loaded PE	The target file is in the loaded PE information.
Network Connection	The target file is in the network connection information.
DLL Injection	The DLL file loaded in the Winlogon, Explorer Process has different hash value with the one in the catalog, and is loaded into other processes.
Malware Path	The target file is located in the path in Table 6. and Table 7.
Packing	The WRITE property of the entry point section of the executable file is included.
File Properties	No file properties of the target file are included.
File Attribute	More than two attributes are included in file attributes of the target file.
Suspicious File Name	A name of the target file is same as the name of default process with more than 80%.
Abnormal File Path	The file name is same with one of Windows system files and locates in different path.
File Signature	The file signature of the target file is different with the file extension.
File Size	The size of the target file is Less than 500 KB.
Malicious API	The target file includes the API that is used for malicious behavior.
Run / Shell	The target file is registered in the registry in Table 2 and Table 3.
Service Time	LastWriteTime of LEGACY_〈SERVICE NAME〉 key in the registry is within three days from the analysis date.

일상적으로 사용되는 정상 파일들을 실험에 반영하고자 기본 프로그램(FTP), 안티바이러스 제품, 보안 프로그램(은행 사이트 접근에 따라 설치되는 프로그램)을 설치하고 실행한다.

또한 악성코드의 동향을 반영하고자 Table 14.와 같이 안랩의 ASEC Report vol42[21]에 수록된 악성코드 유형별 분포를 적용한다. 분포 결과에 비례하여 해당 진단명을 가진 악성코드를 실행한다.

3.1절에서 수집한 정보를 분석 대상으로 하고 3.2절에서 설명한 판단 항목들에 대해 Table 13.의 판단 기준을 적용하여 실험 결과를 얻는다.

Table 14. The distribution of malware types

Malware Type	Percentage	Sample Count
Trojan	53.1%	106
Worm	7.8%	16
Script	4.5%	0
Adware	4.0%	8
Virus	3.8%	8
Dropper	3.1%	6
Downloader	1.7%	4
Exploit	1.7%	0
Spyware	0.7%	2
Appcare	0.2%	1

정상 파일보다 악성코드에서 높은 확률로 존재할 것이라 추측하였던 판단 항목들에 대해 실험 결과를 통해 확인할 수 있다. Fig.20.은 실험한 악성코드와 정상 파일이 판단 항목들에 얼마만큼의 확률로 존재하는지 보여준다.

판단 항목에 따른 악성코드와 정상 파일의 확률을 Table 15.를 통해 비교할 수 있다.

Table 15. The probabilities according to standards

Standards	Malware	Normal files
Service Time	11.3%	1.3%
Run, Shell	10.6%	1.3%
Malicious API	13.2%	0.6%
File Size	84.1%	77.8%
File Signature	6.6%	10.1%
Abnormal File Path	2.6%	0.6%
Suspicious File Name	6.6%	0.0%
File Attribute	17.2%	3.8%
File Properties	71.5%	16.5%
Packing	45.7%	2.5%
Malware Path	66.2%	17.1%
DLL Injection	3.3%	0.0%
Network Connection	2.6%	1.9%
Loaded PE	35.1%	41.8%
Digital Signature	97.4%	46.2%
Catalog	9.9%	0.0%

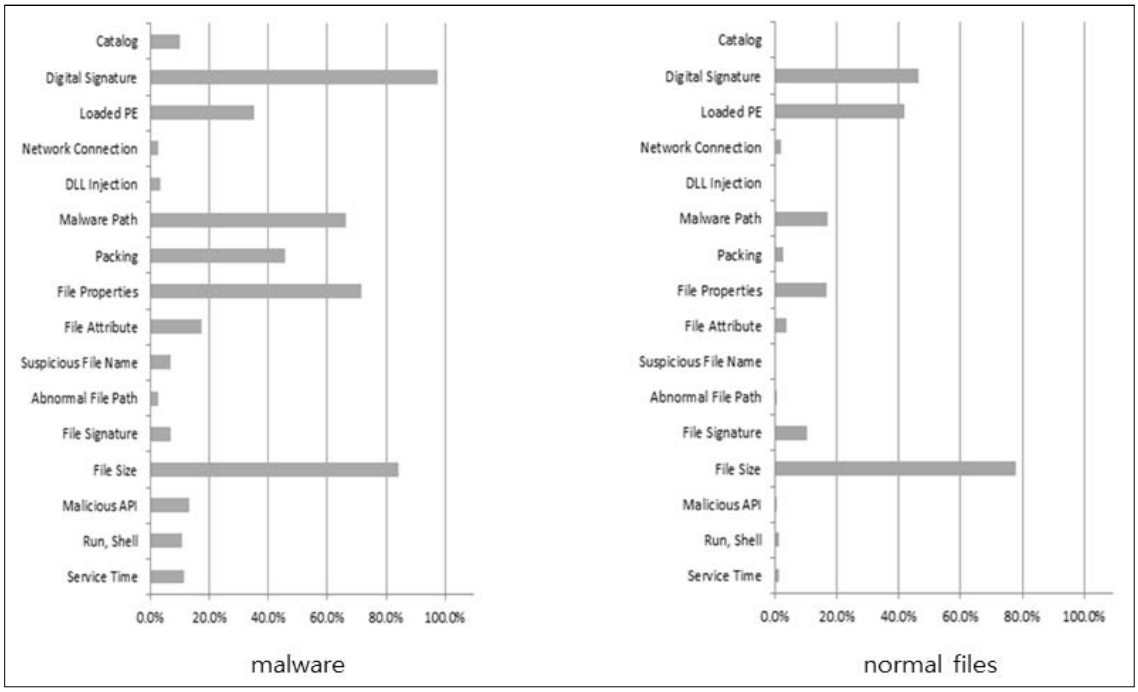

Fig.20. The probabilities in malware and normal files according to standards

대부분의 항목에서 악성코드가 정상 파일보다 높은 비율로 존재하나 File Signature와 Loaded PE 항목은 낮은 비율로 존재한다. 또한 정상 파일에는 존재하지 않고 악성코드에만 존재하는 Suspicious File Name, DLL Injection, Catalog 항목은 악성코드 판단에 중요한 특징이 된다.

Table 16. The score according to the ratio of standards

Standards	Ratio	Score
File Signature	-1.53%	-2
Loaded PE	-1.19%	-1
File Size	1.08%	1
Network Connection	1.37%	1
Digital Signature	2.11%	2
Malware Path	3.87%	4
Abnormal File Path	4.33%	4
File Properties	4.33%	4
File Attribute	4.53%	5
Run, Shell	8.15%	8
Service Time	8.69%	9
Packing	18.28%	18
Malicious API	22.0%	22
Suspicious File Name		22
DLL Injection		22
Catalog		22

각각의 판단 항목에서 악성코드의 확률을 정상 파일의 확률로 나눈다. 해당 비율을 근거로 Table 16.과 같이 점수를 부여한다.

-2와 -1의 점수를 가진 판단 항목은 악성코드보다는 정상 파일에서 더 많은 비율로 존재하기 때문에 악성코드 판단에서 제외한다. 또한 악성코드와 정상파일이 비슷한 비율로 존재하는 1의 점수를 가진 판단 항목도 악성코드 판단에 의미가 없어 제외한다. 그리고 정상 파일에는 존재하지 않고 악성코드에만 존재하는 판단 항목은 악성코드 판단에 큰 의미를 지니고 있어 가장 큰 점수와 동일하게 부여한다.

IV. 실험

본 연구에서는 Windows XP SP2 운영체제 환경에서 임의의 악성코드 210개와 정상 파일 207개로 실험을 진행하였다.

각각의 샘플을 실행하고 Table 13.의 판단 기준과 일치하는 경우 Table 16.의 판단 항목 점수를 부여한다. 이렇게 부여된 점수를 모두 합산하는 방법으로 결과를 도출한다.

이번 실험을 통해 Fig.21.과 같은 결과를 얻을 수 있었다.

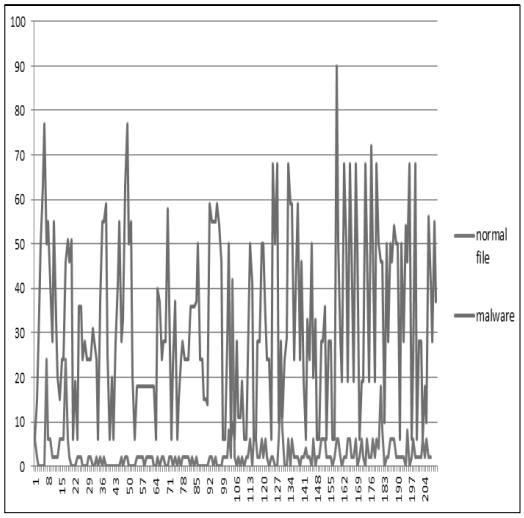

Fig.21. The final score of each sample

Fig.21.을 보면 악성코드에서 더 높은 합산 점수를 가지게 되어 정상 파일과 악성코드가 구분되는 것을 확인할 수 있다.

정상 파일이지만 높은 합산 점수를 가지게 되는 경우는 다음과 같았다.

가장 높은 합산 점수의 정상 파일은 Oracle 회사의 응용 프로그램인 orammc8.exe이다. 디지털 서명이 존재하지 않고 패킹되어 있으며 특이 API를 포함하고 있었다. 다음으로 높은 합산 점수를 받은 정상 파일은 MSI 회사의 BIOS 업데이트 프로그램인 6380v18.exe이다. 이 프로그램은 등록 정보가 존재하지 않고 패킹되어 있었다. 그리고 다음 정상 파일은 Google 회사의 SearchWithGoogleUpdate.exe이며 패킹되어 있었다. 위와 같이 패킹되어 있는 배포 프로그램과 업데이트 프로그램들이 정상 파일이지만 높은 합산 점수를 받게 되는 경우를 확인할 수 있다.

Table 17.은 실험 결과로서 기준 점수에 따른 탐지율과 오탐률이다.

Table 17. The detection rates and the false detection rates according to threshold

Threshold	Detection Count	Detection Rate	False Detection Count	False Detection Rate
0	210	100.0%	207	100.0%
2	210	100.0%	139	67.1%
4	210	100.0%	45	21.7%
6	210	100.0%	43	20.8%
8	180	85.7%	9	4.3%
10	180	85.7%	5	2.4%
11	176	83.8%	5	2.4%
14	173	82.4%	5	2.4%
15	172	81.9%	5	2.4%
18	167	79.5%	5	2.4%
19	156	74.3%	3	1.4%
20	146	69.5%	3	1.4%
22	142	67.6%	3	1.4%
23	142	67.6%	3	1.4%
24	141	67.1%	3	1.4%
25	119	56.7%	1	0.5%
26	118	56.2%	1	0.5%
28	118	56.2%	1	0.5%
29	94	44.8%	1	0.5%
31	93	44.3%	1	0.5%
33	92	43.8%	1	0.5%
35	87	41.4%	1	0.5%
36	87	41.4%	1	0.5%
37	80	38.1%	1	0.5%
40	75	35.7%	1	0.5%
41	74	35.2%	1	0.5%
42	70	33.3%	1	0.5%
44	70	33.3%	0	0.0%
46	70	33.3%	0	0.0%
48	61	29.0%	0	0.0%
50	61	29.0%	0	0.0%
51	39	18.6%	0	0.0%
53	37	17.6%	0	0.0%
54	37	17.6%	0	0.0%
55	34	16.2%	0	0.0%
56	24	11.4%	0	0.0%
58	23	11.0%	0	0.0%
59	22	10.5%	0	0.0%
63	16	7.6%	0	0.0%
68	14	6.7%	0	0.0%
72	4	1.9%	0	0.0%
77	3	1.4%	0	0.0%
90	1	0.5%	0	0.0%

Table 17.을 보면 기준 점수가 낮을 경우 탐지율과 오탐률은 모두 높고, 기준 점수가 높을 경우 탐지율과 오탐률은 모두 낮다. 그러나 기준 점수에 따라 탐지율과 오탐률의 차이가 크게 나타나 이를 알려지지 않은 악성코드 탐지에 이용한다.

F-measure는 실험의 정확도를 측정하기 위한 값으로 정밀도(precision)와 재현율(recall)의 조화평균을 뜻한다. F-measure 측정치가 높다는 것은 정밀도와 재현율 모두가 상당히 크다는 것을 보장한다.

Table 18. The definition of the classification context

	Classification result		
Answer		Positive	Negative
	Positive	tp (true positive)	fn (false negative)
	Negative	fp (false positive)	tn (true negative)

$$\mathrm{Precision}(P) = \frac{tp}{tp+fp} \quad (1)$$

$$Recall(R) = \frac{tp}{tp+fn} \quad (2)$$

$$F-measure(F) = 2^{*}\frac{\mathrm{Precision}^{*}\,Recall}{\mathrm{Precision}+Recall} \quad (3)$$

(1),(2),(3) 식의 평가 방법[22]을 이용하여 기준 점수 별 측정한 결과는 Table 19.에서 보여준다.

Table 19. Precision, Recall, and F-measure according to threshold

Threshold	tp	fp	fn	P	R	F
0	210	207	0	0.504	1.000	0.670
2	210	139	0	0.602	1.000	0.751
4	210	45	0	0.824	1.000	0.903
6	210	43	0	0.830	1.000	0.907
8	180	9	30	0.952	0.857	0.902
10	180	5	30	0.973	0.857	0.911
11	176	5	34	0.972	0.838	0.900
14	173	5	37	0.972	0.824	0.892
15	172	5	38	0.972	0.819	0.889
18	167	5	43	0.971	0.795	0.874
19	156	3	54	0.981	0.743	0.846
20	146	3	64	0.980	0.695	0.813
22	142	3	68	0.979	0.676	0.800
23	142	3	68	0.979	0.676	0.800
24	141	3	69	0.979	0.671	0.797
25	119	1	91	0.992	0.567	0.721
26	118	1	92	0.992	0.562	0.717
28	118	1	92	0.992	0.562	0.717
29	94	1	116	0.989	0.448	0.616
31	93	1	117	0.989	0.443	0.612
33	92	1	118	0.989	0.438	0.607
35	87	1	123	0.989	0.414	0.584
36	87	1	123	0.989	0.414	0.584
37	80	1	130	0.988	0.381	0.550
40	75	1	135	0.987	0.357	0.524
41	74	1	136	0.987	0.352	0.519
42	70	1	140	0.986	0.333	0.498
44	70	0	140	1.000	0.333	0.500
46	70	0	140	1.000	0.333	0.500
48	61	0	149	1.000	0.290	0.450
50	61	0	149	1.000	0.290	0.450
51	39	0	171	1.000	0.186	0.313
53	37	0	173	1.000	0.176	0.300
54	37	0	173	1.000	0.176	0.300
55	34	0	176	1.000	0.162	0.279
56	24	0	186	1.000	0.114	0.205
58	23	0	187	1.000	0.110	0.197
59	22	0	188	1.000	0.105	0.190
63	16	0	194	1.000	0.076	0.142
68	14	0	196	1.000	0.067	0.125
72	4	0	206	1.000	0.019	0.037
77	3	0	207	1.000	0.014	0.028
90	1	0	209	1.000	0.005	0.009

정밀도는 '악성코드 판단 분류'를 통해 악성코드로 탐지된 것 중 실제 악성코드의 비율을 뜻하고 재현율은 실제 악성코드들 중 '악성코드 판단 분류'를 통해 탐지된 악성코드의 비율을 뜻한다.

F-measure 측정치가 가장 높은 값은 0.911으로 정밀도(0.973)와 재현율(0.857) 모두가 상당히 크다. 이에 맞는 기준 점수는 10점이라는 것을 알 수 있다. Table 17.을 보면 기준 점수가 10점일 경우 탐지율 85.7%, 오탐률 2.4%가 된다.

이러한 실험의 결과로 높은 탐지율에 비해 낮은 오탐률을 원한다면 기준점수를 10점으로 설정할 것을 제안한다.

V. 결론

그동안 많은 연구기관들과 기업들은 시그니처 방식으로 탐지하는 안티바이러스 제품에서 벗어나 악성코드를 탐지하는 다양한 방법들을 제안하였다. 본 논문은 디지털 포렌식 기법을 통해 얻을 수 있는 정보들을 활용하여 알려지지 않은 악성코드 탐지 방법을 제시한다.

실행파일이 메모리에 로드되어 실행되면 운영체제는 여러 정보를 변경하고 흔적 정보를 남긴다. 이러한 흔적 정보에서 디지털 포렌식 기법을 활용하여 분석 대상이 되는 파일 정보를 수집한다. 또한 악성코드에 존재하는 주요 특징들을 악성코드 판단 항목으로 이용한다. 그러나 정상 파일과 악성코드 모두에서 해당 특징들을 찾아 볼 수 있기 때문에 실험을 통해 판단 항

목에 따른 악성코드와 정상 파일의 확률을 비교한다. 각각의 판단 항목에서 악성코드의 확률을 정상 파일의 확률로 나누어 해당 비율을 근거로 점수를 부여한다. 마지막으로 판단 기준에 일치하는 판단 항목의 점수를 모두 합산한다. 이에 따른 실험 결과로 악성코드와 정상 파일이 구분되는 것을 확인할 수 있다. 악성코드와 정상 파일을 구분할 수 있는 최적의 기준 점수를 설정하여 높은 탐지율에 비해 낮은 오탐률로 탐지한다.

본 논문에서 제안한 탐지 방법을 이용하면 높은 확률을 가지고 알려지지 않은 악성코드를 탐지할 수 있다. 또한 디지털 포렌식 기법을 활용하여 정보를 수집하기 때문에 조사 대상의 개수를 줄일 수 있다. 따라서 디스크의 모든 파일을 대상으로 하는 안티바이러스 제품에 비해 속도 향상을 기대할 수 있다.

보안 담당자에 의해 이상 징후가 탐지되고 좀비 컴퓨터로 의심되면 알려지지 않은 악성코드를 신속히 확보할 수 있는 초기 대응이 필요하다. 이런 경우 본논문에서 제안한 탐지 방법은 유용하게 쓰일 수 있다. 또한 침해사고를 분석할 때 분석가들은 과도한 데이터와 마주하게 되는데 이 중 알려지지 않은 악성코드는 침해사고 분석의 중요 정보로 제공된다. 따라서 신속한 침해사고 대응이 필요한 경우 제안한 탐지 방법을 활용할 수 있다.

References

〔1〕 Act on the promotion of information and communications network utilization and protection of information, http://www.law.go.kr/LSW/LsInfoP.do?lsiSeq=87471#0000

〔2〕 Karen Scarfone, Tim Grance, and Kelly Masone, "Computer security incident handling guide," NIST Special Publication, Mar. 2008.

〔3〕 Patrick Kral, "The incident handlers handbook," Dec. 2011.

〔4〕 Incident analysing process of KISA, https://www.kisa.or.kr/jsp/common/down.jsp?folder=uploadfile&filename=%EC%A0%9C2010-8%ED%98%B8-%EC%B9%A8%ED%95%B4%EC%82%AC%EA%B3%A0_%EB%B6%84%EC%84%9D_%EC%A0%88%EC%B0%A8(%EB%82%B4%EC%A7%80)%EC%B5%9C%EC%A2%85(fin).pdf

〔5〕 Incident handling process, https://isc.sans.edu/forums/diary/Cyber+Security+Awareness+Month+-+Day+18+-+What+you+should+tell+your+boss+when+there+s+a+crisis+/9760

〔6〕 Steven Alexander, "Finding malware on compromised windows machines," Usenix, Apr. 2005.

〔7〕 Richard Nolan, Colin O'Sullivan, Jake Branson, and Cal Waits, "First responders guide to computer forensics," dtic.mil, Mar. 2005.

〔8〕 Martin Overton, "Malware forensics: detecting the unknown," 2008 Virus Bulletin conference, Oct. 2008.

〔9〕 TaeGuen Kim, In-Kyoung Kim, and Eul Gyu Im, "Malware detection method via major block comparison," Journal of Security Engineering, 9(5), Oct. 2012.

〔10〕 Kyoung-Soo Han, In-Kyoung Kim, and Eul-Gyu Im, "Malware family classification method using API sequential characteristic," Journal of Security Engineering, 8(2), pp. 319-335, Apr. 2011.

〔11〕 Igor Santos, Yoseba K. Penya, Jaime Devesa, and Pablo G. Bringas, "N-grams-based file signatures for malware detection," ICEIS (2), 2009.

〔12〕 ClemensKolbitsch, PaoloMilaniComparetti, ChristopherKruegel, EnginKirda, XiaoyongZhou, and XiaoFengWang, "Effective and efficient malware detection at the end host," The 18th USENIX Security Symposium, 2009.

〔13〕 Min-ho Kim, Minsoo Kim, and Bong-nam Noh, "The framework for malware analysis using statistical information of registry," journal of korean institute of information technology, 10(9), pp. 97-104, Sept. 2012.

〔14〕 Seong-Bin Park, Min-Soo Kim, and

Bong-Nam Noh, "Detection method using common features of malware variants generated by automated tools," journal of korean institute of information technology, 10(9), pp. 67-75, Sept. 2012.
〔15〕 Yong-Wook Chung and Bong-Nam Noh, "Selecting features for measuring similarity between attack toolkits and polymorphic codes," Journal of Security Engineering, 9(1), Feb. 2012.
〔16〕 Harlan Carvey, Windows forensic analysis DVD toolkit, Second Edition, 2009.
〔17〕 Harlan Carvey, Windows registry forensics: advanced digital forensic analysis of the windows registry, 2011.
〔18〕 Joachim Metz, "MSIE Cache File (index.dat) format specification: analysis of the index.dat file format," 2009.
〔19〕 Seungwon Han and Sangjin Lee, "Packed PE file detection for malware forensics," The KIPS Transactions : Part C, 16(5) pp. 555-562, Oct. 2009.
〔20〕 Levenshtein Distance, http://en.wikipedia.org/wiki/Levenshtein_distance
〔21〕 AhnLab's ASEC Report. vol42, http://download.ahnlab.com/asecReport/ASEC_Report_Vol.42_Kor.pdf
〔22〕 Precision and recall, http://en.wikipedia.org/wiki/Precision_and_recall

〈저자소개〉

이 재 호 (Jaeho Lee) 정회원
2003년 2월: 한서대학교 컴퓨터정보학과 학사
2014년 2월: 고려대학교 정보보호대학원 석사
2003년 12월~2005년 3월: 시큐브레인 보안기술분석팀
2005년 4월~현재: 안랩 클라우드분석팀
〈관심분야〉 디지털 포렌식, 악성코드 탐지 및 분석

이 상 진 (Sangjin Lee) 종신회원
1987년 2월: 고려대학교 수학과 학사
1989년 2월: 고려대학교 수학과 석사
1994년 8월: 고려대학교 수학과 박사
1989년 10월~1999년 2월: 한국전자통신연구원 선임 연구원
1999년 3월~2001년 8월: 고려대학교 자연과학대학 조교수
2001년 9월~현재: 고려대학교 정보보호대학원 교수
2008년 3월~현재: 고려대학교 디지털포렌식연구센터 센터장
〈관심분야〉 대칭키 암호, 정보은닉이론, 디지털 포렌식