제 3장 디지털 증거 분석기법

1. 디스크 구조와 파일시스템

디스크에 파티션을 나누거나 하나로 사용하더라도 반드시 디스크의 시작 위치(섹터)에는 MBR이라는 구조가 생성됨
MBR에는 컴퓨터 전원이 켜지면 'BIOS' 점검과 설정을 하기 위한 코드가 담겨있고 파티션 테이블 구조를 포함

파일 시스템은 컴퓨터 디스크 내에 파일들을 일종의 색인 작업을 하여 쉽게 접근할 수 있도록 관리하는 핵심요소
파일과 관련된 메타데이터에 기록된 정보를 토대로 파일의 위치정보 및 상세 정보를 알 수 있음 -> 파일시스템에 기록, 파일시스템의 역할

디스크에 운영체제를 설치하기 앞서 디스크 파티션을 나누고 파일시스템 설치과정을 진행함 -> 설치 완료 후 디스크 상 파일시스템의 시작 위치(섹터)에 'VBR'이 존재
파일 시스템의 구조 (메타 영역:메타데이터의 정보를 가짐 / 데이터 영역:실제 파일의 데이터를 가짐)

디지털포렌식 도구들은 기본적으로 모두 파일시스템 분석
파일시스템 분석이 이루어지고 이후에 아티팩트 혹은 관련 정보들을 분석

1.1. MBR(Master Boot Record)

파티션된 하드 디스크의 첫 섹터에 존재하며 파티션 섹터라고도 불림
기본적인 부팅과정 이벤트 순서 : 컴퓨터를 키면 가장 처음으로 MBR을 읽어와 BIOS 검사, 설정을 한 뒤 파티션 테이블 정보를 확인하여 부팅 가능한 파티션의 위치로 이동

MBR에서 가장 중요한 요소는 "디스크 서명"과 "파티션 테이블"
디스크 서명은 MBR 시작 위치로부터 440byte 이동한 위치(440~444byte), 읽을 때는 x86 아키텍쳐에서 리틀-엔디언 방식을 사용
파티션 테이블은 MBR 시작 위치로부터 446byte 이동한 위치(446~510byte)
파티션은 연속된 저장공간을 하나 이상의 연속되고, 독립된 영역으로 나누어 사용할 수 있도록 정의한 규약, 하나의 하드디스크를 여러 개의 파티션으로 나누어 논리적으로 여러 개의 하드디스크인 것처럼 사용 가능

파티션 작업을 한 디스크에는 MBR의 파티션 테이블 안에 주 파티션 엔트리들이 있으며 하나의 파티션 엔트리는 파티션 테이블에서 16바이트를 할당 받음

[확장 파티션 - EBR (Extended Boot Record)]

확장 파티션을 사용하면 확장 파티션을 통해 논리 드라이브를 무제한으로 생성 가능

주 파티션 3개를 보존하고 나머지 1개 엔트리로 확장 파티션으로 구성, 확장 파티션을 논리 파티션으로 표현하여 추가적인 파티션 구성

-확장 파티션 : 디스크에는 하나의 확장 파티션만 포함될 수 있지만 확장 파티션은 여러 개의 논리 파티션으로 세분화 가능

-논리 드라이브 : 논리 파티션이라고 하고, 기본 디스크의 확장 파티션에서 생성된 볼륨 논리 드라이브는 형식을 지정하고 드라이브 문자를 할당 받을 수 있지만, 운영체제를 설치하고 호스팅 할 수 없음

[EBR (Extended Boot Record) 구조]

MBR의 기본 구조와 동일하지만, 실제 사용은 파티션 테이블 부분만 사용

BPB영역인 446바이트 까지는 보통 '0'으로 채워짐, 첫 번째 파티션 엔트리는 해당 확장 파티션의 위치가 기록, 두 번째 타피션 엔트리는 추가 확장 파티션인 EBR의 위치가 기록, 세 번째, 네 번째 파티션은 사용되지 않고 '0'으로 채워짐

파티션 엔트리의 구조는 MBR의 파티션 엔트리와 동일

1.2. VBR (Volume Boot Record)

볼륨 부트 레코드는 파티션이 저장된 경우 파티션의 시작 섹터, 일정 섹터만큼 이동된 섹터에 존재, 파일시스템이 설치되는 섹터 위치

파티션이 저장되지 않은 USB같은 경우는 디스크의 첫 섹터에 VBR이 존재

파일시스템을 나타내는 시그니처, 섹터의 크기, 클러스터의 크기, 볼륨의 전체 크기, 볼륨 시리얼 번호, 파일이 저장되는 위치 및 정보를 가지는 메타파일의 위치 정보 등을 알 수 있음

1.2.1. FAT (File Allocation Table) 파일시스템

첫 번째 섹터에는 볼륨 부트 레코드가 존재, 뒤로 예약 영역, 파일 할당 테이블, 루트 디렉토리, 데이터 영역이 존재

|포렌식 관점|
FAT 파일 시스템 VBR 내부 구조의 주요 항목
파일 시스템에는 백업 VBR이 존재, FAT의 경우에는 시작 위치에서 6섹터 이동한 위치에 백업 VBR이 존재
VBR 백업본 위치는 BPB에 기록되어 있음

1.2.2. NTFS(New Technology File System) 파일 시스템

NTFS는 윈도우 NT 계열 운영체제의 파일시스템으로 윈도우 2000, 윈도운 XP 이후 최신 버전 등에도 포함됨

-메타 파일
파일시스템의 주요 자료에 대한 정보가 저장되어 있는 파일들 ($MFT, $Logfile, $UsnJrnl)

-ADS (Alternate Data Stream)
NTFS 파일 시스템에만 존재하는 추가적으로 생성되는 데이터스트림
다중 데이터스트림을 지원하고 파일의 요약정보나 데이터를 기록/저장 가능

2. Registry

마이크로소프트 윈도우 32/64비트 버전과 윈도우 모바일 운영체제에서 설정 등의 특정 항목을 담고 있는 데이터베이스
키와 서브 키, 값 세가지 기본 요소를 갖으며 다섯 개의 루트 키가 있고, 마스터 키와 유도 키로 나눌 수 있음

2.1. 시스템 정보

레지스터리에는 윈도우 버전, 설치 일시, 타임존, 사용자 계정, 네트워크, 공유 폴더, 서비스 목록, 응용 프로그램 설치/제거 정보 등이 기록되어 있음
레지스트리 벨류의 데이터는 유니코드, 평문 등으로 레지스터리 키의 벨류마다 다르게 저장됨

2.2. 사용자 행위 분석

레지스트리에는 사용자 행위에 대한 프로파일 정보를 기록

사용자의 선호 사용 파일 및 응용 프로그램, 사용자가 입력한 키워드 정보, 공유, 사이트 접속 기록 등도 알 수 있다.

사용자 행위에 대한 정보를 저장하고 있는 레지스트리 하이브 파일은 사용자 홈 폴더에 존재하는 “NTUSER.DAT” 파일

2) Comdlg32

설치된 프로그램을 통해서 실행된 파일 또는 저장된 파일 정보

포렌식 관점

Comdlg32 레지스트리 키 하위로 존재하는 서브 키들 중 ‘LastVisitedPidlMRU’와 ‘OpenSavedPidlMRU’키에 사용자 행위가 기록되어 있음

2.3. 외부저장매체

윈도우 운영체제에서 외부장치를 연결하는 경우 레지스트리에 장치에 대한 정보가 기록됨

기록된 정보는 다양한 레지스트리 키와 값에서 확인 가능

3. Windows Artifact

3.1. 프리패치(Prefetch) / 슈퍼패치(Superfetch)

프리패치는 윈도우XP 이후의 운영체제에서 관리하는 메모리 정책으로, 실행 파일을 메모리에 로딩할 때 빠르게 참조할 목적으로 개발됨

슈퍼패치는 사용자의 프로그램 사용 패턴을 기록하여 자주 사용하는 프로그램일 경우 지속적으로 메모리에 상주시키는 것을 목적으로 추가됨

프리패치 파일은 ‘[실행 파일 이름]-[해시].pt’ 이름으로 생성되고, ‘*.pf’ 확장자,

슈퍼패치 파일은 ‘*.db’ 확장자

|포렌식 관점|

프리패치 파일 내부에는 원본 실행 파일이 최근까지 ‘실행된 시간 정보’와 ‘실행된 횟수’, ‘실행된 파일 이름’과 원본 실행 파일에 의해 사용되는 ‘DLL 파일들의 이름’이 기록되어 있음

3.2. 바로가기(Link) 파일

원본 파일의 위치를 포함하고 있는 파일로 어디에나 생성될 수 있음
원본 파일의 이름으로 생성되고 '*.lnk' 확장자

3.3. 점프리스트(Jumplist)

윈도우 Vista 버전부터 추가된 기능으로 윈도우가 실행된 화면 하단의 작업표시줄에 실행된 프로그램에서 최근 실행된 목록들을 기록하고 있는 아티팩트
점프리스트 파일은 다른 두 곳의 경로에 존재하고 '*.automaticdestinations-ms'와 '*.customdestinations-ms' 확장자를 갖음

|포렌식 관점|
점프리스트 파일은 OLE컴파운드 파일과 같이 내부에 여러 스트림으로 구성되어 있음
각각의 스트림들은 윈도우 바로가기 파일과 유사한 구조로 저장됨

3.4. 휴지통(Recycle.bin)

사용자에 의해 삭제된 파일들을 저장
운영체제 내부적으로 별도의 폴더 형태로 존재, 윈도우 사용자 계정 별로 각각 존재

3.5. 이벤트 로그(Event Log)

윈도우 운영체제에서 발생되는 특정 이벤트를 체계적으로 기록한 바이너리 로그 파일
주로 침해사고 조사에 많은 초점을 두고 분석

|포렌식 관점|
분석은 각 로그에 기록되어 있는 'Event ID'로 조회하여 확인
발생된 이벤트와 프로그램에 대한 정보, 사용자 로그인과 관련된 기록 등의 정보 저장
악성코드가 실행될 때 대게는 시스템에 문제를 발생시키는 경우가 많아 에러나 경고 로그 등을 우선적으로 분석하여 침해사고 발생 시 악성코드에 대한 정보를 파악 가능
Application.evtx / System.evtx / Security.evtx

3.6. setupapi.dev.log

시스템과 연결된 매체정보를 저장하고 있는 파일로 하드웨어 장치가 시스템에 최초 연결되어 드라이버 설치를 진행함으로써 해당 내용을 기록함

평문으로 저장되어 텍스트 편집기로 확인 가능

3.7. Recentfilecache.bcf (Windows 7)

운영체제 버전에 따른 호완성 문제를 해결하기 위해 윈도우가 캐시한 데이터를 기록한 파일

프로세스가 생성될 때 실행 파일의 경로를 임시로 저장하기 위한 용도로 사용됨

3.8. Amcache.hve

윈도우 8부터 생성된 파일로 'Recentfilecache.bcf' 파일을 대체하기 위한 파일

최근 실행된 프로그램 정보를 저장, 레지스트리 하이브 파일과 구조가 동일

포렌식적으로 가치 있는 폴더는 'File'

3.9. 썸네일 캐시 (Thumbnail Cache)

윈도우 시스템에서 보여지는 이미지들을 썸네일 데이터베이스 파일로 저장

썸네일 파일은 'thumcache_X.db' 이름으로 생성 (X는 이미지의 크기를 나타냄)

3.10. 아이콘 캐시 (Icon Cache)

윈도우 8 이후 버전부터 아이콘과 관련된 이미지들을 데이터베이스 파일에 저장

썸네일 파일과 동일한 형태 'iconcache_X.db' 이름으로 생성됨

썸네일과 다르게 오직 아이콘 이미지만 저장되어 있음

3.11. NTFS 메타(Meta) 파일

NTFS 파일시스템 볼륨의 핵심 메타 파일로는 '$MFT(Master File Table), $Logfile, $UsnJrnl'이 있음

NTFS 파일시스템에만 존재하는 파일로 파일시스템에서 파일들의 이벤트 이력 등을 관리, 저장

'$MFT'는 시스템 내에 모든 폴더와 파일 메타 정보를 포함, '$Logfile'과 '$UsnJrnl'는 파일의 생성, 수정, 삭제에 대한 트렌잭션 과정을 저장

3.12. 시그니처(Signature) 분석

파일의 확장자만으로 파일을 식별하는 방법이 아닌 파일 고유의 데이터를 확인하여 파일의 종류를 식별

모든 파일은 파일 고유 포맷의 특정 바이트로 이루어져 있는 파일 시그니처가 존재, 일반적으로 '매직 넘버'라고도 함

보통은 파일의 첫 바이트부터 일정 부분만큼을 파일의 시그니처로 지정하기도 하고 마지막 부분을 지정하기도 함 (헤더 시그니처 / 푸터 시그니처)

3.13. 해시(hash) 분석

파일의 데이터를 해시 함수, 알고리즘을 통해 계산하여 산출된 데이터

파일의 고유한 값으로 서로 다른 두 파일의 해시 데이터가 일치한다는 것은 두 파일은 동일한 파일임을 의미

종류로는 MD5, SHA-1, SHA-256, SHA-512, CRC32 등