DOI: 10.3745/KIPSTC.2009.16C.5.555

악성코드 포렌식을 위한 패킹 파일 탐지에 관한 연구

한 승 원[†] · 이 상 진[††]

요 약

악성코드 사고 조사에서 가장 중요한 것은 신속하게 악성코드를 탐지하고 수집하는 것이다. 기존의 조사 방법은 시그니처 기반의 안티바이러스 소프트웨어를 이용하는 것이다. 시그니처 기반의 탐지는 실행파일 패킹, 암호화 등을 통해 쉽게 탐지를 회피할 수 있다. 그렇기 때문에 악성코드 조사에서 패킹을 탐지하는 것도 중요한 일이다. 패킹탐지는 패킹 시그니처 기반과 엔트로피 기반의 탐지 방법이 있다. 패킹 시그니처 기반의 탐지는 새로운 패킹을 탐지하지 못하는 문제가 있다. 그리고 엔트로피 기반의 탐지 방법은 오탐의 문제가 존재한다. 본 논문에서는 진입점 섹션의 엔트로피 통계와 패킹의 필수적인 특징인 'write' 속성을 이용하여 패킹을 탐지하는 기법을 제시한다. 그리고 패킹 PE 파일을 탐지하는 도구를 구현하고 도구의 성능을 평가한다.

키워드 : 악성코드 포렌식, PE 파일 분석, 엔트로피, 패킹 탐지

Packed PE File Detection for Malware Forensics

Seungwon Han[†] · Sangjin Lee[††]

ABSTRACT

In malware accident investigation, the most important thing is detection of malicious code. Signature based anti-virus softwares have been used in most of the accident. Malware can easily avoid signature based detection by using packing or encryption method. Because of this, packed file detection is also important. Detection methods can be divided into signature based detection and entropy based detection. Signature based detection can not detect new packing. And entropy based detection has a problem with false positive. We provides detection method using entropy statistics of entry point section and 'write' properties of essential characteristic of packed file. And then, we show packing detection tool and evaluate its performance.

Keywords : Malware Forensics, PE File Analysis, Entropy, Packing Detection

1. 서 론

악성코드 사고 조사에서 악성코드 탐지는 매우 중요한 과제이다. 일반적으로 악성코드 탐지는 안티바이러스 소프트웨어를 사용하는데 이 소프트웨어는 시그니처(Signature)를 기반으로 악성코드를 탐지한다. 시그니처 기반의 악성코드 탐지 방법은 빠르고 효과적이지만, 탐지 규칙이 많아지면 속도가 느려지고, 해당 시그니처가 없는 경우 탐지하지 못하는 문제가 있다[1]. 악성코드 탐지에 실패하게 되면 정확한 사고 조사를 할 수 없다.

악성코드 제작자는 실행 파일 패킹 및 암호화 기법을 사용한다[2-3]. 이를 통해서 시그니처 기반의 악성코드 탐지를 회피한다. 안티 바이러스 및 정보보호 관련 연구 기관인 AV-Test사에 따르면 악성코드의 92% 이상이 패킹을 사용하고 있다[4]. 그러므로 효율적인 악성코드 사고 조사를 위해서는 패킹 파일을 탐지하고 탐지된 파일 중에서 악성 여부를 판단할 필요가 있다.

패킹 PE 파일 탐지 방법은 크게 시그니처 기반의 탐지와 엔트로피 기반의 탐지로 나눌 수 있다. 시그니처 기반의 탐지는 패킹 파일의 특성을 추출하고 이 룰을 바탕으로 탐지한다. 이 방법은 새로운 패커를 탐지하지 못하는 문제가 있다. 엔트로피 기반의 탐지는 데이터의 빈도를 기반으로 하는데 데이터 영역의 선택에 따라 오탐이 발생할 수 있다.

본 논문에서는 대부분의 악성코드가 탐지와 분석을 회피하기 위해 패킹기법을 사용한다는 것에 착안하여, 패킹 PE 파일을 판단하는 것에 초점을 맞추고 있다. 패킹 파일의 탐지를 위해 PE 파일의 진입점 섹션 엔트로피를 사용하였고 오탐을 줄이기 위해 패킹된 PE 파일의 필수적인 특징을 추가하였다. 이를 바탕으로 탐지 도구 'REMINDer (REsponse tool for Malware INDication)'를 구현하였다. 공개 도구와 구현한 도구의 탐지율 및 시간을 측정하여 성능을 평가하였다.

※ 본 연구는 한국과학재단을 통해 교육과학기술부의 바이오연구개발사업으로부터 지원받아 수행되었습니다(M10640030004-08N4003-00410).
† 준 회 원 : 고려대학교 정보경영공학전문대학원 석사과정
†† 종신회원 : 고려대학교 정보경영공학전문대학원 정교수
논문접수 : 2009년 6월 29일
수 정 일 : 1차 2009년 8월 7일
심사완료 : 2009년 8월 26일

2장에서는 패킹 파일 탐지 관련 연구와 패킹 및 악성 파일 탐지 도구에 대한 조사를 하였고, 3장에서는 PE 파일과 패킹된 PE에 대해서 설명하고, 패킹 탐지를 위한 엔트로피와 오탐을 줄이기 위한 패킹의 필수 특징에 대하여 제시하였다. 4장에서는 이전 실험을 바탕으로 탐지 방안을 제시하였다. 5장에서는 도구 구현 및 실험을 통해 탐지 성능을 비교하였고, 마지막으로 6장에서는 본 연구의 결론 및 향후 연구에 대해 기술하였다.

2. 관련 연구

2.1 패킹 탐지 연구

Lyda[5]는 엔트로피 분석을 기반으로 암호화되거나 패킹된 악성코드를 탐지하는 기술에 대해 설명하였다. 분석가들이 평문과 실행파일, 패킹, 암호화 된 파일을 구분 기준으로 사용할 수 있는 패킹 혹은 암호화된 실행파일의 엔트로피 메트릭스 셋을 제시하였다. 엔트로피 계산은 바이트 자체의 의미와 상관없이 실행파일에 포함된 바이트 발생 빈도를 이용한다. 저자는 실행 코드의 바이트 통계 분포 조사를 통해 빠르고 효과적으로 패킹된 파일을 확인할 수 있다고 말하고 있다. Lyda는 전체파일에 대한 엔트로피 계산을 통해 패킹과 일반 파일을 구분하였는데 패킹되지 않는 파일에서 엔트로피가 높게 나타나는 오탐이 발생할 수 있다. 그리고 패킹된 섹션이 작을 경우 엔트로피가 낮게 나타나 미탐이 발생할 여지가 있다.

Choi[6]는 PE 파일의 헤더 분석을 기반으로 패킹 파일을 탐지하는 도구 'PHAD'를 제안했다. 악성코드가 패킹되거나 암호화된 경우에 패킹 및 암호화된 실행 코드는 먼저 언패킹(Unpacking)을 해야 하는데, 그 전 단계로 패킹된 실행파일을 탐지해야 한다고 설명하고 있다. 패킹된 파일의 PE 헤더에는 일반적이지 않은 특징 요소가 있으며 이중 8개를 CV(Characteristic Vector)로 정의하여, 유클리디안 거리(Euclidean distance)를 계산함으로써 패킹되지 않은 파일과 구별하였다. Choi가 제안하는 방법은 몇몇 패커들이 사용하는 특징을 추출하고 그 특징을 바탕으로 탐지하는 기법을 사용하였기 때문에 분석된 패킹에 대해서는 탐지가 가능하지만 그 외의 새로운 패킹에 대해서는 탐지하지 못할 가능성이 있다. 본 논문에서는 패킹의 필수적인 요소를 실험을 통해 추출하였고 엔트로피를 계산을 통해 패킹 탐지율을 높였다. 이를 통해 변형된 패킹의 탐지도 가능하다.

2.2 공개 도구

PEID[7]는 PE 파일의 패커 및 컴파일러를 탐지하는 가장 일반적인 공개 도구이다. 현재 PEID는 600개 이상의 시그니쳐를 기반으로 PE 파일의 패커 및 컴파일러를 탐지할 수 있다. 추가 기능으로 OEP Finder, Krypto Analyzer, Generic Unpacker를 제공하고 있으며 디렉터리를 입력 받아 동시에 여러 파일을 검사하는 기능도 제공하고 있다. 그러나 PEID는 시그니쳐가 없는 경우 탐지하지 못하는 문제점이 있다.

MRC(MANDIANT Red Curtain)[8]는 시스템의 침해사고 대응을 위한 공개 도구이다. 이 도구는 시스템의 침해 여부를 빠르고 효과적으로 조사하기 위한 목적으로 설계되었다. PE 파일의 구조 분석을 통해 정상 파일인지 여부를 판단한다. 그리고 암호화 또는 압축 등의 회피 기법이 적용된 데이터의 경우 PE 파일의 엔트로피가 구조화된 데이터와 비교해 상대적으로 높다는 점을 이용해 가중치를 주는 방식이다. 침해 여부의 결과는 위협 스코어를 계산하여 조사할 필요가 있는 파일을 Red로 표현하고 의심스러운 것은 Yellow, 정상적인 PE 파일은 Green으로 나타낸다. 또한 PE 파일의 시그니처를 기반으로 패킹 여부를 판단한다.

본 논문에서 PEID, MRC와 REMINDer의 패킹 탐지율 및 소요 시간을 비교하여 각각의 성능을 평가한다.

3. PE 파일 포멧과 패킹된 PE 파일 포맷의 차이점

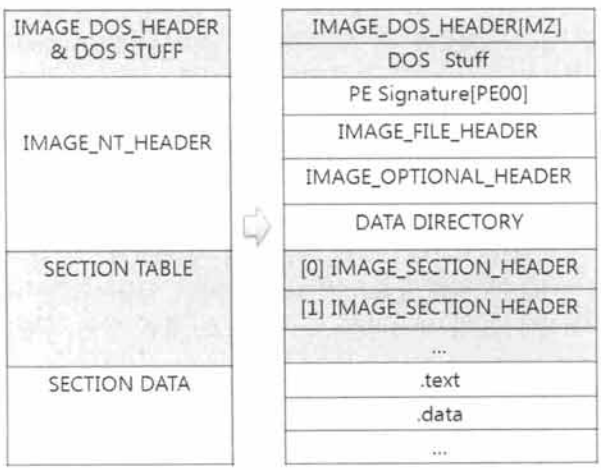
본 장에서는 PE 파일의 포멧과 패킹된 PE 파일의 포맷에 대해서 알아본다. 패킹된 PE 파일의 탐지를 위해 엔트로피 실험 결과를 제시하고 오탐을 줄이기 위해 패킹 PE 파일의 필수적인 특징을 추출한다.

3.1 PE 파일 포멧

PE(Portable Executable)포맷은 32비트나 64비트 버전 마이크로소프트 윈도우에서 사용되는 실행 파일의 파일 포맷을 말한다. 이를 지원하는 확장자로는 .exe, .obj, .dll, .sys 등이 있다[9].

실행파일 구조는 (그림 1)과 같이 DOS Header와 MS-DOS Stub으로 시작한다. MS-DOS Stub는 과거 도스와의 호환을 위한 DOS HEADER와 도스 환경에서 파일이 실행될 경우에 사용될 도스 실행 이미지가 들어있다.

IMAGE_NT_HEADER는 PE 고유의 식별자로 시작하며 실행될 수 있는 시스템, 섹션 수, 시간 값, 실행 속성 등이 포함되어 있다. 코드, 데이터, 리소스, 디버그 정보와 같은



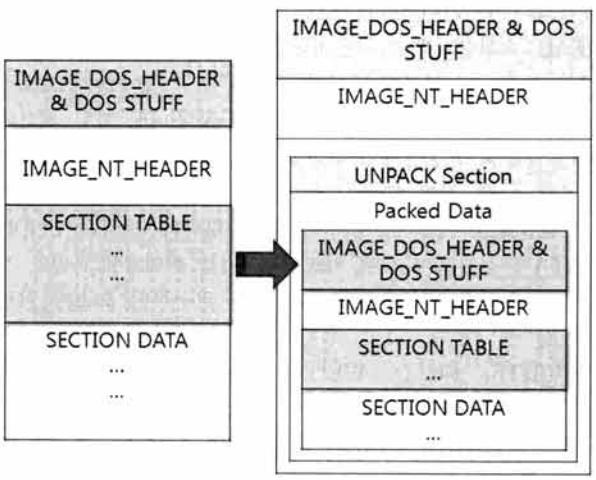
(그림 1) PE 파일 포맷

서로 다른 데이터 속성을 가지는 데이터들로 분리, 관리되는 SECTION 테이블이 존재한다. 그리고 실제 실행되는 실행코드와 데이터 혹은 리소스와 같은 여러 가지 실제적인 데이터가 포함된 SECTION DATA로 구성되어 있다.

3.2 패킹된 PE 파일 포멧

패킹이란 실행 파일을 압축하는 것을 의미하며, 실행되면서 압축이 풀리고 실제 파일이 실행되게 된다. 그래서 패킹을 '실행압축'이라고도 한다. 패킹의 상업용 소프트웨어의 코드를 크랙으로부터 보호하려는 목적으로 개발되었으나 악성코드 제작자들이 자신이 제작한 코드를 압축함으로써 분석을 어렵게 하고, 사이즈를 줄임으로써 네트워크상에서 보다 빠르게 전파시키려는 목적으로 이용하고 있다[10].

패킹을 하면 (그림 2)와 같이 본래의 코드는 실행 압축되어 감춰지기 때문에 언패킹 과정을 거쳐야 바이너리 분석이 가능하게 된다. 만일 패킹을 벗겨내지 못한다면 분석이 어려워진다. 이러한 이점 때문에 대부분의 악성코드 제작자들은 패킹기법을 사용한다.



(그림 2) 패킹된 PE 파일 포맷

3.3 패킹 PE 파일의 엔트로피

패킹 PE 파일을 실행 압축이라고 표현하는 것에서 알 수 있듯이 패킹 PE 파일은 데이터를 압축 및 암호화하여 저장한다. 그렇기 때문에 패킹된 섹션의 데이터는 랜덤하다. 이런 특징은 Byte Entropy 측정을 통해 알 수 있다. 엔트로피는 데이터 스트림의 랜덤성을 측정하는데 사용된다. 엔트로피를 계산하여 패킹 및 암호화 되었는지 여부를 판단할 수 있다. 엔트로피 결과 값의 범위는 0에서 8까지 나타난다.

이전 연구에서 Lyda는 패킹 및 암호화된 악성코드를 찾기 위한 방법으로 Bintropy를 사용하였고, 엔트로피 메트릭스 셋을 제시하였다. 데이터 셋으로 일반 텍스트 파일, 실행파일, 패킹된 실행 파일, 암호화된 실행파일을 선택했고, 실행 파일의 구분을 없애기 위해서 512 bytes 블록 단위로 엔트로피를 계산하였다. 실행파일의 경우 많은 블록에 0이 나타나고 그로 인해 엔트로피가 급격히 낮아지는 문제가 있어 이를 보완하기 위해 1/2 이상 0이 포함된 블록은 제외하였다. Lyda의 엔트로피 메트릭스 셋은 <표 1>과 같다.

<표 1>과 같이 텍스트 파일과 실행파일, 패킹된 실행 파일, 암호화된 실행 파일을 구분할 수 있다고 설명하고 있다.

통계 처리에서 의미가 있기 위해서는 가능한 모든 데이터의 발생 빈도가 최소한 5 이상이어야 한다. 따라서 바이트 빈도수에 대한 엔트로피를 측정하려면 최소한 5*256 바이트 이상의 데이터가 필요하다. 즉 1,280 bytes 크기 이상의 데이터를 가지고 실험해야 그 신뢰성을 보장할 수 있다. 그런데 Lyda는 이를 엄격하게 다루지 않고 Bintropy라는 통계 패키지만을 이용하였으며, 데이터의 크기, 데이터 처리 방법 등을 명확하게 언급하고 있지 않아 현실적인 적용에 어려움이 있다.

본 논문에서는 통계 처리의 신뢰성을 보장하기 위해 1,280 bytes 이상의 데이터에 대한 엔트로피를 계산하였다. 그리고 일반 PE 파일과 패킹된 PE 파일을 구분할 수 있는 기준점을 확인하기 위해 두 가지 형태로 엔트로피를 계산하였다. 첫 번째는 파일 전체의 엔트로피를 측정하는 것이다. 패킹된 PE 파일의 경우에 압축 및 암호화를 하기 때문에 전체 파일에서의 엔트로피가 높게 나타날 것이기 때문이다. 두 번째는 진입점(Entry Point) 섹션의 엔트로피 값이다. 일반 PE 파일은 진입점 섹션에 프로그램의 주요 코드가 존재한다. 패킹된 PE 파일도 마찬가지로 진입점 섹션에 패킹을 해제하는 코드와 패킹 데이터가 있다. 엔트로피 실험에 사용된 섹션 데이터의 평균값은 118 kbytes 이다. 엔트로피 계산에는 C. Shannon이 제안한 정보 엔트로피(information entropy)[11] 계산식을 사용하였다. 정보 엔트로피는 신호나 사건에 있는 정보의 양을 엔트로피의 개념을 빌려 설명한 것이다.

$$H(X) = \sum_{i=1}^{n} P(i)\log_2(\frac{1}{P(i)})$$
$$= -\sum_{i=1}^{n} P(i)\log_2 P(i)$$

실험 데이터로 일반 PE 파일을 'Windows', 'Program Files' 디렉터리에서 400개를 추출하였고, 패킹된 파일 400개를 Honeypot에서 수집하였다. 그 중에서 각각 200개를 표본

〈표 1〉 Lyda의 Entropy Metrics

Data Set	Average Entropy	99.99% Confidence Intervals (Low to High)
Plain text	4.715	4.401 - 5.030
Native executables	6.227	6.084 - 6.369
Packed executables	7.233	7.199 - 7.267
Encrypted executables	7.303	7.295 - 7.312

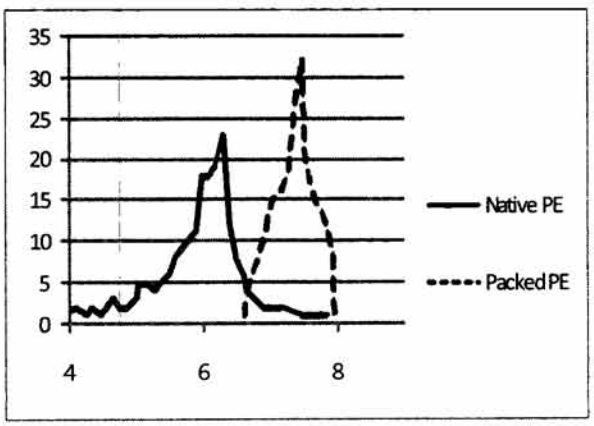
데이터로 사용하여 엔트로피를 측정하였고 계산 결과는 다음과 같다.

<표 2>를 그래프로 표현하면 (그림 3)과 (그림 4)와 같다. x축은 엔트로피이고 y축은 엔트로피의 개수를 나타낸다.

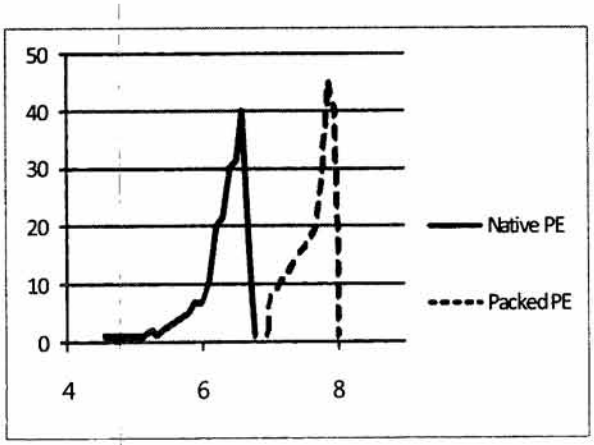
그림에서 보는 것과 같이 (그림 3) 전체 파일 영역의 엔트로피는 겹치는 부분이 생긴다. 그렇기 때문에 전체 파일에 대한 엔트로피는 패킹을 탐지하기 위한 기준으로 적합하지 않다. 그러나 (그림 4) 진입점 섹션의 경우 일반 PE 파일과 패킹된 PE 파일의 엔트로피 값이 확연히 구분되는 것을 알 수 있다. 즉 진입점 섹션의 엔트로피를 통해 일반 PE 파일과 패킹된 PE 파일을 구분할 수 있다. 본 논문에서는 패킹 탐지를 위한 요소로 진입점 섹션의 엔트로피 값을 사용한다.

〈표 2〉 엔트로피 결과

Data Area	DATA SET	Average Entropy	99% Confidence Intervals (Low to High)
Entry Point Session	Native PE File	6.18	5.05~6.69
	Packed PE File	7.68	6.85~7.99
Total Area	Native PE File	5.86	3.96~7.46
	Packed PE File	7.33	6.63~7.92



(그림 3) 전체 파일의 엔트로피 분포



(그림 4) 진입점 섹션의 엔트로피 분포

3.4 패킹 PE 파일의 필수 요소 추출

Choi의 연구에서는 패킹 PE 파일의 특징 벡터를 8개 추출하고 뉴클리디안 거리를 측정함으로써 패킹된 PE 파일 탐지를 제시했는데 본 논문에서는 여러 개의 특징 벡터를 포함하는 필수 요소를 추출하는데 초점을 맞추고 있다.

일반적인 PE 파일의 헤더에서 PE 파일의 특성을 나타내는 여러 가지 정보가 있는데 그중에서 가장 주목할 만한 것은 SECTION TABLE의 IMAGE_SECTION_HEADER의 Characteristics 속성 값 4bytes이다. 이 속성은 섹션의 실행 권한에 따라 EXECUTE, READ, WRITE를 첫 번째 1byte로 표현하고, 데이터 타입에 따라 CODE인지 DATA인지를 마지막 1byte로 표현한다. Characteristics 속성을 나타내는 속성 값은 다음 <표 3>과 같다.

일반적인 PE 파일의 경우 실행 섹션의 속성은 4bytes로 표현되며 '0x60000020'의 값을 가진다. 이것은 EXECUTE, READ 속성을 가지고, CODE 데이터라는 것을 나타낸다. 이와 같은 속성이 PE 파일의 실행을 위한 필수적인 요소라고 생각할 수 있는데 실제는 그렇지 않다. PE 파일 섹션의 모든 속성을 변경시켜 실험하였다.

<표 4>와 같이 Characteristics의 값은 EXECUTE 권한에 상관없이 모두 실행 가능하고, 데이터 타입은 CODE, DATA에 상관없이 실행 가능하다. 이를 통해 PE 파일을 실행하기 위해서 EXECUTE, CODE가 필수적인 요소가 아님을 알 수 있다. 다시 말해 PE 파일의 실행을 위해서 EXECUTE, READ, WRITE 속성뿐만 아니라 CODE, DATA 속성은 얼마든지 변경이 가능하다.

위의 실험과 같이 EXECUTE 속성이 아닌 PE 파일이 실행이 가능한 이유는 Windows의 DEP(데이타 실행 방지, Data Execution Prevention)[12]가 활성화되어 있지 않기 때문이다. Windows의 DEP라는 것은 악의적인 실행 코드가 시스템 메모리의 특정 영역에서 실행되는 것을 방지하기 위

〈표 3〉 Section Characteristics

Value	Description
20 00 00 00	IMAGE_SCN_MEM_EXECUTE (실행될 수 있는 섹션)
40 00 00 00	IMAGE_SCN_MEM_READ (읽기 가능 영역 섹션)
80 00 00 00	IMAGE_SCN_MEM_WRITE (쓰기 가능영역 섹션)
00 00 00 20	IMAGE_SCN_CTN_CODE (코드로 채워진 섹션)
00 00 00 40	IMAGE_SCN_CTN_INITIALIZED_DATA(초기화된 데이터 섹션)
00 00 00 80	IMAGE_SCN_CTN_UNINITIALIZED_DATA(비초기화된 데이터 섹션)

〈표 4〉 일반 PE 파일의 Characteristics에 따른 실행가능 여부

Characteristics	속 성	실행 가능
20 00 00 *0	EXECUTE	Yes
40 00 00 *0	READ	Yes
80 00 00 *0	WRITE	Yes
60 00 00 *0	EXECUTE, READ	Yes
A0 00 00 *0	EXECUTE, WRITE	Yes
C0 00 00 *0	READ, WRITE	Yes
E0 00 00 *0	EXECUTE, READ, WRITE	Yes

해서 이를 추가적으로 확인하는 하드웨어 및 소프트웨어적인 기술을 말한다. DEP는 메모리 위치가 명시적으로 실행 코드를 포함하지 않는 경우 프로세스의 모든 메모리 위치를 비실행으로 표시하는 방법으로 악의적인 실행 코드가 실행되는 것을 방지한다.

일반적인 PE 파일과 달리 패킹된 PE 파일의 경우, 패킹된 PE 파일의 섹션은 PE 파일이 실행되면서 패킹을 해제할 코드가 필요하다. 그리고 해제된 데이터를 쓰는 권한이 필요하다. 그래서 패킹 파일은 섹션의 Characteristics 속성에 READ, EXECUTE 뿐만 아니라 WRITE 권한이 포함된다. 그 결과 대부분의 패킹된 PE 파일의 Characteristics의 값은 '0xE00000*0'으로 나타난다. PE 파일과 마찬가지로 패킹된 PE 파일에서 위의 세 가지가 필수적인 요소인지 속성 값을 변경하여 실험하였고 결과는 다음 표와 같다.

<표 5>에서 보는 것과 같이 패킹 PE 파일 섹션의 Characteristics 속성 중에 WRITE이 포함된 경우에만 실행이 가능하다. 이를 바탕으로 섹션의 Characteristics에 WRITE 속성은 패킹된 PE 파일의 필수적인 요소임을 알 수 있다.

〈표 5〉 패킹 PE 파일의 Characteristics에 따른 실행가능 여부

Characteristics	속 성	실행 가능
20 00 00 *0	EXECUTE	No
40 00 00 *0	READ	No
80 00 00 *0	WRITE	Yes
60 00 00 *0	EXECUTE, READ	No
A0 00 00 *0	EXECUTE, WRITE	Yes
C0 00 00 *0	READ, WRITE	Yes
E0 00 00 *0	EXECUTE, READ, WRITE	Yes

4. 패킹 PE 파일 탐지 방안

3장에서 일반 PE 파일과 패킹된 PE 파일을 구분할 수 있는 2가지 특징이 존재한다는 것을 실험을 통해 설명하였다. 이를 바탕으로 2가지 특성을 이용한 실험을 통해 효율적인 탐지 방안 및 순서도를 제시한다.

4.1 패킹 PE 파일 탐지

4.1.1 엔트로피 기반의 탐지

패킹된 파일은 압축 및 암호화 되어 있기 때문에 데이터의 엔트로피가 높다. 이전 장에서 실험을 통해 확인한 바와 같이 진입점 세션의 엔트로피를 통해 패킹 PE와 일반 PE를 판단할 수 있다. 패킹된 PE 파일의 진입점 섹션의 엔트로피 범위가 6.85-7.99이고 일반 PE 파일의 진입점 섹션의 엔트로피 범위가 4.55-6.82이므로 진입점 섹션의 엔트로피가 6.85 이상인 것을 패킹된 PE 파일로 판단한다. 엔트로피 계산만을 이용하여 패킹 탐지가 가능한지 실험을 통해 확인하였다. 실험 데이터로는 Windows 디렉터리의 파일 200개와 패킹 PE 파일 200개를 대상으로 하였다. <표 6>은 진입점 섹션의 엔트로피를 이용하여 패킹 탐지율을 확인한 결과이다.

위 결과와 같이 일반 PE 파일에서도 엔트로피가 높은 경우가 있기 때문에 오탐(False Positive)이 생긴다. 그리고 패킹 파일이지만 엔트로피가 낮아 패킹이라고 탐지하지 못하는 미탐(False Negative)도 존재한다. 실험 결과에서도 알 수 있듯이 엔트로피 분석만으로는 패킹 탐지에 오탐과 미탐이 발생하기는 하지만 엔트로피는 패킹과 일반 PE 파일을 구분 짓는 특징임을 알 수 있다.

〈표 6〉 엔트로피를 이용한 패킹 탐지

Detection Method	False Positive	False Negative	Detection Count
Entropy	14/200	3/200	182/200

4.1.2 패킹 해제의 특징 탐지

패킹된 파일은 언패킹을 위해 일반 PE 파일의 섹션 속성과 차이가 있다. 3장에서 실험을 통해서 확인한 바와 같이 패킹된 PE 파일은 섹션의 Characteristics 속성인 EXECUTE, READ과 CODE, DATA는 변조가 가능하나 WRITE 속성은 꼭 필요하다. 따라서 진입점 섹션을 찾고 WRITE 속성이 포함되었는지를 확인함으로써 패킹 PE 파일을 탐지할 수 있다.

섹션의 Characteristics 속성을 사용하는 것은 Choi가 패킹을 탐지하기 위해서 섹션의 속성을 사용한 것은 동일하나, Choi는 패킹을 탐지하기 위해 패킹 파일의 여러 가지 특징을 패턴화한 반면 본 논문에서 제시한 진입점 섹션의 Write 속성은 패킹 PE 파일의 필수적인 요소로 Choi의 패턴을 대부분 포함한다.

Write 속성만을 이용하여 패킹을 탐지를 해본 결과 <표 7>과 같이 오탐이 발생하였다. 일반 PE 파일 중에서 적은 수의 파일에서 패킹이 아닌 경우에도 Write 속성을 가지고 있었다. 그러나 Write 속성은 패킹과 일반 PE 파일을 구분 짓는 특징임을 알 수 있다.

지금까지의 실험을 바탕으로 오탐을 줄이기 위해서 두 가지

〈표 7〉 Write 속성을 이용한 패킹 탐지

Detection Method	False Positive	False Negative	Detection Count
Writable	8/200	0/200	192/200

특징을 결합하여 실험하였다. 엔트로피 계산에 비해 진입점 섹션의 Write 속성을 판단하는 것이 상대적으로 빠르기 때문에 Write 속성을 먼저 검사한 후에 Entropy를 계산하여 탐지 속도의 효율성을 높일 수 있다. Write 속성과 Entropy를 이용한 패킹 탐지율 실험 결과는 다음과 같다.

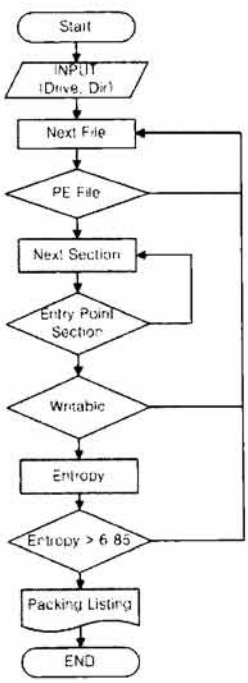
<표 8>의 결과에서 알 수 있듯이 두 개의 패킹 특징을 조합하여 패킹 탐지의 성능이 향상되었다. Write 속성만으로 탐지할 때 생기는 오탐은 엔트로피 계산을 통해서 제거할 수 있고, 엔트로피만으로 탐지했을 때의 오탐은 Write 속성을 검사함으로써 제거할 수 있다. 그런데 Write 속성이면서 엔트로피가 상대적으로 낮은 몇 개의 파일에 대한 미탐의 문제가 존재한다.

〈표 8〉 Write 속성과 Entropy를 이용한 패킹 탐지율

Detection Method	False Positive	False Negative	Detection Count
Write & Entropy	0/200	3/200	197/200

4.2 패킹 PE 탐지 방안 및 순서도

지금까지의 실험을 바탕으로 패킹 탐지 순서도는 (그림 5)와 같다.



(그림 5) 패킹 탐지 흐름도

패킹 PE 파일 탐지는 하드디스크 혹은 디렉터리를 입력으로 받는 것으로 시작한다. 입력받은 위치에서 순차적으로 하부의 파일을 검색하여 PE 파일 시그니쳐인 'MZ'를 확인한다. PE 파일인 경우 섹션의 개수를 확인하고 진입점 섹션인지를 검색한다. 진입점 섹션일 때 해당 섹션의 Characteristics 속성이 Write가 포함되었는지를 확인하여 1차 패킹 PE 파일을 선별한다. 그리고 Write 속성이 있는 섹션의 엔트로피를 계산하고 사전에 정의한 일반 PE와 패킹 PE 파일의 경계 엔트로피인 6.85와 비교함으로써 패킹을 최종 탐지한다. 이와 같은 방식은 기존의 시그니쳐 기반의 패킹 탐지 도구보다 빠르고 정확하게 패킹 PE 파일을 탐지할 수 있다.

5. 성능평가

본 장에서는 패킹 PE 파일의 탐지율과 탐지 소요 시간을 측정하여 제시한 방법으로 구현한 REMINDer의 성능을 평가한다. 비교 대상은 공개 도구인 PEID와 MRC이다.

5.1 탐지율

패킹 파일 탐지율을 측정하기 위해 공개 패킹 탐지 도구인 PEID, MRC와 REMINDer를 비교하였다. 탐지율 평가를 위해 사용한 데이터는 'Windows', 'Program Files' 디렉터리에서 추출한 일반 PE 파일 400개와 Honeypot에서 수집한 400개의 패킹된 파일이다. 그 중에서 표본 데이터로 사용한 것외의 각각의 파일 200개를 사용하였다. 패킹에 사용된 패커는 UPX, Asprotect, Molebox, Pecompack, Armadillo, Pecrypt, FSG, MEW, YodaCryptor 등이다. 탐지 결과는 다음과 같다.

<표 9>에서 보는 것과 같이 일반 파일에 대한 오탐은 세 도구 모두 없었다. 그리고 PEID와 MRC는 시그니쳐 기반으로 패킹을 탐지하기 때문에 해당 시그니쳐가 없는 경우 위와 같이 오탐이 발생하였다. 그리고 REMINDer는 2.5%의 미탐이 발생했고 그 이유는 몇 개의 패킹 파일에서 엔트로피가 상대적으로 낮았기 때문이다.

위의 결과를 통해 일부 미탐이 존재하나 본 논문에서 제시하는 패킹 PE 파일 탐지 도구의 성능이 다른 공개 도구에 비해 좋은 것을 확인할 수 있다.

〈표 9〉 패킹 파일 탐지율(%)

대상파일	조사도구	False Positive	False Negative	Detection Rate
일반 및 패킹파일	REMINDer	0	2.5	97.5
	PEID	0	12	88
	MRC	0	60.5	39.5

5.2 소요 시간

패킹 PE 파일 탐지 소요 시간을 측정하기 위해 공개 패킹 탐지 도구인 PEID, MRC와 REMINDer를 비교하였다. 탐지 시간 측정에 사용한 연산기는 Pentium4 3 GHz CPU

에 2 GBytes 메모리의 PC이다. 그리고 실험 데이터는 'Program Files' 디렉터리로 하였다. 악성코드 사고에서는 하드디스크 전체 혹은 감염 가능한 디렉터리가 조사 대상이 되므로 많은 실행 파일이 존재하는 'Program Files' 디렉터리를 대상으로 선택 하였다. 이 디렉터리에는 약 9 기가바이트(GigaBytes)의 용량에 41,555개의 파일이 존재한다. 탐지에 소요되는 시간은 <표 10>과 같다.

<표 10>의 결과에서 보는 것과 같이 제안하는 방법을 통한 탐지 도구의 성능이 월등히 좋은 것으로 확인할 수 있다. 성능평가에서는 9 기가바이트를 대상으로 탐지 시간을 측정하였지만 최근 수백 기가바이트에서 테라바이트(Tera Bytes) 용량의 하드디스크까지 저가에 널리 유통되고 있다. 테라바이트의 상황을 가정한다면 MRC의 경우 약 14시간이 소요되고 REMINDer은 약 8시간이 소요되게 된다. 제한된 시간에 많은 사고조사를 해야 한다는 현실을 감안할 때 악성코드 사고 조사에서 정확한 악성코드 탐지뿐만 아니라 신속한 탐지 또한 중요하다.

〈표 10〉 소요시간

대상파일	조사도구	시간(초)
Program Files	REMINDer	5분 8초
	PEID	7분 6초
	MRC	8분 23초

6. 결 론

악성코드 사고 조사에서 시그니쳐 기반의 안티 바이러스 제품이나 패킹 탐지 도구는 탐지 회피가 가능하기 때문에 패킹 탐지를 위한 다른 접근 방법이 필요하다.

본 논문에서는 패킹을 탐지에서 패킹의 필수적인 특징인 진입점 섹션의 'WRITE' 속성과 엔트로피를 결합한 방법을 사용하였고, 패킹 탐지 도구를 구현하였다. 공개 도구와의 실험을 통해서 제안하는 도구의 탐지율이 높고 탐지 시간이 적게 걸리는 것을 확인하였고 성능이 향상되었음을 알 수 있다. 본 논문에서 제시한 패킹 파일 탐지 도구를 활용하면 악성코드 사고조사에서 신속하게 증거 파일을 탐지할 수 있을 것이다.

향후에는 패킹된 파일 중에서 낮은 엔트로피를 갖는 패킹 파일에 대한 탐지와 패킹된 파일 중에서 악성 여부를 판단하는 방안에 대해 연구할 예정이다.

참 고 문 헌

[1] Nwokedi Idika, Aditya P. Mathur, "A Survey of Malware Detection Techniques", Purdue University, 2007.

[2] Mihai Christodorescu, Somesh Jha, Johannes Kinder, "Software transformations to improve malware detection", Journal in Computer Virology, Springer, pp.253-265, 2007.

[3] Fanglu Guo, Peter Ferrie, Tzi-cker Chiueh, "A Study of the Packer Problem and Its Solutions", Recent Advances in Intrusion Detection, Springer, pp.98-115, 2008.

[4] T. Brosch and M. Morgenstern, "Runtime Packers: The Hidden Problem", Proc. Black Hat USA, Black Hat, 2006; https://www.blackhat.com/presentations/bh-usa-06/BH-US-06-Morgenstern.pdf

[5] Robert Lyda, James Hamrock, "Using entropy analysis to find encrypted and packed malware", Security & Privacy, IEEE Vol.5, Issue2, pp.40-45, 2007.

[6] Yang-seo Choi, Ik-kyun Kim, Jin-tae Oh, Jae-cheol Ryou, "PE File Header Analysis-Based Packed PE File Detection Technique (PHAD)," International Symposium on Computer Science and its Applications, pp.28-31, 2008.

[7] Peid hompage, http://www.peid.info

[8] MRC homepage, http://www.mandiant.com/mrc/

[9] 이호동, Windows 시스템 실행파일의 구조와 원리, 한빛출판사, pp.1-30, 2005.

[10] James M. Aquilina, Eoghan Casey, Cameron H. Malin, "Malware Forensics - Investigating and Analyzing Malicious Code", Syngress, pp.140-151, 2008.

[11] Thomas M. Cover and Joy A. Thomas, "Elements of Information Theory", Second Edition. Wiley Interscience, pp. 1-16, 2006.

[12] Nenad Stojanovski, Marjan Gusev, Danilo Gligoroski, Svein.J.Knapskog, "Bypassing Data Execution Preventioni on MicrosoftWindows XP SP2", Second International Conference on Availability Reliability and Security(ARES '7), 2007.

한 승 원

e-mail : normalhan@korea.ac.kr

2003년 건양대학교 정보전산학과(학사)

2008년 고려대학교 정보경영공학과(석사수료)

관심분야 : 컴퓨터 및 네트워크 포렌식, 악성코드 사고 조사, 봇넷 분석 등

이 상 진

e-mail : sangjin@korea.ac.kr

1987년 고려대학교 수학과(이학사)

1989년 고려대학교 수학과(이학석사)

1994년 고려대학교 수학과(이학박사)

1989년 10월~1999년 2월 ETRI 연구원 역임

1999년 3월~현 재 고려대학교 정교수

관심분야 : 디지털 포렌식, 모바일 포렌식, 심층 암호, 해쉬 함수 등